RMIS View/Print Document Cover Sheet

This document was retrieved from the Documentation and Records Management (DRM) ISEARCH System. It is intended for Information only and may not be the most recent or updated version. Contact a Document Service Center (see Hanford Info for locations) if you need additional retrieval information.

Accession #: D196079529

Document #: SD-534-CSWL-001

Title/Desc:

SEQUENCE & BATCH LANGUAGE PROGRAMS & ALARM RELATED
C PROGRAMS FOR THE 242A MCS [SEC 1 OF 6 PAGE 1 TO
126]

Pages: 132

This document was too large to scan as a whole document, therefore it required breaking into smaller sections.

Document number: SD-534-CSWL-001

Section 1 of 6

Title: Sequence and Batch Language programs
and Alarm Related "C" Programs for the
242-A MCS

Date: 4/15/96 Revision: 3

Originator: Berger, J.F
Co: _____

Recipient: _____
Co: _____

References: ECN-627504

COMPLETE

ENGINEERING CHANGE NOTICE

Page 1 of 21. ECN 627504Proj.
ECN

2. ECN Category (mark one) Supplemental ☐ Direct Revision ☒ Change ECN ☐ Temporary ☐ Standby ☐ Supersede ☐ Cancel/Void ☐		3. Originator's Name, Organization, MSIN, and Telephone No. JF Berger, EPPE, R1-43, 3-4563		3a. USQ Required? ☐ Yes ☒ No		4. Date 04/08/96	
5. Project Title/No./Work Order No. 242-A MCS SABL Programs		6. Bldg./Sys./Fac. No. 242-A		7. Approval Designator N/A		8. Document Numbers Changed by this ECN (includes sheet no. and rev.) WHC-SD-534-CSWL-001, REV. 2	
9. Related ECN No(s). N/A		10. Related PD No. N/A		11a. Modification Work ☐ Yes (fill out Blk. 11b) ☒ No (NA Blks. 11b, 11c, 11d)		11b. Work Package No. N/A	
11c. Modification Work Complete N/A		11d. Restored to Original Condition (Temp. or Standby ECN only) N/A		12. Description of Change This ECN revises document WHC-SD-534-CSWL-001 Rev. 2 in its entirety and changes the revision to 3.		13a. Justification (mark one) Criteria Change ☒ Design Improvement ☐ Environmental ☐ Facility Deactivation ☐ As-Found ☐ Facilitate Const ☐ Const. Error/Omission ☐ Design Error/Omission ☐	
13b. Justification Details Incorporating all outstanding ECNs into the document.		14. Distribution (include name, MSIN, and no. of copies) JF Berger R1-43 1 M. Teats R1-43 1 Sta #15 E6-12 Sta #16 S5-00		RELEASE STAMP DATE: 4-15-96 STA #16 HANFORD RELEASE 16			

627504

15. Design Verification Required	16. Cost Impact						17. Schedule Impact (days)	
	ENGINEERING			CONSTRUCTION				
	Additional	☐	\$ N/A	Additional	☐	\$ N/A	Improvement	☐ N/A
	Savings	☐	\$ N/A	Savings	☐	\$ N/A	Delay	☐ N/A
☒ Yes								
☐ No								

18. Change Impact Review: Indicate the related documents (other than the engineering documents identified on Side 1) that will be affected by the change described in Block 12. Enter the affected document number in Block 19.

SDD/DD		Seismic/Stress Analysis		Tank Calibration Manual	
Functional Design Criteria		Stress/Design Report		Health Physics Procedure	
Operating Specification		Interface Control Drawing		Spares Multiple Unit Listing	
Criticality Specification		Calibration Procedure		Test Procedures/Specification	
Conceptual Design Report		Installation Procedure		Component Index	
Equipment Spec.		Maintenance Procedure		ASME Coded Item	
Const. Spec.		Engineering Procedure		Human Factor Consideration	
Procurement Spec.		Operating Instruction		Computer Software	
Vendor Information		Operating Procedure		Electric Circuit Schedule	
OM Manual		Operational Safety Requirement		ICRS Procedure	
FSAR/SAR		IEFD Drawing		Process Control Manual/Plan	
Safety Equipment List		Cell Arrangement Drawing		Process Flow Chart	
Radiation Work Permit		Essential Material Specification		Purchase Requisition	
Environmental Impact Statement		Fac. Proc. Samp. Schedule		Tickler File	
Environmental Report		Inspection Plan			
Environmental Permit		Inventory Adjustment Request		None	

19. Other Affected Documents: (NOTE: Documents listed below will not be revised by this ECN.) Signatures below indicate that the signing organization has been notified of other affected documents listed below.

Document Number Revision

N/A

20. Approvals

Signature _____
OPERATIONS AND ENGINEERING

Date _____

Signature

Date _____

Cog. Eng. JF Berger

4-8-96

PF

Cog. Mgr. RJ Nicklas

4/11/96

QA

QA

So

Safety

Design

Environ.

Exhibits

Other Informal Reviewer

4-11-96

Other

DEPARTMENT OF ENERGY

Signature or a Control Number that tracks the Approval Signature

ADDITIONAL

SEQUENCE AND BATCH LANGUAGE PROGRAMS AND ALARM RELATED "C" PROGRAMS FOR THE 242-A MCS

John F Berger
Westinghouse Hanford Company, Richland, WA 99352
U.S. Department of Energy Contract DE-AC06-87RL10930

EDT/ECN: 627504 UC: 2030
Org Code: 77310 Charge Code: N17A992
B&R Code: EWJ3/20071 Total Pages: 131 733

Key Words: Distributive Control System, Monitoring and Control System,
242-A Evaporator, SABL

Abstract: A Distributive Process Control system was purchased by Project B-534, "242-A Evaporator/Crystallizer Upgrades". This control system, called the Monitor and Control system (MCS), was installed in the 242-A evaporator located in the 200 East Area. The purpose of the MCS is to monitor and control the Evaporator and monitor a number of alarms and other signals from various Tank Farm facilities. Applications software for the MCS was developed by the Waste Treatment Systems Engineering (WTSE) group of Westinghouse.

The standard displays and alarm scheme provide for control and monitoring, but do not directly indicate the signal location or depict the overall process. To do this, WTSE developed a second alarm scheme

TRADEMARK DISCLAIMER. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof or its contractors or subcontractors.

Printed in the United States of America. To obtain copies of this document, contact: WHC/BCS Document Control Services, P.O. Box 1970, Mailstop H6-08, Richland WA 99352, Phone (509) 372-2420; Fax (509) 376-4989.

Samuel K Salazar 4-15-96
Release Approval Date

Release Stamp

Approved for Public Release

**SEQUENCE AND BATCH LANGUAGE PROGRAMS
AND ALARM-RELATED 'C' PROGRAMS
FOR THE 242-A MCS**

Prepared By:
EPPE

TABLE OF CONTENTS

1.0	INTRODUCTION	5
2.0	PURPOSE	7
3.0	SCOPE	7
4.0	DEFINITIONS	8
5.0	DESCRIPTION	11
5.1	Alarm Schemes	11
5.1.1	GSE Alarm Scheme	11
5.1.2	New Alarm Scheme	11
5.2	Sequence Units and IPN Order	12
5.3	RPFs	13
5.4	UPFs	13
5.5	Sequence Database	14
5.6	Pointers, Tables, And How The Programs Work	14
5.6.1	Pointers	14
5.6.2	Procedure Used By All Programs	16
5.6.3	PCMI DIN Alarms	17
5.6.4	PCMO DIN Alarms	18
5.6.5	PCMO AI Red and Yellow Alarms	19
5.6.6	PCMO AI White Alarms	20
5.6.7	Device Alarms	22
5.7	White Alarm Summary Program	22
5.8	Unit Relative Feature of Programs	24
5.9	Program Statuses, Diagnostics, and Corrective Action	24
5.10	"C" Programs Related To Alarm Scheme	25
5.11	FIGURE 1. Program Relationships	27
5.12	Other Alarm-Related "C" Programs	28
5.12	SKID Key Configuration	28
6.0	REBUILDING DATABASES AND PROGRAMS	29
7.0	APPENDICES	30
	APPENDIX A - IPN ORDER	31
	APPENDIX B -SEQUENCE DATABASE (SDB)	60
	PCMOASDB	60
	PCMIASDB	70
	APPENDIX C - RECIPE PROTOTYPE FILES (RPF)	74
	ALLO	74

WMC-SD-534-CSWL-001 REV 3

242A	88
ALL1	96
FARM	107
EMPTY	116
APPENDIX D - UNIT PROTOTYPE FILES (UPF)	127
TOTALO	127
SHORTO	148
PRCESS	160
OVVIEW	174
TOTAL1	200
TNKFRM	230
BLANKK	253
APPENDIX E - SEQUENCE AND BATCH LANGUAGE (SABL)	263
224BATCH	263
244HVAC	276
WSTXFR	287
350BATCH	298
FEEDBATCH	307
CA1BATCH	321
PB1BATCH	333
PB2BATCH	345
EA1BATCH	360
EC1BATCH	373
RWBATCH	384
URWBATCH	398
VENTBATCH	410
PCBATCH	423
IXBATCH	438
SCBATCH	449
BASINBAT	461
UTILBATCH	471
AMUBATCH	483
LERFBATCH	496
CASSBATCH	508
TAG	512
TIMEINHA	514
PGMOA	516
WHITE1	521
ANBATCH	555
APBATCH	570
AWBATCH	589
XYZBATCH	604
242BATCH	620
PGM1A	630
APPENDIX F - "C" PROGRAMS FOR ALARMS	634
DOGOA	634

WHC-SD-534-CSWL-001 REV 3

DOG0B	637
DOG1A	640
DOG1B	644
WATCH	648
INHA	656
UKF14	660
UKF34	671
UKF43	680
UKF44	690
APPENDIX G - STRING PACKET LISTING	696
APPENDIX H - REBUILD PROCEDURE	716
APPENDIX I - SKID KEY CONFIGURATION	725

1.0 INTRODUCTION

A GSE Process Solutions Inc. (GSE) D/3¹ Distributive Process Control system was purchased by Project B-534, "242-A Evaporator/Crystallizer Upgrades". This control system, called the Monitor and Control System (MCS), was installed in the 242-A Evaporator located in the 200 East Area. The purpose of the MCS is to monitor and control the Evaporator and monitor a number of alarms and other signals from various Tank Farm facilities. Applications software for the MCS was developed by the Waste Treatment Systems Engineering (WTSE) group of Westinghouse.

The standard GSE displays and alarm scheme provide for control and monitoring, but do not directly indicate the signal location or depict the overall process. To do this, WTSE developed a second alarm scheme which uses special programs, annunciator keys, and process graphics. The special programs are written in two languages; Sequence and Batch Language¹ (SABL), which is a proprietary language of GSE, and "C" language. The WTSE-developed alarm scheme works as described below:

- o SABL relates signals and alarms to the annunciator keys, called SKID keys. When an alarm occurs, a SABL program causes a SKID key to flash, and if the alarm is of yellow or white priority then a "C" program turns on an audible horn (the D/3 system uses a different audible horn for the red priority alarms). The horn and flashing key draws the attention of the operator.
- o The SKID keys relate the alarms to custom graphic displays, which depict the process much like the Process and Instrumentation Engineering Diagrams (P&IDs). When an operator presses a flashing SKID key, a graphic showing an alarm message is displayed.
- o Each graphic contains a menu, which may be used to direct the operator to other more detailed displays.
- o When an alarm is acknowledged from a graphic display, user-written "C" code manages the acknowledgement of all alarm messages in that graphic and sequence unit throughout the system (on other D/3 displays such as faceplates, P1, and P2 displays). Likewise, when an alarm is acknowledged from a faceplate display or other D/3 displays the corresponding alarm messages are acknowledged in the appropriate graphics.

The engineering tools used to develop the alarm scheme includes following GSE D/3 utilities; the Management of Database Elements (MDL) utility, the Recipe Management Subsystem (RMS), the Unit Prototype File (UPF) utility, the Sequence Database (SDB) utility, the Sequence Instruction

---

¹ Trademark of GSE Process Solutions Inc.

Compiler (SIC) utility, the String Editor (SED), and the Sequence Key Interactive Definer Utility (SKID). Additionally, the source code of user-written "C" programs are compiled to create task images. Finally, a custom graphics development package installed on a personal computer is used to develop the graphics for the D/3 system. Development of the WTSE alarm scheme from scratch is summarized below. Once developed, modifications will involve some combination of the steps below depending the extent of the modification.

- o Signals are defined in the continuous database using the MDL utility. Signals in the continuous database are grouped into process areas or systems by declaring the Internal Point Number (IPN) for each signal. Usually, the IPN of a signal should not be defaulted.
- o Variables used in sequence programs are defined using the RMS and UPF utilities. The unit prototype files declare the associated recipe prototype files.
- o The sequence databases are built using the SDB utility, which defines the sequence units, the first External Point Name (EPN) of each signal type (analog or digital), the total number of each signal type in a sequence unit, and the associated UPF.
- o SABL programs are written and compiled.
- o String packets are created using SED. The string packets contain the custom alarm messages for the graphics.
- o The annunciator key layout is defined.
- o Graphics are created on a personal computer, down-loaded to the D/3, and compiled.
- o "C" programs are written and compiled.
- o All databases are loaded and all programs are installed and run. At this point, the alarm scheme may be tested to verify functionality.

2.0 PURPOSE

The purpose of this document is to provide a hardcopy of the software for the alarm scheme developed by WTSE. Revision 1 of this document contains listings of software contained in revision 0 of the magnetic media (WHC-SD-534-CSWD-008).

3.0 SCOPE

This document provides the following information:

- o A list of the IPN order of the signals required to construct the sequence databases, and lists the EPNs contained in each sequence unit.
- o A list of the sequence databases.
- o A list of the sequence program variables. These variables are defined in the unit prototype files (UPF) and recipe prototype files (RPF).
- o Listings of SABL and "C" programs that manage alarms in the WTSE alarm scheme and control the MCS graphics.
- o The custom alarm messages (string packets) created with SED for the graphics.
- o The layout of the annunciator keys.
- o Listings of "C" programs for inhibiting and de-inhibiting certain evaporator process signals.
- o Discussion of programs and databases where appropriate.

The Tag List (WHC-SD-534-SWD-001) defines any alarms associated with a signal.

It is assumed that the reader has some programming background and has a good understanding of the process of developing applications for the MCS. It is assumed that the reader is already familiar with the GSE continuous control package for D/3 version 3.3, is familiar with the operating system of the host computer, and understands SABL coding. Additionally, the reader should be familiar with "C" language programming. Although not necessary for this document, the reader should be familiar with the development of graphics, the string packets, and annunciator key layout.

4.0 DEFINITIONS

Analog Input (AI):

A signal indicating analog values such as temperature, pressure, or liquid flows. Usually a 4 to 20 mA or 0 to 100 mV signal which is converted by the MCS to engineering units whose range represents 0 to 100% of the calibrated range of a transmitter. Sometimes used to control flow rates, pump speeds, liquid levels, or temperatures by analog loop which provides a 4 - 20 mA output signal. For thermocouples (T/Cs) or resistance temperature devices (RTDs), the input range is roughly -10 to 100 mV, where negative base volts corresponds to freezing temperatures.

Analog chain:

An arrangement of software process control blocks, or database blocks; to handle analog signal addressing, scaling, alarming, and process control.

Continuous control database:

An arrangement of digital and analog signals into AI inputs, digital inputs (DINs), master devices (MDVs), and device (DEVs) which are named by EPN and ordered by IPN. Includes alarm limits for the critical (red) and non-critical (yellow) alarms. Alarm limits for informational supervisory (white) alarms are imbedded in SABL programs.

Device (DEV):

For control of valves and motors by means of fixed logic, including interlock logic (see also master device). Uses some combination of analog inputs, digital inputs, and command (operator) keyboard inputs. Provides one or more digital output (DO) signals, and/or provides a software input to another logic device or analog chain. Devices usually have more than two statuses.

Digital Input (DIN):

A discrete (on or off) signal using 24 VDC or 110 VAC. Used for leak detectors, radiation alarms, valve positions, motor statuses, and so on.

Display Control Modules (DCM):

A computer which communicates with the process control modules and creates all displays appearing on the operator workstations.

External Point Name (EPN):

A name, up to nine characters long, assigned to each AI, DIN, MDV, and DEV defined in the continuous control database.

Faceplate:

A standard GSE display that shows the primary indications of a single EPN. Up to 16 EPNs can be grouped together and displayed simultaneously on one operator workstation screen. A grouping of up to 16 EPNs in a single display is sometimes called a faceplate group, or just faceplate. The faceplate group display is also a standard GSE display.

Graphic:

A display created by engineering using a graphics development package installed on a personal computer to show the relationship of signals to each other and their alarms. Usually, the graphic appears much like a process flow diagram.

Internal Point Number (IPN):

A number assigned to each signal type (AI, DIN, MDV, or DEV) which locates the EPN within the continuous database. Sequence programs scan signals of one type in order of the IPN.

Master Device (MDV):

A block of database containing the actual boolean logic programming for control of valves and motors. The boolean logic program statements usually are a series of gates (EQUATE, AND, OR, XOR, NOT, FLIP FLOP, TDE, TDD, or OS).

Operator Control Module (OCM):

An operator console, or workstation, used to monitor and control Evaporator and Tank Farm signals. An OCM displays information such as faceplates, alarm lists, graphics, and trends.

Process Control Module (PCM):

Redundant computers that monitors all signals and runs all SABL programs and certain user-written "C" programs (DOGOA, DOGOB, DOG1A, DOG1B). A PCM performs all process control functions.

Recipe Prototype File:

A file which lists flags, integers, floating points, pointers, timers, and string variables used in sequence programs. The file name is 4 characters long. Each SABL program uses variables found in one UPF or one RPF.

Sequence and Batch Language (SABL):

A GSE proprietary language that can access all blocks of the continuous database with puts and fetches.

Sequence Database (SDB):

A utility that relates continuous control database signals to sequence variables and sequence units. The SDB utility defines the sequence units, the first EPN of each signal type, the total number of each signal type in a sequence unit, and the associated UPF.

Sequence Key Interactive Definer Utility (SKID):

Usually relates displays to one of 64 annunciator keys on an OCM. A SKID key can be defined to run a sequence program, run a "C" program, change a sequence variable, command a device, and bring up a display by pressing the key. The light of each key can be off, on, or flashing as commanded by a SABL program.

Sequence Unit:

All signals in one process area or system are ordered into a sequence unit to allow access by the sequence database. Example sequence units are PB1, PB2, CA1, EA1, and EC1.

Signal type:

AI, DIN, MDV, or DEV.

String Editor (SED):

A utility used to create alarm messages, or strings, in graphics that change depending on the value of an integer variable in the sequence database. The application of SED is the only way to create flashing alarm messages in a graphic.

Unit Prototype File (UPF):

A file which lists scratch pad variable flags, integers, floating points, timers, pointers, and string variables used by SABL programs. The file name is 6 characters long. Each UPF must have a RPF assigned to it. Each SABL program uses variables found in one UPF or one RPF.

5.0 DESCRIPTION

5.1 Alarm Schemes

GSE provided a standard D/3 alarm scheme and WTSE developed a new alarm scheme. The new scheme adds functionality to the scheme provided by GSE. Both schemes are used (the new scheme does not replace the GSE scheme) in the complete management of alarms.

5.1.1 GSE Alarm Scheme

The MCS has two alarm systems, P1 (critical, red) and P2 (non-critical, yellow), that are part of the GSE package. Critical alarms are indicated by a flashing red P1 on all OCM screens and produce an audible alarm. The audible alarm is silenced by pressing the "Silence" key on the OCM keyboard. A red alarm summary list, up to 4 pages long, is displayed by pressing the P1 operator console key. Non-critical alarms are indicated by a flashing yellow P2 on all OCM screens. There is no audible alarm for the P2 alarms as defined in the GSE software package. A yellow alarm summary list, up to 4 pages long, is displayed by pressing the P2 operator console key. In the P1 and P2 alarm summary lists, acknowledged alarms appear steady, unacknowledged alarms flash, and unacknowledged alarms which have cleared appear flashing green.

5.1.2 New Alarm Scheme

The GSE alarm package was considered inadequate by itself, requiring the development of an additional alarm scheme. The new alarm scheme had to satisfy the following 4 requirements:

- o Graphics must roughly identify alarm locations.
- o Graphics must display alarms using customized flashing alarm messages that are acknowledgeable from the graphic.
- o Include a third alarm priority level called informational, or advisory (white) alarms.
- o Provide audible alarms for the yellow and white alarms.

The D/3 system critical alarm horn is activated for red alarms. The new alarm scheme commands a device to turn on a separate horn if a yellow or white alarm occurs, and to flash a SKID key for any alarm that occurs. Both horns are silenced by pressing the "Silence" key on the OCM keyboard. Graphics are displayed by pressing one of 64 annunciator keys. When any alarm occurs, the SKID key associated with the alarm flashes. Pressing this key displays the graphic. The alarm is shown by a flashing message in the appropriate alarm

color; red, yellow, or white. At the same time, an alarm message is printed at system alarm typer. The Tank Farm graphics show a list of the last 5 alarms at the bottom of each screen. All alarms on one graphic are acknowledged system wide by pressing the acknowledge key. Conversely, alarms acknowledged at the faceplate or alarm list level are also acknowledged on the graphic. White alarms also light a White Alarm annunciator key. Pressing this key displays up to three pages of white alarms.

5.2 Sequence Units and IPN Order

Forty SKID keys have been assigned to 40 graphics involving 33 sequence units using the new alarm scheme described in Section 5.1.2. For the Evaporator, EPNs in each graphic are assigned to one sequence unit, are controlled by one SABL program, and are associated with one SKID key. Tank farm graphics differ somewhat in that one SABL program usually controls 4, and up to 5, separate SKID keys.

The continuous database is structured by the order in which EPNs are defined. This order is established by the IPN of each EPN. This structure creates the foundation for the sequence database and sequence units. A table of EPNs, IPNs, and Sequence Units is found in the IPN order list (Appendix A). Appendix A is arranged by sequence unit and IPN. The "First Graph" column shows the graphic number where the EPNs are found. The column "ACAD File Name" indicates the personal computer (PC) file name for the graphic. The PC files are found in directory D:\A386\DRAW with ".DWG" extensions. The column "Number per Seq Unit" shows the total IPNs allowed in each unit for the AI, DIN, and DEV signal types. The total allowable IPNs are defined in the sequence database (see Appendix B, and Section 5.5). The "PCM", "MCS EPN", "SIG TYPE", and "IPN" columns are self explanatory. The remaining contents of Appendix A are discussed as needed in the remainder of this document. The relationship between the IPN order (and EPNs) and graphic alarms through the use of pointers is discussed further in Section 5.6.1.

Usually, the continuous database is setup with spare IPNs (or EPNs not defined) at the end of each sequence unit. Therefore, there are fewer EPNs in a sequence unit than the maximum allowed by the sequence database. Hopefully as signals are added to the continuous database over time, these "spare" IPNs are sufficient in quantity to add new signals without having to reshuffle the entire IPN order, which is a major undertaking. It would be nice to add plenty of spares to the continuous database to handle all future needs. However, there is a tradeoff between the number of spares available and the additional memory consumed by having more spares. Generally, 5 to 10% is a reasonable amount of spare IPNs to last a long time.

5.3 RPFs

Recipe Prototype Files (RPF) define all flags, integers, analog variables, second timers, minute timers, and string variables that can be modified from the OCM. All RPFs are found in Appendix C. SABL requires defining RPFs, but these variables are only used in the new alarm scheme as an extension to the UPF scratch pad variables table.

5.4 UPFs

Unit Prototype Files (UPF) define all scratchpad variables used by the sequence programs. All UPFs are found in Appendix D. Each UPF has one RPF associated with it. The UPF for PCMO is slightly different from the UPF for PCML. Because of this, the variables used by the programs running in the two PCMs are slightly different.

The following table lists the sequence unit names, numbers, program names, and their associated PCM, RPF, and UPF. Note the similarity between sequence unit names and sequence program names.

SEQUENCE UNIT NAME	SEQUENCE UNIT NUMBER	SEQUENCE PROGRAM NAME	PCM	RPF	UPF
WHOLEO	0	TAG.SEQ	0	ALLO	TOTALO
WHOLEO	0	TIMEINHA.SEQ	0	ALLO	TOTALO
A244	1	244BATCH.SEQ	0	FARM	SHORTO
HVC244	2	244HVAC.SEQ	0	FARM	SHORTO
WASTE	3	WSTXFR.SEQ	0	FARM	SHORTO
A350	4	350BATCH.SEQ	0	FARM	SHORTO
FEED	5	FEEDBATCH.SEQ	0	242A	PROCESS
CA1	6	CA1BATCH.SEQ	0	242A	PROCESS
PB1	7	PB1BATCH.SEQ	0	242A	PROCESS
PB2	8	PB2BATCH.SEQ	0	242A	PROCESS
EA1	9	EA1BATCH.SEQ	0	242A	PROCESS
EC1	10	EC1BATCH.SEQ	0	242A	PROCESS
RWATER	11	RWBATCH.SEQ	0	242A	PROCESS
URW	12	URWBATCH.SEQ	0	242A	PROCESS
VVENT	13	VENTBATCH.SEQ	0	242A	PROCESS
PC	14	PCBATCH.SEQ	0	242A	PROCESS
IX	15	IXBATCH.SEQ	0	242A	PROCESS
SC	16	SCBATCH.SEQ	0	242A	PROCESS
BASINS	17	BASINBAT.SEQ	0	242A	PROCESS
UTIL	18	UTILBATCH.SEQ	0	242A	PROCESS
ROOMS	19	-----	0	242A	PROCESS
AMU	20	AMUBATCH.SEQ	0	242A	PROCESS
LERF	21	LERFBATCH.SEQ	0	242A	PROCESS
EVAP	22	CASSBATCH.SEQ	0	242A	OVREVIEW
VAPOR	23	-----	0	242A	OVREVIEW

SEQUENCE UNIT NAME	SEQUENCE UNIT NUMBER	SEQUENCE PROGRAM NAME	PCM	RPF	UPF
COND	24	-----	0	242A	OVVIEW
BLANKO	25	PGMOA.SEQ	0	EMPTY	BLANKK
WHOLE1	0	WHITE1.SEQ	1	ALL1	TOTAL1
ANFARM	1	ANBATCH.SEQ	1	FARM	TNKFRM
APFARM	2	APBATCH.SEQ	1	FARM	TNKFRM
AWFARM	3	AWBATCH.SEQ	1	FARM	TNKFRM
XYZFRM	4	XYZBATCH.SEQ	1	FARM	TNKFRM
BLNDG	5	242BATCH.SEQ	1	FARM	TNKFRM
BLANK1	6	PGM1A.SEQ	1	EMPTY	BLANKK

5.5 Sequence Database

The SDB utility associates a number of AI, DIN, and DEVs with a unit prototype file. SDB output files for PCMO and PCMI are found in Appendix B. These listings give the sequence unit number, sequence unit name, UPF name, the total quantity of DEVs and EPN of the first DEV, total quantity of AIs and EPN of the first AI, and total quantity of DINs and EPN of the first DIN.

5.6 Pointers, Tables, And How The Programs Work

5.6.1 Pointers

Analog and digital signal values are all accessed through pointers. The first DEV, DIN, and AI in a sequence unit are assigned to pointers. The first integer and flag, also called variables, of a table are also assigned to pointers. By arranging variables into tables and comparing the value of integers and flags in the tables to the value of the EPNs, alarms are found, acknowledged, or cleared. Pointers start with the first EPN in a sequence unit and are stepped by one through all signals, which are arranged by IPN, in the unit. Therefore, the IPN order and use of pointers determines the relationship between alarm messages in graphics and EPNs associated with the graphics and SKID keys.

Alarm programs work by assigning the first DEV, DIN, or AI in a sequence unit to a pointer, checking its alarm status, and then moving on to the next EPN by stepping the pointer. Each EPN has one integer (TYPE_n where n is a number, or pointer PTYPE) assigned to it that defines the type (red, yellow, white, etc.) of alarms it has. Each EPN also has another integer that indicates the status of that alarm (INT_n, or pointer PSTAT). It is the INT_n variables that are declared in a graphic file in association with an alarm message string (or string packet conversion number) for a DIN. The string packets are defined using the SED utility, which generates the D/3 file STRING.BIN and list file STRING.OUT. The string packet listing is found in Appendix G.

The association of the INTn batch variables to the EPNs within each unit are found in the IPN order list ("SDB Int" column of Appendix A). The association of the TYPEn batch variables to the EPNs within each unit follow the same order. For example, by referring to Appendix A one can see that the DIN LDS-PP, in unit A244 (unit 1 of PCMO), is associated with INT17. Likewise, LDS-PP is also associated with TYPE17. The actual TYPEn designations (0 for none, 1 for white, etc.) are inferred from the Tag List (WHC-SD-534-CSCM-001), which defines all alarms associated with a signal.

Programs running in PCMO and PCMI work the same way with a few subtle differences, particularly the use of pointer PSKID. In addition, the names of variables used in PCMO differ somewhat from the names of variables in PCMI programs because different UPFs are involved. These differences are presented below and in Sections 5.6.3, 5.6.4, and 5.6.5.

Despite these differences, the meaning of the values of the program variables are identical. The values of some common pointers are described below. Differences in the variable names whose attributes are assigned (ie: ASSIGN PTYPE = TYPE0) to the pointers are also noted.

Values of pointer PTYPE (TYPEn for PCMO, IALMn for PCMI) are:

- 0 = no alarm is defined
- 1 = white alarm
- 2 = yellow alarm
- 3 = yellow and white alarm combination
- 4 = red alarm
- 6 = red and yellow alarm combination

Values of pointer PSTAT (INTn for all programs) are:

- 0 = no alarm
- 1 = alarm found but not acknowledged, results in flashing message in graphic
- 2 = alarm is acknowledged
- 3 = alarm is inhibited (INHA)
- 4 = signal is off scan or disabled (OSCN, or DIS)
- 5 = signal has a bad input (IBAD)

Values of pointer PREV (FRVn for PCMO, FXn for PCMI) when associated with DINs are:

- 0 = alarm when DIN_VAL is 0
- 1 = alarm when DIN_VAL is 1

Note: For digital inputs, alarms occur if PREV = PEPN:DIN_VAL

Values for pointer PREV (FRVn for PCMO, FXn for PCMI) when associated with AIs are:

- 0 = Hi or HiHi alarm
- 1 = Lo or LoLo alarm

The attribute of variable FPASn (for all programs) is assigned to pointer PPASS. If PPASS equals 1, the SKID light turns off (provided there are no other alarms on the graphic) and graphic alarm message disappears after the alarm is acknowledged.

5.6.2 Procedure Used By All Programs

The same procedure is used by all programs to handle alarms, although subtle variations may exist between programs. The commented code listings of each sequence program are found in Appendix E. The procedure used by the programs for alarm management is summarized below.

1. Does the EPN have an alarm defined? If so, PTYPE is greater than zero. The type (PTYPEn, or IALMn) is a constant defined in the initialization section of a program.
2. If an alarm is defined, is the EPN inhibited? If so, PSTAT is less than 3.
3. If not inhibited, is there an alarm present? Check by comparing PDIN:DIN_VAL and PREV.
4. If an alarm is present, does the system know of the alarm and has it been acknowledged (PSTAT equals 2 if true)? If not, then go to the Alarm Subroutine, ALM.
5. What type of alarm is it (red, yellow, or white)? Check the PTYPE.
6. For PCMI programs, what graphic is it found in? Check the PSKID.
7. If acknowledged, should the alarm message vanish and the SKID lamp turn off? If so, PPASS equals 1. PPASS is a constant defined in the initialization section of a program. Note that if other alarms exist in the graphic, the message for an alarm may vanish but the lamp may remain on.
8. Has the alarm been acknowledged either from graphics or the system? Check by bitwise AND (AND (PDIN:DIN_ALMCT,32)).
9. Summarize all alarms in this unit and flash the skid key accordingly. For example, CURENT = OR (CURENT,NXT).

The primary variations between programs include:

- A few programs handle alarms found in a PCM other than the PCM the program is running in. In these cases, the program must fetch the data from the other PCM. This programming is hard-coded. PCM fetches are performed in RWBATCH, WSTXFR, WHITE1, XYZBATCH, 242BATCH, and CASSBATCH.
- A few programs manage alarms associated with logic devices (see Section 5.6.7).
- Reset CAL block output for totalizers. Programs that do this are RWBATCH, ANBATCH, APBATCH, and AWBATCH.
- A permissive always involves an Input-Output Switch (IOS) block. For loops having a permissive, set the PID output % (the O/C display bar % value) to zero when a permissive is lost (value of the TOGGLE goes to zero) to match the loop output. Programs that do this are FEEDBATCH, PB2BATCH, EA1BATCH, and VENTBATCH.
- Synchronizing of cascades to provide a bumpless transfer of a cascaded loop setpoint. Programs where this is done includes PCBATCH, FEEDBATCH, and PB2BATCH.
- EPN statuses, of DEVs and DINs, for the interlock graphics.

5.6.3 PCM1 DIN Alarms

The pointer PSKID in PCM1 programs, which are mainly for Tank Farms, is used to indicate which graphic has an active alarm, and pointer PSN (variable FSNn) is a flag for the graphic to acknowledge. The reason these pointers are needed is because one Tank Farm sequence program in PCM1 may control 4 or 5 SKID lamps. In PCMO programs, one sequence program controls one lamp. The following is a sample of the pointer table used in the ANBATCH program for 241-AN Tank Farm signals. Pointer names are in parenthesis. Similar pointer tables are used for the other Tank Farm programs; APBATCH, AWBATCH, and XYZBATCH. There is no column for pointer PSN in the table below because 1) the FSNn variables are assigned to a graphic and not to individual EPNs, and 2) only 4 or 5 FSNn variables (FSN0 to FSN3 or FSN4) are used in each program, one for each graphic.

<u>EPN</u> <u>(PDIN)</u>	<u>TYPE</u> <u>(PTYPE)</u>	<u>STATUS</u> <u>(PSTAT)</u>	<u>DIRECTION</u> <u>(PREV)</u>	<u>VANISH</u> <u>(PPASS)</u>	<u>GRAPHIC</u> <u>(PSKID)</u>
YS-AN-101	IALMO	INT0	FX0	FPAS0	SKID0
RSH-AN101	IALM1	INT1	FX1	FPAS1	SKID1
RSH-AN01C	IALM2	INT2	FX2	FPAS2	SKID2
YS-AN-102	IALM3	INT3	FX3	FPAS3	SKID3
RSH-AN102	IALM4	INT4	FX4	FPAS4	SKID4

<u>EPN</u> <u>(PDIN)</u>	<u>TYPE</u> <u>(PTYPE)</u>	<u>STATUS</u> <u>(PSTAT)</u>	<u>DIRECTION</u> <u>(PREV)</u>	<u>VANISH</u> <u>(PPASS)</u>	<u>GRAPHIC</u> <u>(PSKID)</u>
RSH-AN02C	IALM5	INT5	FX5	FPAS5	SKID5
YS-AN-103	IALM6	INT6	FX6	FPAS6	SKID6
RSH-AN103	IALM7	INT7	FX7	FPAS7	SKID7
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

The first EPN and the attributes of the first integer and flag found on the first row of the above table are assigned to their corresponding pointer. For example, these assignments are made in the code as follows:

```

ASSIGN PDIN = 'YS-AN-101'
ASSIGN PTYPE = IALM0
ASSIGN PSTAT = INTO
ASSIGN PREV = FX0
ASSIGN PPASS = FPAS0
ASSIGN PSKID = SKID0

```

Alarms are found by comparing the 'EPN':DIN VAL values to values in the PREV column of the table. The program looks at all DINs, AIs, and sometimes the DEVs in a unit. When a white or yellow alarm occurs, an audible alarm horn (different from the critical alarm horn) is turned on. This is done by setting the flag FALMON in any sequence unit. This flag is scanned by a "C" program running in the PCM. The subsequent actions of this and other "C" programs are discussed in Section 5.10.

Within the sequence program, the SKID key status is the result of performing a logical "OR" on the PSTAT column and converting it to the appropriate SKID key function number where:

- 0 = SKID lamp off
- 1 = SKID lamp steady
- 2 = SKID lamp flash
- 3 = SKID lamp off-synchronized flash

5.6.4 PCMO DIN Alarms

PCMO digital inputs, which are primarily Evaporator signals, are handled the same way as the PCMI DINs, which are predominately Tank Farm signals, but are associated with a different UPF. Because of this, some variable names change. The following pointer table is a sample of PB1 digital signals showing the different variable names. Pointer names are in parenthesis. Similar pointer tables exist for the other PCMO sequence units and DINs.

EPN (PDIN)	TYPE (PTYPE)	STATUS (PSTAT)	DIRECTION (PREV)	VANISH (PPASS)	IPN
YS-PC-105	TYPE0	INT0	FRV0	FPAS0	70
YS-PC105A	TYPE1	INT1	FRV1	FPAS1	71
ZS-CA1-7N	TYPE2	INT2	FRV2	FPAS2	72
YS-PB1-1	TYPE3	INT3	FRV3	FPAS3	73
ZS-CA1-7C	TYPE4	INT4	FRV4	FPAS4	74
LDS-SMPL2	TYPE5	INT5	FRV5	FPAS5	75
ZS-CA1-8N	TYPE6	INT6	FRV6	FPAS6	76
ZS-CA1-8C	TYPE7	INT7	FRV7	FPAS7	77
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

5.6.5 PCMO AI Red and Yellow Alarms

As with PCMO DINs, the PCMO analog inputs are primarily Evaporator signals.

The type of analog alarm is always checked for each EPN prior to executing the analog alarm code. If TYPE_n for an EPN is initialized as zero, then there are no defined alarms for the EPN and the bulk of the analog alarm coding is bypassed.

A new analog alarm exists if the analog alarm count (AI ALMCT) number is less than 16 but greater than zero. Evaporator graphics use PSKID to indicate the previous analog alarm status. A new alarm occurs when PSKID does not equal PAI:AI ALMCT, and the signal is not inhibited, and the alarm is current. In addition, if the PAI:AI ALMCT is less than 3, then the alarm is non-critical and the yellow-white audible horn is activated. Hi and Lo alarms are generally yellow (non-critical). HiHi and LoLo alarms are always red (critical), meaning an audible alarm will be set off on an OCM by the D/3 software provided by GSE. AI ALMCT values are:

- 0 = no alarm
- 1 = Hi alarm
- 2 = Lo alarm
- 3 = HiHi alarm
- 4 = LoLo alarm
- 16 = all alarms are inhibited
- 32 = all alarms are acknowledged

A sample of the pointer table for analog signals in sequence unit PB1 is summarized below. Pointer names are in parenthesis. Usually PSTAT, PREV, and PPASS are not used in the code for analog red and yellow alarms. In almost all PCMO programs PAI, PTYPE, and PSKID are used. Similar pointer tables exist for the other PCMO sequence units and AIs.

EPN (PAI)	TYPE (PTYPE)	STATUS (PSTAT)	DIRECTION (PREV)	VANISH (PPASS)	AI ALMCT (PSKID)
KY-PB1-1	TYPE10	INT10	FRV10	FPAS10	SKID10
FI-CA1-7	TYPE11	INT11	FRV11	FPAS11	SKID11
II-PB1-1	TYPE12	INT12	FRV12	FPAS12	SKID12
VI-PB1-2A	TYPE13	INT13	FRV13	FPAS13	SKID13
VI-PB1-1A	TYPE14	INT14	FRV14	FPAS14	SKID14
VI-PB1-3A	TYPE15	INT15	FRV15	FPAS15	SKID15
FI-CA1-1	TYPE16	INT16	FRV16	FPAS16	SKID16
PI-CA1-9	TYPE17	INT17	FRV17	FPAS17	SKID17
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

There are a few minor differences in the pointer tables and code used in some of the PCMO sequence programs. First, in PCBATCH and IXBATCH the pointer PTYPE is not used. Instead, PALARM (or FALMn) is used for the same purpose as PTYPE, but in different direction. In most programs, if PTYPE is set then the section of code is executed. However in PCBATCH and IXBATCH, if PALARM is not set, which is the case for all EPNs in these units, then the section of code is executed. Although a column for PALARM in the above table is not shown, the variable assignments in PCBATCH are similar (ie: for II-A-C100, the attribute of FALM5 is assigned to PALARM, and so on).

Secondly, sequence programs in PCMO do not all require similar sections of code to handle the different types of alarms; red, yellow, and white. Some units do not have white alarms, so there may not be a section of code in the program for that unit to handle white alarms. Similarly, some units do not have red or yellow alarms, so there may not be a section of code in the program to handle them.

Finally, there are some code inconsistencies between programs. These are relatively minor and it is not the intent of this document to discuss them, but the reader needs to be aware that they exist. Just as there usually are several ways to arrive at the correct solution to a mathematical problem, there are even more possible combinations of code that will result in the same desired outcome. For example, the initialization of variables for analog signals may be done inside the same loop as the digitals, or the analogs may be initialized in a separate loop.

5.6.6 PCMO AI White Alarms

All white alarms for analog signals are in PCMO. Most of these are Evaporator signals, but several are Tank Farm signals. PCM1 contains some white alarms, but these are only digital inputs. The white alarm summary page program runs in PCM1 and is discussed in Section 5.7.

WHC-SD-534-CSWL-001 REV 3

Coding for white analog alarms use variables not used in the coding for analog red and yellow alarms. The type (TYPE_n) of alarm is a constant defined during program initialization. TYPE_n equals one (1) for white alarms. As with other programs, the alarm status (INT_n) integer variables are related to an EPN and graphic alarm message. An EPN is assigned to pointer PAI before running analog white alarm subroutine, ANLG. The ANLG subroutine consists of two primary loops; one for handling the Hi and HiHi alarms, and second nearly identical loop for the Lo and LoLo alarms. The proper loop for each EPN is defined by the reverse flag variables (FRV_n), which are constants established in the initialization section of a program. Values of the FRV_n variables follow the convention:

0 = HI or HIHI alarm
1 = LO or LOLO alarm

Some EPNs have more than one white alarm. In such cases, EPNs will be associated with multiple FRV variables. The white alarm setpoints are controlled by the Tag List (WHC-SD-534-SWD-001). The total number of white alarms for each EPN can be determined from the Tag List, from sequence code listings, or from the IPN order (Appendix A). For example, Appendix A shows two white alarms for PI-EA1-13 in unit EA1. If an alarm must disappear after it is acknowledged, then the constant FPAS_n is set equal to one (1) during initialization. To determine if an alarm exists, analog measured value AI MEAS is compared to floating point value AIn. For example, the measured value of PI-EA1-13 is compared to AI0, and so on. The white alarm setpoints, AIn, are defined in the initialization section of a program. The pointer table for EA1 unit white alarms is listed below. Similar pointer tables are used for other units.

<u>EPN</u> <u>(PAI)</u>	<u>TYPE</u> <u>(PTYPE)</u>	<u>STATUS</u> <u>(PSTAT)</u>	<u>DIRECTION</u> <u>(PREV)</u>	<u>VANISH</u> <u>(PPASS)</u>	<u>ALARM</u> <u>(PALARM)</u>
PI-EA1-13	TYPE26	INT26	FRV26	FPAS26	AI0
PI-EA1-13	TYPE27	INT27	FRV27	FPAS27	AI1
FIC-EA1-1	TYPE28	INT28	FRV28	FPAS28	AI2
PI-EA1-1	TYPE29	INT29	FRV29	FPAS29	AI3
PI-EA1-1	TYPE30	INT30	FRV30	FPAS30	AI4
TI-EA1-5	TYPE31	INT31	FRV31	FPAS31	AI5

Values of these variables as found in EA1BATCH.SEQ are:

<u>EPN</u> <u>(PAI)</u>	<u>TYPE</u> <u>(PTYPE)</u>	<u>STATUS</u> <u>(PSTAT)</u>	<u>DIRECTION</u> <u>(PREV)</u>	<u>VANISH</u> <u>(PPASS)</u>	<u>ALARM</u> <u>(PALARM)</u>
PI-EA1-13	1	INT26	1 (LoLo)	0	75.0
PI-EA1-13	1	INT27	1 (Lo)	0	80.0
FIC-EA1-1	1	INT28	1 (Lo)	0	3000.0
PI-EA1-1	1	INT29	1 (Lo)	0	17.7
PI-EA1-1	1	INT30	0 (Hi)	0	29.7
TI-EA1-5	1	INT31	0 (Hi)	0	190.0

When one EPN has more than one white alarm, the more extreme alarm has priority. This means that a HiHi has priority over a Hi alarm, and a LoLo has priority over a Lo. When the more extreme alarm resets, the less extreme alarm becomes active again. A Lo alarm resets when the measured value is greater than the Lo alarm limit plus AI scan block deadband, and a Hi alarm resets when the measured value is less than the Hi alarm limit minus the deadband.

5.6.7 Device Alarms

A few programs handle alarms associated with logic devices. The code for device alarms is very similar to the code for the digital alarms, except that a) the code for device alarms is usually hard-coded, whereas the code for most DINs use pointers and tables, and b) device alarm programming uses DEV_STAT and DEV_ALMCT, whereas the code for DINs use DIN_VAL and DIN_ALMCT.

5.7 White Alarm Summary Program

The white alarm summary page program running in PCMI and found in unit WHOLE1 is called WHITE1. This program runs independently of all other batch programs. The alarm values are hard-coded into this lengthy SABL program. Signals reset as described in the last paragraph of Section 5.6.6. As new alarms occur, they are displayed at the top of the list, much like P1 and P2 alarms, and cause SKID key 60 to flash. Pressing SKID key 60 causes it to return to a dark-board condition. White alarms are eliminated from the list when inhibited or when they clear.

The flags and integers listed below are used differently than other programs. Pointer names are in parenthesis. The maximum of the range, 191, is the total number of each variable in the UPF. Currently, 177 of each variable are being used, with many of these being spares or old, abandoned white alarms. The letter "n" in the variable name indicates a number between 1 and 177.

<u>VARIABLE (POINTER)</u>	<u>TYPE</u>	<u>RANGE</u>	<u>PURPOSE</u>
FXn (PSTAT)	FLAG	0 - 191	For DINs, is the real-time digital input value. For AIs, is set to 1 if AI is in an alarm state.
FRVn (PREV)	FLAG	0 - 191	FRV is the predetermined value of a digital input when in the alarm state. An alarm exists when PREV equals PSTAT. Set equal to 1 for all analog alarms.

FDSn (PTYPE)	FLAG	0 - 191	If equal to 1, then alarm is inhibited.
FALn (PALARM)	FLAG	0 - 191	Set to 1 when an alarm is found to prevent repeat. Is reset to 0 when alarm clears.
INTn (PLINE)	INTEGER	0 - 191	Is the line in the white alarm list display where the alarm message resides. If 0, then no alarm.

The white alarm program first initializes all alarm conditions by setting flags FRVn and FDSn. The inhibit flag, FDSn, for each signal is initialized as a 1. For signals that have white alarms, the program then reads the DIN ALMCT, AI ALMCT, or DEV ALMCT bit values (at the start of the program repeat loop, AGAIN:) to determine if they are really inhibited or not and sets FDSn accordingly. Next, it determines the real-time signal status and sets FXn to the value of each DIN or analog alarm. If a signal is not inhibited (if not, PTYPE = FDSn = 0), then the program checks if to see if the signal was previously alarmed (if not, PALARM = FALn = 0). If not inhibited and not previously alarmed, then the program checks to see if a new alarm exists. A new alarm exists when FXn equals FRVn (when PREV = PSTAT). If a new alarm is found, then the program does the following:

- Set PALARM (FALn) equal to 1
- Set PLINE (INTn) equal to ILOOP
- Flash the SKID lamp
- Index, or bump, all messages in the white alarm summary graphics by 1.

If a signal is inhibited (PTYPE = FDSn = 1) and it was previously alarmed (PALARM = FALn = 1) then the program clears the alarm messages for that signal from the alarm summary graphic. To accomplish this the program does the following:

- Set PALARM (FALn) equal to 0
- Step to the alarm message line displayed in the graphic. That is, step down ILOOP lines from first line (STEP PLINE BY ILOOP).
- Step all message lines backwards to erase the alarm message and set PLINE (INTn) equal to 0.

A signal that once had a white alarm which was deleted is handled in a unique fashion. This is because it was felt that some white alarms might be reinstated later. For all signals, FDSn (PTYPE) is a constant set equal to 1 (PTYPE = 1 for inhibited) in the initialization section of the code. When the inhibit flag, PTYPE, is set the program bypasses the alarm status checking code contained in the program loop labeled "LOP:", and alarm messages of inhibited signals are kept out of the white alarm summary graphics. By taking

advantage of this, white alarms are essentially deleted by commenting specific lines of code pertaining to the individual EPNs. Reinstating deleted white alarms is a simple programming matter of removing semi-colons in commented lines. An example of lines of code that have been commented to delete AW Farm pump device white alarms are listed below (see Appendix E for the actual code). Similar lines of code would be commented for deleted analog or digital white alarms.

- In the initialization section, definition of FRV70 through FRV75
- In the AGAIN program loop, current settings of FDS70 through FDS75 which reads the DEV ALMCT bit values for inhibit checking
- In the AGAIN program loop, current device statuses FX70 through FX75
- In the alarm subroutine, "ALM:", graphic message strings from labels "S70:" through "S75:"

Appendix A indicates the EPNs which have white alarms associated with them. This is indicated in the "White Numb" column. For example, the digital signal PSH-FP in unit A244 is a white alarm. The number "084" refers to the variable number or label (ie: FX84, FRV84, FDS84, FAL84, INT84, and "S84:") in the white alarm program that deals with this signal.

5.8 Unit Relative Feature of Programs

All MCS SABL programs using the same UPF are unit relative, which means several things. First, the code from one program to the next is similar if not identical, which is wonderful from a programmers standpoint. However, the 242-A process does not lend itself to unit relativity because there is no similarity between process units as in a production facility. As a result, there is a down-side to this unit relativity. For example, ANBATCH can run in unit APFARM, or CA1BATCH can run in unit AMU. This is highly undesirable because if this were to happen false alarms would be presented to the operator. To prevent this, a GETUNIT command early in each program determines which sequence unit a program is running in. If a program is running in the wrong unit then the program aborts, a SKID key (key 48 labeled "Program Status") flashes, and a message is printed at the system alarm typer.

5.9 Program Statuses, Diagnostics, and Corrective Action

Two SABL programs, one in each PCM, monitor the status of all sequence programs. PGM0A runs in the PCM0 processors, and PGM1A runs in the PCM1 processors. Program statuses are displayed on graphic 43. Normal or healthy statuses (RUNNING, WAIT, or CRITICAL) are shown in a steady green color. Abnormal statuses (NOT-RUNNING, TRAPPED, or FAULT on AUDALARMO) are displayed in a flashing red color on the graphic and cause the Program Status SKID key (key 48) to flash. The abnormal statuses, NOT-RUNNING and TRAPPED, require corrective action (aborting, unloading, reloading, and running the indicated

program). Command files in directory D3DATA:[100001] of the CDCM are used to correct these problems.

In the Program Status graphic, error lines and error codes are shown in white for diagnostics. Error number 187 (COULD NOT DO REMOTE ACCESS) is the most common error code shown, if any. A few SABL programs must get data from another PCM (ie: PCMOA program RWBATCH must get the status of DIN TSL242A81 from PCM1A). If the requesting PCM cannot communicate with the other PCM for any reason, then error 187 will occur. Error 187 will occur anytime one PCM is offline or in load-request. There are a host of other reasons why communications may be lost. Most are temporary. Temporary loss of communications usually indicates some hardware problem exists, but could be the result of a data highway that is too busy.

Another occasional problem occurs when the yellow-white alarm horn becomes stuck and cannot be shut off for some reasonable duration. To diagnose this, access the Program Diagnostic display, graphic 133, and observe the flags FALMON, ACK, FLGACK, and FSL0. These flags should normally be green or temporarily red.

5.10 "C" Programs Related To Alarm Scheme

There are six alarm-related "C" programs that enhance the alarm management scheme. These all work together to minimize operator time and effort to find and acknowledge alarms. The standard GSE D/3 package requires that each alarm in a graphic be selected and acknowledged individually. The six "C" programs get around this feature so that pressing the operator console "ACK" key once acknowledges all alarms in the graphic. The only requirement for this to work is that the alarm acknowledge device in a graphic must be selected. The alarm acknowledge device is automatically selected when a graphic is called up. However, the operator may have to re-select this device (using Quick Select key 0) if another EPN was temporarily selected. The alarms in a graphic are also acknowledged throughout the entire D/3 system.

Four of the "C" programs run in the PCMs. One program runs in each PCM for redundancy. The other two programs run on the CDCM. One of these two tasks watches flags in the PCM sequence databases, and the other acknowledges all alarms in a graphic. The programs are named as follows:

<u>Micro Name</u>	<u>Program Name</u>
PCMOA	DOGOA
PCMOB	DOGOB
PCM1A	DOG1A
PCM1B	DOG1B
CDCM	WATCH (installed as task DOG)

Appendix F contains the code listings of these programs.

The relationships between these programs are illustrated in Figure 1. When a white or yellow alarm occurs, a sequence program sets the flag FALMON to a value of one. This flag is scanned by a "C" program running in the PCM called DOGOA, DOGOB, DOG1A, or DOG1B (hereafter referred to as DOGxx) depending on the PCM. If a DOGxx program finds FALMON set, then DOGxx commands the audible alarm device (AUDALARMO) on to turn on the horn, and resets the FALMON flag. Also, DOGOA and DOGOB look at flag ACK (in PCMO units), and DOG1A and DOG1B look at the FLGACK flag (in PCM1 units). If ACK is set to a value of one, then either DOGOA or DOGOB (depending on the PCM selected) sets flag FSLO to a value of one in unit BLANK0. If FLGACK is set to a value of one, then either DOG1A or DOG1B (depending on the PCM selected) sets flag FSLO to a value of one in unit BLANK1. The WATCH program scans the FSLO flag in each of these two BLANKx units. If WATCH finds that either flag is set, then WATCH resets the flag and runs function ACKFLG. The function ACKFLG acknowledges all flashing alarms in the displayed graphic from which the operator console "Alarm ACK" key was pressed. Conversely, if the "Alarm ACK" key was pressed from a P1 or P2 display, then those alarms are acknowledged in the graphics where they are found.

FIGURE 1. Program Relationships

5.11 Other Alarm-Related "C" Programs

There are four other programs that deal with inhibited alarms. The code listings of these programs are found in Appendix F. One of these is INHA.C, which is installed on a DCM as task INHALM. Task INHALM is run from an annunciator SKID panel by pressing key number 53, which is labeled "INHA". This task provides a list of inhibited alarms for graphics 80 through 84. It works in conjunction with SABL program TIMEINHA, which provides the date and time that the program was run. The date and time are displayed on each of the graphics numbered 80 through 84. TIMEINHA is also activated by pressing SKID key 53.

The other three programs are user key functions (UKFs), or General Functions (GFs). General Functions 43 and 44 are activated after the Evaporator is shut down to inhibit select alarms. UKF43 is run only on the CDCM and inhibits the alarms and clears the CDCM alarm summary pages. UKF44 is then run on DCMO to clear its alarm summary pages. General Function 14 (provided by GSE) is then run on the CDCM to recreate the alarms in the alarm summary pages.

Conversely, UKF34 is activated prior to starting the Evaporator up and cancels the inhibited status of the same select EPNs. After UKF34 is run, no other tasks or user key functions need to be run.

5.12 SKID Key Configuration

The annunciator key configuration for each OCM is built from command files in directory D3DATA:[100010] of the CDCM. These files are named CDCMSKO.CMD, CDCMSK1.CMD, DCMOSKO.CMD, DCMOSK1.CMD, and DCMOSK2.CMD. These five files are identical. Appendix I contains a listing of CDCMSKO.CMD.

6.0 REBUILDING DATABASES AND PROGRAMS

A word about rebuilding databases is appropriate at this point. Whenever the continuous database is rebuilt, a new serial number is automatically assigned to that database. This serial number is the fine thread that is associated with the sequence database and graphics. Its purpose is to warn when something has been rebuilt, but something else has not. When such discrepancies occur a large flashing red message, such as "... SERIAL MISMATCH", will appear in a graphic.

When the continuous database is rebuilt, usually most everything else must be rebuilt. The nice thing about the RPFs and UPFs is that unless they have been modified, they do not need to be rebuilt. However, the sequence database, sequence programs, and graphics must be rebuilt. If only one continuous database is rebuilt, it is safest to also rebuild the sequence database, programs, and graphics for the continuous database that was not rebuilt. This will take quite a bit more time, but in the long run it could save some grief. The reason for taking the extra time is that the same cross-reference files (D3DATA:[300001]SDBMASTER.XRF, SDBSHORT.XRF) are written to when the SDB utility is run for PCMO or PCML. It is best to start off clean by renaming the "XRF" files to "OLD" files, and let SDB write new files. A procedure for rebuilding the entire applications software is found in Appendix H.

7.0 APPENDICES

APPENDIX A – IPN ORDER

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
0 - W H O L E O				217-DEV; 379-AIN; 348-DIN			
244ALARM	DEV	0	-	-	-	-	0
YS244AIL	DI	0	-	-	-	-	0
II-P244A1	AI	0	-	-	-	-	0
1 - A 2 4 4				15-DEV; 30-AIN; 50-DIN			
244ALARM	DEV	0	-	-	340	244A	0
P-244-A1	DEV	1	-	-	340	244A	0
SPARE	DEV	2	-	-	340	244A	0
P-244-A3	DEV	3	-	-	340	244A	0
SPARE	DEV	4	-	-	-	-	0
SPARE	DEV	5	-	-	-	-	0
SPARE	DEV	6	-	-	-	-	0
SPARE	DEV	7	-	-	-	-	0
SPARE	DEV	8	-	-	-	-	0
SPARE	DEV	9	-	-	-	-	0
YS244A1IL	DI	0	0	-	340	244A	0
YS-P244A1	DI	1	1	-	340	244A	0
SPARE	DI	2	2	-	-	-	0
SPARE	DI	3	3	-	-	-	0
YS-P244A3	DI	4	4	-	340	244A	0
SPARE	DI	5	5	-	-	-	0
ZS-MOV-2F	DI	6	6	-	340	244A	0
ZS-MOV2DR	DI	7	7	-	340	244A	0
SPARE	DI	8	8	-	-	-	0
SPARE	DI	9	9	-	-	-	0
SPARE	DI	10	10	-	-	-	0
SPARE	DI	11	11	-	-	-	0
SPARE	DI	12	12	-	-	-	0
SPARE	DI	13	13	-	-	-	0
SPARE	DI	14	14	-	-	-	0
PSH-SN215	DI	15	15	-	-	-	0
PSH-SN216	DI	16	16	-	-	-	0
LDS-PP	DI	17	17	-	340	244A	0
SPARE	DI	18	-	-	-	-	0
PSH-FP	DI	19	-	-	-	-	0
II-P244A1	AI	0	20	-	340	244A	0
SPARE	AI	1	21	-	-	-	0
II-P244A3	AI	2	22	-	340	244A	0
DI-244-SP	AI	3	23	-	340	244A	0
WFI-244SP	AI	4	24	-	340	244A	0
SPARE	AI	5	25	-	-	-	0
LI-244TK	AI	6	26	-	340	244A	0
DI-244-TK	AI	7	27	-	340	244A	0
WFI-244TK	AI	8	28	-	340	244A	0
TI-244-TK	AI	9	29	31	340	244A	0
PI-244-TK	AI	10	30	-	340	244A	0
19 MORE	AI						

THESE AIS ARE IN ADDITIONAL UNITS

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
2 - H V C 2 4 4					1-DEV; 5-AIN; 5-DIN		
HVAC244	DEV	10	-	-	346	244AHVAC	0
PDSL-EF-2	DI	20	0	-	341	244AHVAC	0
PDSH-EF-1	DI	21	1	-	341	244AHVAC	0
RSH-A	DI	22	2	-	341	244AHVAC	0
RSH-EXH-1	DI	23	3	-	341	244AHVAC	0
QXS-EXH-1	DI	24	4	-	341	244AHVAC	0
RI-244ANN	AI	11	5	-	341	244AHVAC	0
FI-244EXH	AI	12	6	8,9	341	244AHVAC	0
RI-244EXH	AI	13	7	-	341	244AHVAC	0
II-350-1	AI	14	-	-	-	-	0
LI-350TK	AI	15	-	-	-	-	0
3 - W A S T E					1-DEV; 0-AIN; 18-DIN		
WSTXFR	DEV	11	-	-	349	WSTXFR1	0
LDS-AR151	DI	25	0	-	344	WSTXFR	0
LDS-ER151	DI	26	1	-	344	WSTXFR	0
PSHSNV229	DI	27	2	-	344	WSTXFR	0
LDS-ER152	DI	28	3	-	344	WSTXFR	0
SPARE	DI	29	4	-	-	-	0
SPARE	DI	30	5	-	-	-	0
SPARE	DI	31	6	-	-	-	0
SPARE	DI	32	7	-	-	-	0
LDS-ER153	DI	33	8	-	344	WSTXFR	0
PSHPSN232	DI	34	9	-	344	WSTXFR	0
PSHPSN234	DI	35	10	-	344	WSTXFR	0
SPARE	DI	36	11	-	-	-	0
SPARE	DI	37	12	-	-	-	0
SPARE	DI	38	13	-	-	-	0
SPARE	DI	39	14	-	-	-	0
LDS-AX155	DI	40	15	-	344	WSTXFR	0
LDS-DR713	DI	41	16	-	344	WSTXFR	0
LDS-DR717	DI	42	17	-	344	WSTXFR	0
4 - A 3 5 0					3-DEV; 17-AIN; 5-DIN		
350ALARM	DEV	12	-	-	343	350A	0
P-350-1	DEV	13	-	-	343	350A	0
SPARE	DEV	14	-	-	-	-	0
LSH-350-2	DI	44	1	-	343	350A	0
LDS-350-2	DI	45	2	-	343	350A	0
LDS-PW481	DI	46	3	-	343	350A	0
SPARE	DI	47	-	-	-	-	0
SPARE	DI	48	-	-	-	-	0
II-350-1	AI	14	4	-	343	350A	0
LI-350TK	AI	15	5	-	343	350A	0
DI-350-TK	AI	16	6	-	343	350A	0
WFI-350TK	AI	17	7	-	343	350A	0
FQI-350-2	AI	18	8	-	343	350A	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
FQI-350-1	AI	19	9	-	343	350A	0
FI-350-1	AI	20	10	-	343	350A	0
SPARE	AI	21	-	-	-	-	0
SPARE	AI	22	-	-	-	-	0
SPARE	AI	23	-	-	-	-	0
SPARE	AI	23	-	-	-	-	0
SPARE	AI	24	-	-	-	-	0
SPARE	AI	25	-	-	-	-	0
SPARE	AI	26	-	-	-	-	0
SPARE	AI	27	-	-	-	-	0
SPARE	AI	28	-	-	-	-	0
SPARE	AI	29	-	-	-	-	0
5 - F E E D					10-DEV; 15-AIN; 10-DIN		
PAW102ALM	DEV	15	-	-	301	AWFEED	0
P-AW-102	DEV	16	-	-	301	AWFEED	0
HV-CA1-1	DEV	17	-	-	301	AWFEED	0
VCA1-INLK	DEV	18	-	-	301	AWFEED	0
SAPRE	DEV	19	-	-	-	-	0
SAPRE	DEV	20	-	-	-	-	0
SAPRE	DEV	21	-	-	-	-	0
SAPRE	DEV	22	-	-	-	-	0
SAPRE	DEV	23	-	-	-	-	0
SAPRE	DEV	24	-	-	-	-	0
YS-AW102	DI	50	0	-	301	AWFEED	0
YS-CA1-1	DI	51	1	-	301	AWFEED	0
LDS-SMPL1	DI	52	2	-	301	AWFEED	0
PSH-102-3	DI	53	3	-	301	AWFEED	0
YSAW102IL	DI	54	4	-	301	AWFEED	0
ZS-CA1-1C	DI	55	5	-	301	AWFEED	0
ZS-CA1-1N	DI	56	6	-	301	AWFEED	0
SPARE	DI	57	-	-	-	-	0
SPARE	DI	58	-	-	-	-	0
SPARE	DI	59	-	-	-	-	0
II-AW-102	AI	30	7	7	301	AWFEED	0
DI-AW-102	AI	31	8	-	301	AWFEED	0
FIC-CA1-1	AI	32	9	-	301	AWFEED	0
FQI-CA1-1	AI	33	10	-	301	AWFEED	0
KY-CA1-1	AI	34	11	-	301	AWFEED	0
CI-WVR-2	AI	35	12	-	301	AWFEED	0
CI-WVR-3	AI	36	13	-	301	AWFEED	0
CI-WVR-1	AI	37	14	-	301	AWFEED	0
PI-102-AW	AI	38	15	-	301	AWFEED	0
KY-102-3	AI	39	16	-	301	AWFEED	0
SPARE	AI	40	-	-	-	-	0
SPARE	AI	41	-	-	-	-	0
SPARE	AI	42	-	-	-	-	0
SPARE	AI	43	-	-	-	-	0
SPARE	AI	44	-	-	-	-	0
6 - C A 1					10-DEV; 35-AIN; 10-DIN		
CA1-ALARM	DEV	25	-	-	010	CA1	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
SELECT-WF	DEV	26	-	-	010	CA1	0
SELECT-DI	DEV	27	-	-	010	CA1	0
HV-PDSPRY	DEV	28	-	-	010	CA1	0
SPARE	DEV	29	-	-	-	-	0
SPARE	DEV	30	-	-	-	-	0
SPARE	DEV	31	-	-	-	-	0
SPARE	DEV	32	-	-	-	-	0
SPARE	DEV	33	-	-	-	-	0
SPARE	DEV	34	-	-	-	-	0
PSH-CA111	DI	60	0	-	010	CA1	0
WFSH-CA11	DI	61	1	-	010	CA1	0
WFSH-CA12	DI	62	2	-	010	CA1	0
PDSH-CA1	DI	63	3	-	010	CA1	0
SPARE	DI	64	-	-	-	-	0
SPARE	DI	65	-	-	-	-	0
SPARE	DI	66	-	-	-	-	0
SPARE	DI	67	-	-	-	-	0
SPARE	DI	68	-	-	-	-	0
SPARE	DI	69	-	-	-	-	0
PI-CA1-11	AI	45	4	-	010	CA1	0
PIC-CA1-7	AI	46	5	-	010	CA1	0
TI-CA1-6S	AI	47	6	-	010	CA1	0
TI-CA1-6	AI	48	7	-	010	CA1	0
TDI-BPR-1	AI	49	8	-	010	CA1	0
WFI-CA1-1	AI	50	9	-	010	CA1	0
LI-CA1-1	AI	51	10	-	010	CA1	0
LI-CA1-1G	AI	52	11	-	010	CA1	0
LIC-CA1-1	AI	53	12	-	010	CA1	0
WFI-CA1-2	AI	54	13	-	010	CA1	0
LI-CA1-2	AI	55	14	-	010	CA1	0
LI-CA1-2G	AI	56	15	-	010	CA1	0
LIC-CA1-2	AI	57	16	-	010	CA1	0
DIC-CA1-1	AI	58	17	-	010	CA1	0
DI-CA1-1	AI	59	18	-	010	CA1	0
DI-CA1-2	AI	60	19	-	010	CA1	0
DDI-CA1-1	AI	61	20	35,36	010	CA1	0
WFI-CA1-3	AI	62	21	-	010	CA1	0
LI-CA1-3	AI	63	22	-	010	CA1	0
LI-CA1-3G	AI	64	23	-	010	CA1	0
PDI-FH1-1	AI	65	24	37	010	CA1	0
PDI-FH2-1	AI	66	25	38	010	CA1	0
PDI-FH3-1	AI	67	26	39	010	CA1	0
FTC-CA1-6	AI	68	27	-	010	CA1	0
FQI-CA1-6	AI	69	28	-	010	CA1	0
PDI-CA1-2	AI	70	29	-	010	CA1	0
HIC-CA114	AO	71	30	-	010	CA1	0
HIC-CA115	AO	72	31	-	010	CA1	0
PDI-CA1-1	AI	73	32	-	010	CA1	0
TI-CA1-10	AI	74	33	-	010	CA1	0
SPARE	AI	75	34	-	-	-	0
SPARE	AI	76	-	-	-	-	0
SPARE	AI	77	-	-	-	-	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
SPARE	AI	78	-	-	-	-	0
SPARE	AI	79	-	-	-	-	0
7 - P B 1					15-DEV; 20-AIN; 15-DIN		
PB1-ALARM	DEV	35	-	-	012	PB1	0
PB-1	DEV	36	-	-	012	PB1	0
PB1WFINLK	DEV	37	-	-	012	PB1	0
PB1-BYPAS	DEV	38	-	-	012	PB1	0
HV-CA1-7	DEV	39	-	-	012	PB1	0
HV-CA1-9	DEV	40	-	-	012	PB1	0
BOT-DUMP	DEV	41	-	-	012	PB1	0
HV-SUMP-1	DEV	42	-	-	012	PB1	0
JGV-SUMP	DEV	43	30	-	012	PB1	0
P-C-105A	DEV	44	-	-	012	PB1	0
P-C-105	DEV	45	-	-	012	PB1	0
DUMP-INLK	DEV	46	-	-	012	PB1	0
HV-CA1-10	DEV	47	-	-	-	-	0
SPARE	DEV	48	-	-	-	-	0
SPARE	DEV	49	-	-	-	-	0
YS-PC-105	DI	70	0	-	012	PB1	0
YS-PC105A	DI	71	1	-	012	PB1	0
ZS-CA1-7N	DI	72	2	-	012	PB1	0
YS-PB1-1	DI	73	3	-	012	PB1	0
ZS-CA1-7C	DI	74	4	-	012	PB1	0
LDS-SMPL2	DI	75	5	-	012	PB1	0
ZS-CA1-8N	DI	76	6	-	012	PB1	0
ZS-CA1-8C	DI	77	7	-	012	PB1	0
ZS-CA1-9C	DI	78	8	-	012	PB1	0
ZS-CA1-9N	DI	79	9	-	012	PB1	0
ZS-SUMP1C	DI	80	10	-	012	PB1	0
ZS-SUMP1N	DI	81	11	-	012	PB1	0
SPARE	DI	82	-	-	-	-	0
SPARE	DI	83	-	-	-	-	0
SPARE	DI	84	-	-	-	-	0
KY-PB1-1	AI	80	0	-	012	PB1	0
FI-CA1-7	AI	81	1	-	012	PB1	0
II-PB1-1	AI	82	2	-	012	PB1	0
VI-PB1-2A	AI	83	3	-	012	PB1	0
VI-PB1-1A	AI	84	4	-	012	PB1	0
VI-PB1-3A	AI	85	5	-	012	PB1	0
FI-CA1-1	AI	86	6	-	012	PB1	0
PI-CA1-9	AI	87	7	-	012	PB1	0
FQI-PB1-W	AI	88	8	-	012	PB1	0
FI-CA1-3	AI	89	9	-	012	PB1	0
TI-CA1-7	AI	90	10	-	012	PB1	0
WFI-SUMP1	AI	91	11	-	012	PB1	0
DI-SUMP-1	AI	92	12	-	012	PB1	0
LI-SUMP1	AI	93	13	-	012	PB1	0
PI-SUMP-1	AI	94	14	-	012	PB1	0
PDI-CA1-3	AI	95	15	-	012	PB1	0
ZI-CA1-10	AI	96	16	-	012	PB1	0
PI-CA1-20	AI	97	17	31	012	PB1	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
SPARE	AI	98	-	-	-	-	0
SPARE	AI	99	-	-	-	-	0
8 - P B 2					15-DEV; 20-AIN; 20-DIN		
PB2-ALARM	DEV	50	-	-	015	PB2	0
PB-2	DEV	51	-	-	015	PB2	0
HV-CA1-2	DEV	52	-	-	015	PB2	0
HV-CA1-2S	DEV	53	-	-	015	PB2	0
HV-CA1-5	DEV	54	-	-	015	PB2	0
HV-CA1-6	DEV	55	-	-	015	PB2	0
JGV-STMFL	DEV	56	-	-	015	PB2	0
CONTROL	DEV	57	-	-	015	PB2	0
PB2SLFILK	DEV	58	-	-	015	PB2	0
PB2WFINLK	DEV	59	-	-	015	PB2	0
PB2GLOBAL	DEV	60	-	-	-	-	0
FQA-RW-1	DEV	61	24	-	015	PB2	0
SPARE	DEV	62	-	-	-	-	0
SPARE	DEV	63	-	-	-	-	0
SPARE	DEV	64	-	-	-	-	0
YS-PB2-1	DI	85	0	-	015	PB2	0
YS-PB2-1L	DI	86	1	-	015	PB2	0
TSH-PB2-1	DI	87	2	-	015	PB2	0
PSH-CA1-3	DI	88	3	-	015	PB2	0
PSH-PB2-1	DI	89	4	-	015	PB2	0
SXS-PB2-1	DI	90	5	-	015	PB2	0
YS-VFD-1	DI	91	6	-	015	PB2	0
SSL-PB2-1	DI	92	7	-	015	PB2	0
SPARE	DI	93	-	-	015	PB2	0
ZS-CA1-21	DI	94	-	-	015	PB2	0
ZS-CA1-22	DI	95	-	-	015	PB2	0
ZS-CA12A1	DI	96	-	-	015	PB2	0
ZS-CA12A2	DI	97	-	-	015	PB2	0
ZS-CA1-6N	DI	98	-	-	015	PB2	0
ZS-CA1-6C	DI	99	-	-	015	PB2	0
ZS-CA1-54	DI	100	-	-	015	PB2	0
ZS-CA1-55	DI	101	-	-	015	PB2	0
SPARE	DI	102	-	-	-	-	0
SPARE	DI	103	-	-	-	-	0
SPARE	DI	104	-	-	-	-	0
HI-WVR-1	AI	100	8	-	015	PB2	0
II-PB2-1	AI	101	9	-	015	PB2	0
SIC-PB2-1	AI	102	10	23	015	PB2	0
VI-PB2-1A	AI	103	11	-	015	PB2	0
PI-CA1-8	AI	104	12	-	015	PB2	0
FQI-PB2-W	AI	105	13	-	015	PB2	0
FI-CA1-2	AI	106	14	-	015	PB2	0
PI-CA1-10	AI	107	15	-	015	PB2	0
KY-CA1-2	AI	108	16	-	015	PB2	0
KY-CA1-2F	AI	109	17	-	015	PB2	0
FIC-CA1-4	AI	110	18	-	015	PB2	0
FQI-CA1-4	AI	111	19	-	015	PB2	0
PI-CA1-5	AI	112	20	-	015	PB2	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
FQI-RW-1	AI	113	21	-	015	PB2	0
FQI-RW-1B	AI	114	22	-	015	PB2	0
SPARE	AI	115	-	-	-	-	0
SPARE	AI	116	-	-	-	-	0
SPARE	AI	117	-	-	-	-	0
FI-RW-1/2	AI	118	-	-	-	-	0
YS-BFP1/2	AI	119	-	-	-	-	0
9 - E A 1					15-DEV; 30-AIN; 5-DIN		
EA1-ALARM	DEV	65	-	-	013	EA1	0
FV-EA1-1	DEV	66	-	-	013	EA1	0
EA1-INLK1	DEV	67	-	-	013	EA1	0
EA1-INLK2	DEV	68	-	-	013	EA1	0
TV-DSH1/2	DEV	69	-	-	013	EA1	0
SELECT-TI	DEV	70	-	-	013	EA1	0
SPARE	DEV	71	-	-	-	-	0
SPARE	DEV	72	-	-	-	-	0
SPARE	DEV	73	-	-	-	-	0
SPARE	DEV	74	-	-	-	-	0
SPARE	DEV	75	-	-	-	-	0
SPARE	DEV	76	-	-	-	-	0
SPARE	DEV	77	-	-	-	-	0
SPARE	DEV	78	-	-	-	-	0
SPARE	DEV	79	-	-	-	-	0
PSL-EA1-1	DI	105	0	-	013	EA1	0
TSH-DSH-3	DI	106	1	-	013	EA1	0
ZS-EA1-1C	DI	107	2	-	013	EA1	0
SPARE	DI	108	-	-	-	-	0
SPARE	DI	109	-	-	-	-	0
TI-EA1-7	AI	120	3	-	013	EA1	0
TI-EA1-7S	AI	121	4	-	013	EA1	0
TI-EA1-1	AI	122	5	-	013	EA1	0
TI-EA1-1S	AI	123	6	-	013	EA1	0
TDI-EA1-1	AI	124	7	-	013	EA1	0
PI-EA1-12	AI	125	8	-	013	EA1	0
PI-EA1-2	AI	126	9	-	013	EA1	0
PDI-EA1-1	AI	127	10	-	013	EA1	0
PI-STM-1	AI	128	11	-	013	EA1	0
PI-EA1-13	AI	129	12	24,25	013	EA1	0
PI-EA1-14	AI	130	13	-	013	EA1	0
FIC-EA1-1	AI	131	14	26	013	EA1	0
ZI-EA1-1	AI	132	15	-	013	EA1	0
TI-DSH-2	AI	133	16	-	013	EA1	0
PI-EA1-1	AI	134	17	27,28	013	EA1	0
PI-EA1-11	AI	135	18	-	013	EA1	0
PDI-EA1-2	AI	136	19	-	013	EA1	0
TI-DSH-3	AI	137	20	-	013	EA1	0
TI-EA1-5	AI	138	21	29	013	EA1	0
CI-EA1-U2	AI	139	22	-	013	EA1	0
CI-EA1-U1	AI	140	23	-	013	EA1	0
SPARE	AI	141	-	-	-	-	0
SPARE	AI	142	-	-	-	-	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
SPARE	AI	143	-	-	-	-	0
SPARE	AI	144	-	-	-	-	0
SPARE	AI	145	-	-	-	-	0
SPARE	AI	146	-	-	-	-	0
SPARE	AI	147	-	-	-	-	0
SPARE	AI	148	-	-	-	-	0
SPARE	AI	149	-	-	-	-	0
10 - E C 1					10-DEV; 30-AIN; 10-DIN		
EC1-ALARM	DEV	80	-	-	016	EC1	0
HV-EC1-1	DEV	81	-	-	016	EC1	0
HVEC2/3-1	DEV	82	-	-	016	EC1	0
SPARE	DEV	83	-	-	-	-	0
SPARE	DEV	84	-	-	-	-	0
SPARE	DEV	85	-	-	-	-	0
SPARE	DEV	86	-	-	-	-	0
SPARE	DEV	87	-	-	-	-	0
SPARE	DEV	88	-	-	-	-	0
SPARE	DEV	89	-	-	-	-	0
RSH-CA1-1	DI	110	0	-	016	EC1	0
ZS-EC1-1N	DI	111	-	-	016	EC1	0
ZS-EC1-1C	DI	112	-	-	016	EC1	0
ZSEC2/31N	DI	113	-	-	016	EC1	0
ZSEC2/31C	DI	114	-	-	016	EC1	0
SPARE	DI	115	-	-	-	-	0
SPARE	DI	116	-	-	-	-	0
SPARE	DI	117	-	-	-	-	0
SPARE	DI	118	-	-	-	-	0
SPARE	DI	119	-	-	-	-	0
TI-CA1-9	AI	150	1	-	016	EC1	0
TI-EC1-3	AI	151	2	-	016	EC1	0
TI-EC2-3	AI	152	3	-	016	EC1	0
PI-EC1-4	AI	153	4	-	016	EC1	0
PI-EC2-2	AI	154	5	-	016	EC1	0
CI-EC2-Q1	AI	155	6	-	016	EC1	0
CI-EC3-Q1	AI	156	7	-	016	EC1	0
ZI-CA1-7	AI	157	8	-	016	EC1	0
FI-EC2/3	AI	158	9	-	016	EC1	0
TI-EC1-4	AI	159	10	-	016	EC1	0
PI-EC1-5	AI	160	11	23	016	EC1	0
TI-EC2-1	AI	161	12	-	016	EC1	0
FIC-EC3-1	AI	162	13	24	016	EC1	0
TI-EC3-1	AI	163	14	-	016	EC1	0
TI-EC3-2	AI	164	15	-	016	EC1	0
TI-EC2-2	AI	165	16	-	016	EC1	0
PI-EC3-2	AI	166	17	-	016	EC1	0
TI-EC3-3	AI	167	18	-	016	EC1	0
RI-CA1-1	AI	168	19	-	016	EC1	0
FI-EC1-2	AI	169	20	-	016	EC1	0
FQI-EC1-2	AI	170	21	-	016	EC1	0
TIC-EC1-2	AI	171	22	25	016	EC1	0
FQI-EC2/3	AI	172	-	-	016	EC1	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
CI-EC1-U1	AI	173	-	-	016	EC1	0
SPARE	AI	174	-	-	-	-	0
SPARE	AI	175	-	-	-	-	0
SPARE	AI	176	-	-	-	-	0
SPARE	AI	177	-	-	-	-	0
SPARE	AI	178	-	-	-	-	0
SPARE	AI	179	-	-	-	-	0
11 - R W A T E R					10-DEV; 10-AIN; 10-DIN		
RW-ALARM	DEV	90	-	-	026	RW1	0
HV-RW-1	DEV	91	14	-	011	RW	0
HV-RW-2	DEV	92	15	-	011	RW	0
SPARE	DEV	93	-	-	-	-	0
SPARE	DEV	94	-	-	-	-	0
SPARE	DEV	95	-	-	-	-	0
SPARE	DEV	96	-	-	-	-	0
SPARE	DEV	97	-	-	-	-	0
SPARE	DEV	98	-	-	-	-	0
SPARE	DEV	99	-	-	-	-	0
LDS-BFP-1	DI	120	0	-	011	RW	0
LSL-RWSMP	DI	121	1	-	011	RW	0
LDS-BFP-2	DI	122	2	-	011	RW	0
RSH-RW-1	DI	123	3	-	011	RW	0
RXS-RW-1	DI	124	4	-	011	RW	0
ZS-RW-1C	DI	125	-	-	011	RW	0
ZS-RW-1N	DI	126	-	-	011	RW	0
ZS-RW-2C	DI	127	-	-	011	RW	0
ZS-RW-2N	DI	128	-	-	011	RW	0
SPARE	DI	129	-	-	-	-	0
TSL242A81	DI	242	13	-	011	RW	1
PDI-RW-1	AI	180	5	10	011	RW	0
PI-RW-1	AI	181	6	-	011	RW	0
PDI-RW-2	AI	182	7	11	011	RW	0
RI-RW-1	AI	183	8	-	011	RW	0
PI-RW-2	AI	184	9	12	011	RW	0
SPARE	AI	185	-	-	-	-	0
SPARE	AI	186	-	-	-	-	0
SPARE	AI	187	-	-	-	-	0
SPARE	AI	188	-	-	-	-	0
SPARE	AI	189	-	-	-	-	0
12 - U R W					10-DEV; 15-AIN; 10-DIN		
URW-ALARM	DEV	100	-	-	017	URW	0
RC2-SAMP	DEV	101	-	-	017	URW	0
RC2-PIG	DEV	102	17	-	017	URW	0
SPARE	DEV	103	-	-	-	-	0
SPARE	DEV	104	-	-	-	-	0
SPARE	DEV	105	-	-	-	-	0
SPARE	DEV	106	-	-	-	-	0
SPARE	DEV	107	-	-	-	-	0
SPARE	DEV	108	-	-	-	-	0
SPARE	DEV	109	-	-	-	-	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
FSL-RC2-1	DI	130	0	-	017	URW	0
RSH-RC2-1	DI	131	1	-	017	URW	0
RXS-RC2-1	DI	132	2	-	017	URW	0
YS-RC2-6	DI	133	3	-	017	URW	0
SPARE	DI	134	-	-	-	-	0
SPARE	DI	135	-	-	-	-	0
SPARE	DI	136	-	-	-	-	0
SPARE	DI	137	-	-	-	-	0
SPARE	DI	138	-	-	-	-	0
SPARE	DI	139	-	-	-	-	0
FIC-EC1-1	AI	190	4	15,16	017	URW	0
TI-EC1-1	AI	191	5	-	017	URW	0
ZI-EC1-1	AI	192	6	-	017	URW	0
FI-RC2-1	AI	193	7	-	017	URW	0
FQ1-RC2-1	AI	194	8	-	017	URW	0
FQIC-RC2	AI	195	9	-	017	URW	0
RI-RC2-1	AI	196	10	-	017	URW	0
PIC-EC1-2	AI	197	11	-	017	URW	0
PI-EC1-9	AI	198	12	-	017	URW	0
ZI-EC1-2	AI	199	13	-	017	URW	0
KY-RC2-1	AI	200	14	-	017	URW	0
SPARE	AI	201	-	-	-	-	0
SPARE	AI	202	-	-	-	-	0
SPARE	AI	203	-	-	-	-	0
SPARE	AI	204	-	-	-	-	0
13 - V V E N T					10-DEV; 20-AIN; 20-DIN		
VV-ALARM	DEV	110	-	-	020	VVENT	0
EX-C-1	DEV	111	-	-	020	VVENT	0
EXC1-INLK	DEV	112	-	-	020	VVENT	0
HC1-INLK1	DEV	113	-	-	-	-	0
SPARE	DEV	114	-	-	-	-	0
SPARE	DEV	115	-	-	-	-	0
SPARE	DEV	116	-	-	-	-	0
SPARE	DEV	117	-	-	-	-	0
SPARE	DEV	118	-	-	-	-	0
SPARE	DEV	119	-	-	-	-	0
PDSH-FC51	DI	140	0	-	020	VVENT	0
PDSL-FC51	DI	141	1	-	020	VVENT	0
ZS-FC5-1	DI	142	2	-	020	VVENT	0
PDSH-FC52	DI	143	3	-	020	VVENT	0
ZS-FC5-2	DI	144	4	-	020	VVENT	0
YS-EXC-1	DI	145	5	-	020	VVENT	0
YS-EXC3RL	DI	146	6	-	020	VVENT	0
TSHH-HC13	DI	147	7	-	020	VVENT	0
RXS-VVALP	DI	148	8	-	020	VVENT	0
RSH-VVALP	DI	149	9	-	020	VVENT	0
RXS-VVB/G	DI	150	10	-	020	VVENT	0
RSH-VVB/G	DI	151	11	-	020	VVENT	0
QXS-VV-1	DI	152	12	-	020	VVENT	0
SPARE	DI	153	13	-	-	-	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
PDSH-AS-1	DI	154	14	-	020	VVENT	0
PDSL-AS-2	DI	155	15	-	020	VVENT	0
AXS-NH3-1	DI	156	16	-	020	VVENT	0
SPARE	DI	157	-	-	-	-	0
SPARE	DI	158	-	-	-	-	0
SPARE	DI	159	-	-	-	-	0
PDI-FC5-2	AI	205	17	-	020	VVENT	0
PDI-FC5-1	AI	206	18	-	020	VVENT	0
PDI-DUC1	AI	207	19	-	020	VVENT	0
II-EXC-1	AI	208	20	33	020	VVENT	0
PDI-FC6-1	AI	209	21	-	020	VVENT	0
FQI-AS-5	AI	210	22	-	020	VVENT	0
TI-HC1-1	AI	211	23	-	020	VVENT	0
TI-HC1-2	AI	212	24	-	020	VVENT	0
TI-FC5-1	AI	213	25	-	020	VVENT	0
TDIC-HC11	AI	214	26	-	020	VVENT	0
FI-AS-5	AI	215	27	-	020	VVENT	0
PDI-FC512	AI	216	28	-	020	VVENT	0
RI-VVALP	AI	217	29	-	020	VVENT	0
RI-VVB/G	AI	218	30	-	020	VVENT	0
II-AS-1	AI	219	31	34	020	VVENT	0
AI-NH3-1	AI	220	32	-	020	VVENT	0
FQI-NH3-1	AI	221	-	-	-	-	0
SPARE	AI	222	-	-	-	-	0
SPARE	AI	223	-	-	-	-	0
SPARE	AI	224	-	-	-	-	0
14 - P C				10-DEV; 15-AIN; 10-DIN			
PC-ALARM	DEV	120	-	-	018	PC	0
A-C100	DEV	121	-	-	018	PC	0
P-C100	DEV	122	-	-	018	PC	0
PC-INLK1	DEV	123	-	-	018	PC	0
P-C106	DEV	124	-	-	018	PC	0
SPARE	DEV	125	-	-	-	-	0
SPARE	DEV	126	-	-	-	-	0
SPARE	DEV	127	-	-	-	-	0
SPARE	DEV	128	-	-	-	-	0
SPARE	DEV	129	-	-	-	-	0
YS-A-C100	DI	160	0	-	018	PC	0
LSL-C1001	DI	161	1	-	018	PC	0
LSL-SP-1	DI	162	2	-	018	PC	0
YS-P-C100	DI	163	3	-	018	PC	0
YS-RC3-2	DI	164	4	-	018	PC	0
YS-P-C106	DI	165	5	-	018	PC	0
SPARE	DI	166	-	-	-	-	0
SPARE	DI	167	-	-	-	-	0
SPARE	DI	168	-	-	-	-	0
SPARE	DI	169	-	-	-	-	0
II-A-C100	AI	225	6	-	018	PC	0
WFIC-C100	AI	226	7	-	018	PC	0
LI-C100-G	AI	227	8	-	018	PC	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
LI-C100-1	AI	228	9	21	018	PC	0
LI-C100-2	AI	229	10	-	018	PC	0
TI-C100-1	AI	230	11	-	018	PC	0
II-P-C100	AI	231	12	-	018	PC	0
FIC-C1005	AI	232	13	-	018	PC	0
FQI-C100	AI	233	14	-	018	PC	0
PI-FC1-3	AI	234	15	-	018	PC	0
PDI-FC1-1	AI	235	16	-	018	PC	0
PI-FC1-4	AI	236	17	-	018	PC	0
II-P-C106	AI	237	18	-	018	PC	0
PDI-FC5/2	AI	238	19	-	018	PC	0
FQI-PB12W	AI	239	20	-	-	-	0
15 - I X					10-DEV; 25-AIN; 20-DIN		
IX-ALARM	DEV	130	-	-	044	IX	0
HV-RC3-3	DEV	131	-	-	044	IX	0
RC3-SAMP	DEV	132	-	-	044	IX	0
RC3-PIG	DEV	133	33	-	044	IX	0
HV-2-34	DEV	134	-	-	044	IX	0
HV-2-37	DEV	135	-	-	044	IX	0
SPARE	DEV	136	-	-	-	-	0
SPARE	DEV	137	-	-	-	-	0
SPARE	DEV	138	-	-	-	-	0
SPARE	DEV	139	-	-	-	-	0
ZS-IDX-2	DI	170	0	-	044	IX	0
PSH-ILF-1	DI	171	1	-	044	IX	0
YS-RC3-6	DI	172	2	-	044	IX	0
RXS-RC3-1	DI	173	3	-	044	IX	0
RSH-RC3-1	DI	174	4	-	044	IX	0
FSL-RC3-1	DI	175	5	-	044	IX	0
ZS-RC3-4	DI	176	6	-	044	IX	0
ZS-RC3-5	DI	177	7	-	044	IX	0
ZS-IDX1NM	DI	178	-	-	044	IX	0
ZS-IDX-1B	DI	179	-	-	044	IX	0
ZS-IDX-3B	DI	180	-	-	044	IX	0
ZS-IDX3NM	DI	181	-	-	044	IX	0
ZS-RC3-3D	DI	182	-	-	044	IX	0
ZS-RC33NM	DI	183	-	-	044	IX	0
SPARE	DI	184	-	-	-	-	0
SPARE	DI	185	-	-	-	-	0
SPARE	DI	186	-	-	-	-	0
SPARE	DI	187	-	-	-	-	0
SPARE	DI	188	-	-	-	-	0
SPARE	DI	189	-	-	-	-	0
PI-IDX-13	AI	240	8	-	044	IX	0
PI-IDX-14	AI	241	9	-	044	IX	0
PI-IDX-15	AI	242	10	-	044	IX	0
PI-IDX-16	AI	243	11	-	044	IX	0
PDI-IDX-2	AI	244	12	-	044	IX	0
PDI-IDX-4	AI	245	13	-	044	IX	0
PDI-IDX-3	AI	246	14	-	044	IX	0
PI-IDX-12	AI	247	15	-	044	IX	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
PI-IDX-17	AI	248	16	-	044	IX	0
PDI-IDX-5	AI	249	17	-	044	IX	0
PDI-IDX-1	AI	250	18	-	044	IX	0
PI-IDX-18	AI	251	19	-	044	IX	0
PI-ILS-5	AI	252	20	-	044	IX	0
PI-ILS-6	AI	253	21	-	044	IX	0
FI-RC3-1	AI	254	22	-	044	IX	0
PDI-ILS-1	AI	255	23	-	044	IX	0
FQIC-RC3	AI	256	24	-	044	IX	0
FQI-RC3NM	AI	257	25	-	044	IX	0
FQI-RC3-D	AI	258	26	-	044	IX	0
KY-RC3-1	AI	259	27	-	044	IX	0
RI-RC3-1	AI	260	28	-	044	IX	0
SPARE	AI	261	-	-	-	-	0
SPARE	AI	262	-	-	-	-	0
SPARE	AI	263	-	-	-	-	0
SPARE	AI	264	-	-	-	-	0
16 - S C					10-DEV; 15-AIN; 20-DIN		
SC-ALARM	DEV	140	-	-	014	SC	0
HV-RC1-3	DEV	141	-	-	014	SC	0
RC1-SAMP	DEV	142	-	-	014	SC	0
RC1-PIG	DEV	143	24	-	014	SC	0
HV-EA1-2	DEV	144	-	-	014	SC	0
SPARE	DEV	145	-	-	-	-	0
SPARE	DEV	146	-	-	-	-	0
SPARE	DEV	147	-	-	-	-	0
SPARE	DEV	148	-	-	-	-	0
SPARE	DEV	149	-	-	-	-	0
RSH-EA1-1	DI	190	0	-	014	SC	0
LSL-RC1-1	DI	191	1	-	014	SC	0
LSL-C1031	DI	192	2	-	014	SC	0
YS-P-RC1	DI	193	3	-	014	SC	0
YS-RC1-6	DI	194	4	-	014	SC	0
FSL-RC1-1	DI	195	5	-	014	SC	0
RSH-RC1-1	DI	196	6	-	014	SC	0
RXS-RC1-1	DI	197	7	-	014	SC	0
LSL-C1032	DI	198	8	-	014	SC	0
ZS-EA12NM	DI	199	9	-	014	SC	0
ZS-EA1-2D	DI	200	10	-	014	SC	0
ZS-RC13NM	DI	201	11	-	014	SC	0
ZS-RC1-3D	DI	202	12	-	014	SC	0
SPARE	DI	203	-	-	-	-	0
SPARE	DI	204	-	-	-	-	0
SPARE	DI	205	-	-	-	-	0
SPARE	DI	206	-	-	-	-	0
SPARE	DI	207	-	-	-	-	0
SPARE	DI	208	-	-	-	-	0
SPARE	DI	209	-	-	-	-	0
FI-EA1-3	AI	265	13	-	014	SC	0
FQI-EA1-D	AI	266	14	-	014	SC	0
FQI-EA1NM	AI	267	15	-	014	SC	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
RI-EA1-1	AI	268	16	-	014	SC	0
KY-RC1-1	AI	269	17	-	014	SC	0
RI-RC1-1	AI	270	18	-	014	SC	0
FI-RC1-1	AI	271	19	-	014	SC	0
FQI-RC1-D	AI	272	20	-	014	SC	0
FQI-RC1NM	AI	273	21	-	014	SC	0
FQIC-RC1	AI	274	22	-	014	SC	0
CI-EA1-SE	AI	275	23	-	014	SC	0
SPARE	AI	276	-	-	-	-	0
SPARE	AI	277	-	-	-	-	0
SPARE	AI	278	-	-	-	-	0
SPARE	AI	279	-	-	-	-	0
17 - BASINS					10-DEV; 10-AIN; 20-DIN		
BASIN-ALM	DEV	150	-	-	022	SCBASIN	0
P-207A-S	DEV	151	10	-	022	SCBASIN	0
207-A-P3	DEV	152	-	-	022	SCBASIN	0
HV-SC-1A	DEV	153	-	-	022	SCBASIN	0
HV-SC-2A	DEV	154	-	-	022	SCBASIN	0
HV-SC-3A	DEV	155	-	-	022	SCBASIN	0
HV-SC-1B	DEV	156	-	-	022	SCBASIN	0
HV-SC-2B	DEV	157	-	-	022	SCBASIN	0
HV-SC-3B	DEV	158	-	-	022	SCBASIN	0
SPARE	DEV	159	-	-	-	-	0
YS-SCP3IL	DI	210	0	-	022	SCBASIN	0
LSH-P-1	DI	211	1	-	022	SCBASIN	0
YS-P1-1	DI	212	2	-	022	SCBASIN	0
LSHH-SC-1	DI	213	3	-	022	SCBASIN	0
LSH-SC-1	DI	214	4	-	022	SCBASIN	0
LSL-SC-1	DI	215	5	-	022	SCBASIN	0
LSH-SC-2	DI	216	6	-	022	SCBASIN	0
LSL-SC-2	DI	217	7	-	022	SCBASIN	0
LSH-SC-3	DI	218	8	-	022	SCBASIN	0
LSL-SC-3	DI	219	9	-	022	SCBASIN	0
YS-SC-P3	DI	220	10	-	022	SCBASIN	0
ZS-SC-1DC	DI	221	-	-	022	SCBASIN	0
ZS-SC-1FC	DI	222	-	-	022	SCBASIN	0
ZS-SC-2DC	DI	223	-	-	022	SCBASIN	0
ZS-SC-2FC	DI	224	-	-	022	SCBASIN	0
ZS-SC-3DC	DI	225	-	-	022	SCBASIN	0
ZS-SC-3FC	DI	226	-	-	022	SCBASIN	0
SPARE	DI	227	-	-	-	-	0
SPARE	DI	228	-	-	-	-	0
SPARE	DI	229	-	-	-	-	0
II-SUMPP1	AI	280	-	-	022	SCBASIN	0
II-SC-P3	AI	281	-	-	022	SCBASIN	0
SPARE	AI	282	-	-	-	-	0
SPARE	AI	283	-	-	-	-	0
SPARE	AI	284	-	-	-	-	0
SPARE	AI	285	-	-	-	-	0
SPARE	AI	286	-	-	-	-	0
SPARE	AI	287	-	-	-	-	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
SPARE	AI	288	-	-	-	-	0
SPARE	AI	289	-	-	-	-	0
18 - U T I L					10-DEV; 20-AIN; 20-DIN		
UTILITIES	DEV	160	-	-	036	UTIL1	0
SPARE	DEV	161	-	-	-	-	0
SPARE	DEV	162	-	-	-	-	0
SPARE	DEV	163	-	-	-	-	0
SPARE	DEV	164	-	-	-	-	0
SPARE	DEV	165	-	-	-	-	0
SPARE	DEV	166	-	-	-	-	0
SPARE	DEV	167	-	-	-	-	0
SPARE	DEV	168	-	-	-	-	0
SPARE	DEV	169	-	-	-	-	0
PSLCPE1/2	DI	230	0	-	021	UTIL	0
PSL-IA-2	DI	231	1	-	021	UTIL	0
YS-CPE1-1	DI	232	2	-	021	UTIL	0
YS-CPE2-1	DI	233	3	-	021	UTIL	0
JXS-PWR-1	DI	234	4	-	021	UTIL	0
YS-DIEGEN	DI	235	5	-	021	UTIL	0
YS-GEN-1	DI	236	6	-	021	UTIL	0
ZS-PWR-1	DI	237	7	-	021	UTIL	0
JXS-PWR-3	DI	238	8	-	021	UTIL	0
YSL-UPS-1	DI	239	9	-	021	UTIL	0
ZS-UPS-1	DI	240	10	-	021	UTIL	0
SPARE	DI	241	-	-	-	-	0
SPARE	DI	242	-	-	-	-	0
SPARE	DI	243	-	-	-	-	0
SPARE	DI	244	-	-	-	-	0
SPARE	DI	245	-	-	-	-	0
SPARE	DI	246	-	-	-	-	0
SPARE	DI	247	-	-	-	-	0
SPARE	DI	248	-	-	-	-	0
SPARE	DI	249	-	-	-	-	0
PI-IA-1	AI	290	11	15,16	021	UTIL	0
PI-PA-1	AI	291	12	17	021	UTIL	0
TI-AMBS-1	AI	292	13	18-21	021	UTIL	0
TI-AMBS-2	AI	293	14	22-25	021	UTIL	0
SPARE	AI	294	-	-	-	-	0
SPARE	AI	295	-	-	-	-	0
SPARE	AI	296	-	-	-	-	0
SPARE	AI	297	-	-	-	-	0
SPARE	AI	298	-	-	-	-	0
SPARE	AI	299	-	-	-	-	0
RI-K1-1-1	AI	300	-	-	-	-	0
RI-K1-1-2	AI	301	-	-	-	-	0
FI-K1-1	AI	302	-	-	-	-	0
SPARE	AI	303	-	-	-	-	0
SPARE	AI	304	-	-	-	-	0
SPARE	AI	305	-	-	-	-	0
SPARE	AI	306	-	-	-	-	0
SPARE	AI	307	-	-	-	-	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
SPARE	AI	308	-	-	-	-	0
SPARE	AI	309	-	-	-	-	0
19 - R O O M S					10-DEV; 10-AIN; 10-DIN		
ROOM-ALM	DEV	170	-	-	023	242A	0
AUDALARMO	DEV	171	-	-	-	-	0
SPARE	DEV	172	-	-	-	-	0
SPARE	DEV	173	-	-	-	-	0
SPARE	DEV	174	-	-	-	-	0
SPARE	DEV	175	-	-	-	-	0
SPARE	DEV	176	-	-	-	-	0
SPARE	DEV	177	-	-	-	-	0
SPARE	DEV	178	-	-	-	-	0
SPARE	DEV	179	-	-	-	-	0
SIMDIN1	DI	250	-	-	-	-	0
SPARE	DI	251	-	-	-	-	0
SPARE	DI	252	-	-	-	-	0
SPARE	DI	253	-	-	-	-	0
SPARE	DI	254	-	-	-	-	0
SPARE	DI	255	-	-	-	-	0
SPARE	DI	256	-	-	-	-	0
SPARE	DI	257	-	-	-	-	0
SPARE	DI	258	-	-	-	-	0
SPARE	DI	259	-	-	-	-	0
BUILDING	DEV	75	-	-	038	BLDNG1	1
BLDG-EXH	DEV	76	-	-	-	BLDNG	1
RSH-ARAMU	DI	250	0	-	023	242A	1
RSH-ASAMU	DI	251	1	-	023	242A	1
RSH-K1-12	DI	252	2	-	023	242A	1
RSH-K1-11	DI	253	3	-	023	242A	1
QXS-K1	DI	254	4	-	023	242A	1
PDSH-K161	DI	255	5	-	023	242A	1
PDSL-K161	DI	256	6	-	023	242A	1
PDSH-K162	DI	257	7	-	023	242A	1
PDSL-K162	DI	258	8	-	023	242A	1
PDSL-K163	DI	259	9	-	023	242A	1
PDSL-K164	DI	260	10	-	023	242A	1
YS-K1-1	DI	261	11	-	023	242A	1
RSH-ASSUR	DI	262	12	-	023	242A	1
RSH-ASCON	DI	263	13	-	023	242A	1
RSHARHVAC	DI	264	14	-	023	242A	1
RSH-ARC4	DI	265	15	-	023	242A	1
RSH-ARCB	DI	266	16	-	023	242A	1
RSHAREVAP	DI	267	17	-	023	242A	1
RSH-ARHES	DI	268	18	-	023	242A	1
RXS-ASCON	DI	269	19	-	023	242A	1
RXS-ASAMU	DI	270	20	-	023	242A	1
RXS-ASSUR	DI	271	21	-	023	242A	1
RXSAREVAP	DI	272	22	-	023	242A	1
RXS-ARAMU	DI	273	23	-	023	242A	1
RXSARHVAC	DI	274	24	-	023	242A	1

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
RXS-ARHES	DI	275	25	-	023	242A	1
RXS-ARC4	DI	276	26	-	023	242A	1
RXS-ARCB	DI	277	27	-	023	242A	1
RI-AR-AMU	AI	80	28	-	023	242A	1
RI-ASAMU	AI	81	29	-	023	242A	1
RI-ASSUR	AI	82	30	-	023	242A	1
RI-ASCON	AI	83	31	-	023	242A	1
RI-ARHVAC	AI	84	32	-	023	242A	1
RI-AR-C4	AI	85	33	-	023	242A	1
RI-AR-CB	AI	86	34	-	023	242A	1
TI-K1-11	AI	87	35	-	023	242A	1
TI-K1-4	AI	88	36	-	023	242A	1
TI-K1-12A	AI	89	37	-	023	242A	1
TI-K1-12B	AI	90	38	-	023	242A	1
TI-K1-2	AI	91	39	-	023	242A	1
RI-AREVAP	AI	92	40	-	023	242A	1
RI-ARHES	AI	93	41	-	023	242A	1
RI-K1-1-1	AI	300	42	-	023	242A	0
RI-K1-1-2	AI	301	43	-	023	242A	0
FI-K1-1	AI	302	44	-	023	242A	0
SPARE	AI	303	-	-	-	-	0
SPARE	AI	304	-	-	-	-	0
SPARE	AI	305	-	-	-	-	0
SPARE	AI	306	-	-	-	-	0
SPARE	AI	307	-	-	-	-	0
SPARE	AI	308	-	-	-	-	0
SPARE	AI	309	-	-	-	-	0
20 - A M U					10-DEV; 20-AIN; 10-DIN		
AMU-ALARM	DEV	180	-	-	019	AMU	0
P-E-101	DEV	181	-	-	019	AMU	0
A-E-101	DEV	182	-	-	019	AMU	0
A-E-102	DEV	183	-	-	019	AMU	0
P-E-102	DEV	184	-	-	019	AMU	0
P-E-104	DEV	185	-	-	019	AMU	0
A-E-104	DEV	186	-	-	019	AMU	0
SPARE	DEV	187	-	-	-	-	0
SPARE	DEV	188	-	-	-	-	0
SPARE	DEV	189	-	-	-	-	0
YS-A-E101	DI	260	0	-	019	AMU	0
YS-P-E101	DI	261	1	-	019	AMU	0
YS-A-E102	DI	262	2	-	019	AMU	0
YS-P-E102	DI	263	3	-	019	AMU	0
YS-A-E104	DI	264	4	-	019	AMU	0
YS-P-E104	DI	265	5	-	019	AMU	0
SPARE	DI	266	-	-	-	-	0
SPARE	DI	267	-	-	-	-	0
SPARE	DI	268	-	-	-	-	0
SPARE	DI	269	-	-	-	-	0
TI-E101-2	AI	310	6	20	019	AMU	0
WFI-E101	AI	311	7	21	019	AMU	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
DI-E101-2	AI	312	8	22	019	AMU	0
LI-E101-1	AI	313	9	-	019	AMU	0
FI-E102-1	AI	314	10	-	019	AMU	0
WFI-E102	AI	315	11	23	019	AMU	0
FQI-E102	AI	316	12	24	019	AMU	0
TI-E104-2	AI	317	13	-	019	AMU	0
WFI-E104	AI	318	14	25	019	AMU	0
DI-E104-2	AI	319	15	26	019	AMU	0
LI-E104-1	AI	320	16	-	019	AMU	0
FQI-E104	AI	321	17	-	019	AMU	0
FI-E104-2	AI	322	18	-	019	AMU	0
FIC-E102	AI	323	19	-	019	AMU	0
SPARE	AI	324	-	-	-	-	0
SPARE	AI	325	-	-	-	-	0
SPARE	AI	326	-	-	-	-	0
SPARE	AI	327	-	-	-	-	0
SPARE	AI	328	-	-	-	-	0
SPARE	AI	329	-	-	-	-	0
21 - L E R F					10-DEV; 20-AIN; 20-DIN		
LERF-ALM	DEV	190	-	-	024	LERF3	0
LERFILK	DEV	191	-	-	024	LERF3	0
SPARE	DEV	192	-	-	-	-	0
SPARE	DEV	193	-	-	-	-	0
SPARE	DEV	194	-	-	-	-	0
SPARE	DEV	195	-	-	-	-	0
SPARE	DEV	196	-	-	-	-	0
SPARE	DEV	197	-	-	-	-	0
SPARE	DEV	198	-	-	-	-	0
SPARE	DEV	199	-	-	-	-	0
YS-P-43	DI	270	0	-	024	LERF3	0
LDS-A1	DI	271	1	-	024	LERF3	0
YS-P-44	DI	272	2	-	024	LERF3	0
SPARE	DI	273	3	-	-	-	0
SPARE	DI	274	4	-	-	-	0
YS-P-42	DI	275	5	-	024	LERF3	0
LDS-BSN42	DI	276	6	-	024	LERF3	0
LDS-BSN43	DI	277	7	-	024	LERF3	0
LDS-BSN44	DI	278	8	-	024	LERF3	0
LXS-LCH42	DI	279	9	-	024	LERF3	0
LXS-BSN42	DI	280	10	-	024	LERF3	0
LXS-LCH43	DI	281	11	-	024	LERF3	0
LXS-BSN43	DI	282	12	-	024	LERF3	0
LXS-LCH44	DI	283	13	-	024	LERF3	0
LXS-BSN44	DI	284	14	-	024	LERF3	0
YS-PLC-1	DI	285	15	-	024	LERF3	0
SPARE	DI	286	-	-	-	-	0
SPARE	DI	287	-	-	-	-	0
SPARE	DI	288	-	-	-	-	0
SPARE	DI	289	-	-	-	-	0
LDI-A1	AI	330	20	-	024	LERF3	0
LI-LCH42	AI	331	21	-	024	LERF3	0

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
LI-BSN42	AI	332	22	-	024	LERF3	0
LI-LCH43	AI	333	23	-	024	LERF3	0
LI-BSN43	AI	334	24	-	024	LERF3	0
LI-LCH44	AI	335	25	-	024	LERF3	0
LI-BSN44	AI	336	26	-	024	LERF3	0
SPARE	AI	337	-	-	-	-	0
SPARE	AI	338	-	-	-	-	0
SPARE	AI	339	-	-	-	-	0
SPARE	AI	340	-	-	-	-	0
SPARE	AI	341	-	-	-	-	0
SPARE	AI	342	-	-	-	-	0
SPARE	AI	343	-	-	-	-	0
SPARE	AI	344	-	-	-	-	0
SPARE	AI	345	-	-	-	-	0
SPARE	AI	346	-	-	-	-	0
SPARE	AI	347	-	-	-	-	0
SPARE	AI	348	-	-	-	-	0
SPARE	AI	349	-	-	-	-	0
22 - E V A P					80-DEV; 155-AIN; 110-DIN		
PAW102ALM	DEV	15	-	-	040	OVER1	0
YS-AW102	DI	50	-	-	040	OVER1	0
II-AW-102	AI	30	-	-	040	OVER1	0
23 - V A P O R					30-DEV; 90-AIN; 60-DIN		
EC1-ALARM	DEV	80	-	-	041	OVER2	0
RSH-CA1-1	DI	110	-	-	041	OVER2	0
TI-CA1-9	AI	155	-	-	041	OVER2	0
24 - C O N D					85-DEV; 135-AIN; 100-DIN		
PC-ALARM	DEV	120	-	-	042	OVER3	0
YS-A-C100	DI	160	-	-	042	OVER3	0
II-A-C100	AI	225	-	-	042	OVER3	0
25 - B L A N K O					0-DEV; 0-AIN; 0-DIN		
** - - - M I S C E L L A N E O U S - - -							
PB2GLOBAL	DEV	60	-	-	-	-	0
AUDALARMO	DEV	171	-	-	-	-	0
CASSOAL1A	DEV	200	-	-	090	CASS	0
CASSOAL2A	DEV	202	-	-	090	CASS	0
CASSOAL2B	DEV	203	-	-	090	CASS	0
GBIT0-0/1	DEV	204	-	-	090	CASS	0
HS-AB1ACK	DEV	208	-	-	-	-	0
0 - W H O L E 1					90-DEV; 130-AIN; 280-DIN		
P-AN-101	DEV	0	-	-	-	-	1

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
YS-AN-101	DI	0	-	-	-	-	1
II-AN1011	AI	0	-	-	-	-	1
1 - A N F A R M					15-DEV; 20-AIN; 60-DIN		
P-AN-101	DEV	0	-	-	270	ANFRM	1
P-AN-102	DEV	1	-	-	270	ANFRM	1
P-AN-103	DEV	2	-	-	270	ANFRM	1
P-AN-104	DEV	3	-	-	270	ANFRM	1
P-AN-105	DEV	4	-	-	270	ANFRM	1
P-AN-106	DEV	5	-	-	270	ANFRM	1
P-AN-107	DEV	6	-	-	270	ANFRM	1
HS-AN-SIL	DEV	7	-	-	-	-	1
ANFARM	DEV	8	0	-	270	ANFRM	1
AN-SERPIT	DEV	9	1	-	-	-	1
AN-VLVPIT	DEV	10	2	-	272	ANVLV	1
AN-HVAC	DEV	11	3	-	273	ANHVC	1
AN-271	DEV	12	4	-	274	AN271	1
HS-ARM-AN	DEV	13	5	-	274	AN271	1
SPARE	DEV	14	-	-	-	-	1
YS-AN-101	DI	0	1	-	270	ANFRM	1
RSH-AN101	DI	1	2	-	270	ANFRM	1
RSH-AN01C	DI	2	3	-	270	ANFRM	1
YS-AN-102	DI	3	4	-	270	ANFRM	1
RSH-AN102	DI	4	5	-	270	ANFRM	1
RSH-AN02C	DI	5	6	-	270	ANFRM	1
YS-AN-103	DI	6	7	-	270	ANFRM	1
RSH-AN103	DI	7	8	-	270	ANFRM	1
RSH-AN03C	DI	8	9	-	270	ANFRM	1
YS-AN-104	DI	9	10	-	270	ANFRM	1
RSH-AN104	DI	10	11	-	270	ANFRM	1
RSH-AN04C	DI	11	12	-	270	ANFRM	1
YS-AN-105	DI	12	13	-	270	ANFRM	1
RSH-AN105	DI	13	14	-	270	ANFRM	1
RSH-AN05C	DI	14	15	-	270	ANFRM	1
YS-AN-106	DI	15	16	-	270	ANFRM	1
RSH-AN106	DI	16	17	-	270	ANFRM	1
RSH-AN06C	DI	17	18	-	270	ANFRM	1
YS-AN-107	DI	18	19	-	270	ANFRM	1
RSH-AN107	DI	19	20	-	270	ANFRM	1
RSH-AN07C	DI	20	21	-	270	ANFRM	1
LDS-SP-AN	DI	21	22	-	272	ANVLV	1
RSH-SP-AN	DI	22	23	-	272	ANVLV	1
LDS-AN-PP	DI	23	24	-	272	ANVLV	1
LDS-ANSL1	DI	24	25	-	272	ANVLV	1
LDS-ANSN2	DI	25	26	-	272	ANVLV	1
LDS-COBAN	DI	26	27	-	272	ANVLV	1
PSH-ANSN1	DI	27	28	-	272	ANVLV	1
LDS-DR368	DI	28	29	-	272	ANVLV	1
RSH-AN-K1	DI	29	30	-	273	ANHVC	1
PSH-ANK11	DI	30	31	-	273	ANHVC	1
PSH-ANK12	DI	31	32	-	273	ANHVC	1

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
RXS-AN-K1	DI	32	33	-	273	ANHVC	1
RSH-AN-K2	DI	33	34	-	273	ANHVC	1
PSH-ANK21	DI	34	35	-	273	ANHVC	1
PSH-ANK22	DI	35	36	-	273	ANHVC	1
RXS-AN-K2	DI	36	37	-	273	ANHVC	1
JXS-AN271	DI	37	38	-	274	AN271	1
RSHAN2711	DI	38	39	-	274	AN271	1
RSHAN2712	DI	39	40	-	274	AN271	1
RSH-AN-12	DI	40	41	-	274	AN271	1
QS-AN-271	DI	41	42	-	274	AN271	1
TXS-AN-2	DI	42	43	-	274	AN271	1
YS-AN-ARM	DI	43	44	-	274	AN271	1
YS-AN-MSD	DI	44	45	-	274	AN271	1
SPARE	DI	45	-	-			1
SPARE	DI	46	-	-			1
SPARE	DI	47	-	-			1
SPARE	DI	48	-	-			1
SPARE	DI	49	-	-			1
SPARE	DI	50	-	-			1
SPARE	DI	51	-	-			1
SPARE	DI	52	-	-			1
SPARE	DI	53	-	-			1
SPARE	DI	54	-	-			1
SPARE	DI	55	-	-			1
SPARE	DI	56	-	-			1
SPARE	DI	57	-	-			1
SPARE	DI	58	-	-			1
SPARE	DI	59	-	-			1
II-AN1011	AI	0	-	-	270	ANFRM	1
II-AN1021	AI	1	-	-	270	ANFRM	1
II-AN1031	AI	2	-	-	270	ANFRM	1
II-AN1041	AI	3	-	-	270	ANFRM	1
II-AN1051	AI	4	-	-	270	ANFRM	1
II-AN1061	AI	5	-	-	270	ANFRM	1
II-AN1071	AI	6	-	-	270	ANFRM	1
FQI-AN-1	DI	7	-	-	272	ANVLV	1
KY-ARM-AN	AI	8	-	-	274	AN271	1
SIMAI-1	AI	9	-	-	-	-	1
SPARE	AI	10	-	-	-	-	1
SPARE	AI	11	-	-	-	-	1
SPARE	AI	12	-	-	-	-	1
SPARE	AI	13	-	-	-	-	1
SPARE	AI	14	-	-	-	-	1
SPARE	AI	15	-	-	-	-	1
SPARE	AI	16	-	-	-	-	1
SPARE	AI	17	-	-	-	-	1
SPARE	AI	18	-	-	-	-	1
SPARE	AI	19	-	-	-	-	1
2 - A P F A R M				20-DEV; 20-AIN; 90-DIN			
P-AP-101	DEV	20	84	-	290	APFRM	1
P-AP-102	DEV	21	85	-	290	APFRM	1
P-AP-103	DEV	22	86	-	290	APFRM	1

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
P-AP-104	DEV	23	87	-	290	APFRM	1
P-AP-105	DEV	24	88	-	290	APFRM	1
P-AP-106	DEV	25	89	-	290	APFRM	1
P-AP-107	DEV	26	90	-	290	APFRM	1
P-AP-108	DEV	27	91	-	290	APFRM	1
SPARE	DEV	28	-	-	-	-	1
SPARE	DEV	29	-	-	-	-	1
EXH-AP-K1	DEV	30	-	-	-	-	1
EXH-AP-K2	DEV	31	-	-	-	-	1
HS-AP-SIL	DEV	32	-	-	-	-	1
APFARM	DEV	33	0	-	290	APFRM	1
AP-SERPIT	DEV	34	1	-	-	-	1
AP-VLVPIT	DEV	35	2	-	292	APVLV	1
AP-HVAC	DEV	36	3	-	294	APHVC	1
AP-271	DEV	37	4	-	295	AP271	1
SPARE	DEV	38	-	-	-	-	1
HS-ARM-AP	DEV	39	-	-	295	AP271	1
YS-AP-101	DI	60	1	-	290	APFRM	1
RSH-101AP	DI	61	2	-	290	APFRM	1
RXS-101AP	DI	62	3	-	290	APFRM	1
YS-AP-102	DI	63	4	-	291	AP102	1
RSH-102AP	DI	64	5	-	290	APFRM	1
RXS-102AP	DI	65	6	-	290	APFRM	1
YS-AP-103	DI	66	7	-	290	APFRM	1
RSH-103AP	DI	67	8	-	290	APFRM	1
RXS-103AP	DI	68	9	-	290	APFRM	1
RSH-AP03C	DI	69	10	-	290	APFRM	1
RXS-AP03C	DI	70	11	-	290	APFRM	1
LDS-AP03D	DI	71	12	-	290	APFRM	1
YS-AP-104	DI	72	13	-	290	APFRM	1
RSH-104AP	DI	73	14	-	290	APFRM	1
RXS-104AP	DI	74	15	-	290	APFRM	1
YS-AP-105	DI	75	16	-	290	APFRM	1
RSH-105AP	DI	76	17	-	290	APFRM	1
RXS-105AP	DI	77	18	-	290	APFRM	1
RSH-AP05C	DI	78	19	-	290	APFRM	1
RXS-AP05C	DI	79	20	-	290	APFRM	1
YS-AP-106	DI	80	21	-	290	APFRM	1
RSH-106AP	DI	81	22	-	290	APFRM	1
RXS-106AP	DI	82	23	-	290	APFRM	1
YS-AP-107	DI	83	24	-	290	APFRM	1
RSH-107AP	DI	84	25	-	290	APFRM	1
RXS-107AP	DI	85	26	-	290	APFRM	1
YS-AP-108	DI	86	27	-	290	APFRM	1
RSH-108AP	DI	87	28	-	290	APFRM	1
RXS-108AP	DI	88	29	-	290	APFRM	1
LDSAP02DC	DI	89	30	-	291	AP102	1
SPARE	DI	90	31	-	-	-	1
SPARE	DI	91	32	-	-	-	1
SPARE	DI	92	33	-	-	-	1
SPARE	DI	93	34	-	-	-	1
SPARE	DI	94	35	-	-	-	1
SPARE	DI	95	36	-	-	-	1

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
SPARE	DI	96	37	-	-	-	1
SPARE	DI	97	38	-	-	-	1
YS-AP-MSD	DI	98	39	-	295	AP271	1
YS-AP-ARM	DI	99	40	-	295	AP271	1
RSH-AP-2	DI	100	41	-	295	AP271	1
RXS-AP-2	DI	101	42	-	295	AP271	1
QS-AP-1	DI	102	43	-	295	AP271	1
TXS-AP-2	DI	103	44	-	295	AP271	1
LDS-AP-A	DI	104	45	-	295	AP271	1
LDS-AP-B	DI	105	46	-	295	AP271	1
LDS-AP-C	DI	106	47	-	295	AP271	1
JXS-AP-1	DI	107	48	-	295	AP271	1
FSLAP2711	DI	108	49	-	295	AP271	1
RSH-AP271	DI	109	50	-	295	AP271	1
RXS-AP271	DI	110	51	-	295	AP271	1
PSH-APFP1	DI	111	52	-	292	APVLV	1
ZS-AP-FP1	DI	112	53	-	292	APVLV	1
LDS-AP-FP	DI	113	54	-	292	APVLV	1
YS-AP-K1	DI	114	55	-	294	APHVC	1
FSL-APK13	DI	115	56	-	294	APHVC	1
RSH-APK11	DI	116	57	-	294	APHVC	1
RXS-APK11	DI	117	58	-	294	APHVC	1
FSL-APK12	DI	118	59	-	294	APHVC	1
FSL-APK11	DI	119	60	-	294	APHVC	1
LDS-AP-K1	DI	120	61	-	294	APHVC	1
YS-AP-K2	DI	121	62	-	294	APHVC	1
FSL-APK23	DI	122	63	-	294	APHVC	1
RSH-APK21	DI	123	64	-	294	APHVC	1
RXS-APK21	DI	124	65	-	294	APHVC	1
FSL-APK22	DI	125	66	-	294	APHVC	1
FSL-APK21	DI	126	67	-	294	APHVC	1
LDS-AP-SP	DI	127	68	-	292	APVLV	1
RSH-AP-SP	DI	128	69	-	292	APVLV	1
RXS-AP-SP	DI	129	70	-	292	APVLV	1
LDS-AP-SL	DI	130	71	-	292	APVLV	1
LDS-AP-SN	DI	131	72	-	292	APVLV	1
LDSAP621A	DI	132	73	-	291	AP102	1
LDSAP6502	DI	133	74	-	291	AP102	1
PSH-AP-2	DI	134	75	-	295	AP271	1
PSL-AP-2	DI	135	76	-	295	AP271	1
LDS-AP-VP	DI	136	77	-	292	APVLV	1
FSH-DR712	DI	137	78	-	292	APVLV	1
FXS-DR712	DI	138	79	-	292	APVLV	1
PSH-APFL1	DI	139	80	-	292	APVLV	1
ZS-AP-DR1	DI	140	81	-	292	APVLV	1
ZS-AP-FL1	DI	141	82	-	292	APVLV	1
SPARE	DI	142	83	-	-	-	1
SPARE	DI	143	-	-	-	-	1
SPARE	DI	144	-	-	-	-	1
SPARE	DI	145	-	-	-	-	1
SPARE	DI	146	-	-	-	-	1
SPARE	DI	147	-	-	-	-	1
SPARE	DI	148	-	-	-	-	1
SPARE	DI	149	-	-	-	-	1

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
II-API011	AI	20	-	-	290	APFRM	1
II-API021	AI	21	-	-	291	AP102	1
II-API031	AI	22	-	-	290	APFRM	1
II-API041	AI	23	-	-	290	APFRM	1
II-API051	AI	24	-	-	290	APFRM	1
II-API061	AI	25	-	-	290	APFRM	1
II-API071	AI	26	-	-	290	APFRM	1
II-API081	AI	27	-	-	290	APFRM	1
SPARE	AI	28	-	-	-	-	1
SPARE	AI	29	-	-	-	-	1
SPARE	AI	30	-	-	-	-	1
SPARE	AI	31	-	-	-	-	1
SPARE	AI	32	-	-	-	-	1
FQI-APSP1	AI	33	-	-	292	APVLV	1
KY-ARM-AP	AI	34	-	-	295	AP271	1
SPARE	AI	35	-	-	-	-	1
SPARE	AI	36	-	-	-	-	1
SPARE	AI	37	-	-	-	-	1
SPARE	AI	38	-	-	-	-	1
SPARE	AI	39	-	-	-	-	1

3 - A W F A R M**20-DEV; 20-AIN; 60-DIN**

P-AW-101	DEV	40	-	-	300	AWFRM	1
P-AW-103	DEV	41	-	-	300	AWFRM	1
P-AW-104	DEV	42	-	-	300	AWFRM	1
P-AW-105	DEV	43	-	-	300	AWFRM	1
P-AW-106	DEV	44	-	-	300	AWFRM	1
HS-AW-SIL	DEV	45	-	-	-	-	1
AWFARM	DEV	46	1	-	300	AWFRM	1
AW-SERPIT	DEV	47	2	-	-	-	1
AW-VLVPIT	DEV	48	3	-	303	AWVLV	1
AW-HVAC	DEV	49	4	-	304	AWHVC	1
AW-271	DEV	50	5	-	302	AW271	1
AW-102	DEV	51	6	-	-	-	1
HS-ARM-AW	DEV	52	-	-	302	AW271	1
SPARE	DEV	53	-	-	-	-	1
SPARE	DEV	54	-	-	-	-	1
SPARE	DEV	55	-	-	-	-	1
SPARE	DEV	56	-	-	-	-	1
SPARE	DEV	57	-	-	-	-	1
SPARE	DEV	58	-	-	-	-	1
SPARE	DEV	59	-	-	-	-	1
YS-AW-101	DI	150	0	-	300	AWFRM	1
RSH-AW101	DI	151	1	-	300	AWFRM	1
RSH-AW01C	DI	152	2	-	300	AWFRM	1
RSH-AW102	DI	153	3	-	300	AWFRM	1
RSH-AW02C	DI	154	4	-	300	AWFRM	1
YS-AW-103	DI	155	5	-	300	AWFRM	1
RSH-AW103	DI	156	6	-	300	AWFRM	1
RSH-AW03C	DI	157	7	-	300	AWFRM	1
YS-AW-104	DI	158	8	-	300	AWFRM	1
RSH-AW104	DI	159	9	-	300	AWFRM	1

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
RSH-AW04C	DI	160	10	-	300	AWFRM	1
YS-AW-105	DI	161	11	-	300	AWFRM	1
RSH-AW105	DI	162	12	-	300	AWFRM	1
RSH-AW05C	DI	163	13	-	300	AWFRM	1
YS-AW-106	DI	164	14	-	300	AWFRM	1
RSH-AW106	DI	165	15	-	300	AWFRM	1
RSH-AW06C	DI	166	16	-	-	-	1
LDS-TL021	DI	167	17	-	303	AWVLV	1
LDSTL2223	DI	168	18	-	303	AWVLV	1
LDS-AW-DR	DI	169	19	-	303	AWVLV	1
LDS-AW-PP	DI	170	20	-	303	AWVLV	1
LDS-AW-SL	DI	171	21	-	303	AWVLV	1
LDS-AW-SN	DI	172	22	-	303	AWVLV	1
LDS-SN269	DI	173	23	-	303	AWVLV	1
LDS-SN270	DI	174	24	-	303	AWVLV	1
LDS-COBAA	DI	175	25	-	303	AWVLV	1
PSH-AWSN1	DI	176	26	-	303	AWVLV	1
PSH-AWSN2	DI	177	27	-	303	AWVLV	1
QS-AW-271	DI	178	28	-	302	AW271	1
RSH-SP-AW	DI	179	29	-	303	AWVLV	1
LDS-SP-AW	DI	180	30	-	303	AWVLV	1
RSH-AW-K1	DI	181	31	-	304	AWHVC	1
SPARE	DI	182	32	-	-	-	1
SPARE	DI	183	33	-	-	-	1
SPARE	DI	184	34	-	-	-	1
SPARE	DI	185	35	-	-	-	1
QXS-AWK15	DI	186	36	-	304	AWHVC	1
RSH-AW-K2	DI	187	37	-	304	AWHVC	1
QXSAWK1A2	DI	188	38	-	304	AWHVC	1
RSH-AW-12	DI	189	39	-	302	AW271	1
RSHAW2711	DI	190	40	-	302	AW271	1
RSHAW2712	DI	191	41	-	302	AW271	1
YS-AW-ARM	DI	192	42	-	302	AW271	1
JXS-AW271	DI	193	43	-	302	AW271	1
SPARE	DI	194	44	-	-	-	1
YS-AW-MSD	DI	195	45	-	302	AW271	1
TXS-AW-2	DI	196	46	-	302	AW271	1
SPARE	DI	197	-	-	-	-	1
SPARE	DI	198	-	-	-	-	1
SPARE	DI	199	-	-	-	-	1
SPARE	DI	200	-	-	-	-	1
SPARE	DI	201	-	-	-	-	1
SPARE	DI	202	-	-	-	-	1
SPARE	DI	203	-	-	-	-	1
SPARE	DI	204	-	-	-	-	1
SPARE	DI	205	-	-	-	-	1
SPARE	DI	206	-	-	-	-	1
SPARE	DI	207	-	-	-	-	1
SPARE	DI	208	-	-	-	-	1
SPARE	DI	209	-	-	-	-	1
SPARE	DI	210	-	-	-	-	1
II-AW1011	AI	40	-	-	300	AWFRM	1
II-AW1031	AI	41	-	-	300	AWFRM	1

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
II-AW1041	AI	42	-	-	300	AWFRM	1
II-AW1051	AI	43	-	-	300	AWFRM	1
II-AW1061	AI	44	-	-	300	AWFRM	1
FQI-AW-1	DI	45	-	-	303	AWVLV	1
KY-ARM-AW	AI	46	-	-	302	AW271	1
SPARE	AI	47	-	-	-	-	1
SPARE	AI	48	-	-	-	-	1
SPARE	AI	49	-	-	-	-	1
SPARE	AI	50	-	-	-	-	1
SPARE	AI	51	-	-	-	-	1
SPARE	AI	52	-	-	-	-	1
SPARE	AI	53	-	-	-	-	1
SPARE	AI	54	-	-	-	-	1
SPARE	AI	55	-	-	-	-	1
SPARE	AI	56	-	-	-	-	1
SPARE	AI	57	-	-	-	-	1
SPARE	AI	58	-	-	-	-	1
SPARE	AI	59	-	-	-	-	1

4 - X Y Z F R M

15-DEV; 60-AIN; 60-DIN

P-AY-101	DEV	60	-	31	322	AYFRM	1
P-AY-102	DEV	61	-	32	322	AYFRM	1
P-AZ-101	DEV	62	-	33	323	AZFRM	1
P-AZ-102	DEV	63	-	34	323	AZFRM	1
AFARM	DEV	64	0	-	-	-	1
AYFARM	DEV	65	1	-	-	-	1
AZFARM	DEV	66	2	-	-	-	1
AXYZ	DEV	67	3	-	-	-	1
XYZLEAK	DEV	68	-	-	-	-	1
HSAXYZSIL	DEV	69	-	-	-	-	1
HS-ARMAAX	DEV	70	-	-	-	-	1
SPARE	DEV	71	-	-	-	-	1
SPARE	DEV	72	-	-	-	-	1
SPARE	DEV	73	-	-	-	-	1
SPARE	DEV	74	-	-	-	-	1

PSH-SL100	DI	214	5	-	321	AFRM	1
PSH-SL101	DI	215	6	-	321	AFRM	1
PSH-SL113	DI	216	7	-	321	AFRM	1
PSH-SL114	DI	217	8	-	321	AFRM	1
PSH200213	DI	218	9	-	321	AFRM	1
PSH201214	DI	219	10	-	321	AFRM	1
LDS-SL500	DI	220	11	-	323	AZFRM	1
LDS-SL501	DI	221	12	-	323	AZFRM	1
LDS-SL502	DI	222	13	-	322	AYFRM	1
LDS-SL503	DI	223	14	-	322	AYFRM	1
LDS-SL504	DI	224	15	-	322	AYFRM	1
LDS-SL505	DI	225	16	-	322	AYFRM	1
LDS-SN600	DI	226	17	-	323	AZFRM	1
LDS-SN601	DI	227	18	-	323	AZFRM	1
LDS-DR700	DI	228	19	-	322	AYFRM	1
LXS-AXPOT	DI	229	20	-	320	AXYZFRM	1
LSH-AX-1	DI	230	21	-	320	AXYZFRM	1
LSL-AX-1	DI	231	22	-	320	AXYZFRM	1

MCS EPN	STG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
LDSOBAAX	DI	232	23	-	320	AXYZFRM	1
YS-AAXARM	DI	233	24	-	320	AXYZFRM	1
YS-AAXMSD	DI	234	25	-	320	AXYZFRM	1
RSHAAX3/4	DI	235	26	-			1
RSH-AXZY1	DI	236	27	-	320	AXYZFRM	1
LDSPPCOB1	DI	237	28	-	320	AXYZFRM	1
RXSAA3/4	DI	238	29	-	320	AXYZFRM	1
SPARE	DI	239	30	-	-	-	1
QS-204AR1	DI	240	-	-	-	-	1
QS-204AR2	DI	241	-	-	-	-	1
TSL242A81	DI	242	-	-			1
LDS216A40	DI	243	-	-			1
YS-241A1A	DI	244	-	-			1
SPARE	DI	245	-	-			1
SPARE	DI	246	-	-			1
SPARE	DI	247	-	-			1
SPARE	DI	248	-	-			1
SPARE	DI	249	-	-			1
RSH-ARAMU	DI	250	-	-			1
RSH-ASAMU	DI	251	-	-			1
RSH-K1-12	DI	252	-	-			1
RSH-K1-11	DI	253	-	-			1
QXS-K1	DI	254	-	-			1
PDSH-K161	DI	255	-	-			1
PDSL-K161	DI	256	-	-			1
PDSH-K162	DI	257	-	-			1
PDSL-K162	DI	258	-	-			1
PDSL-K163	DI	259	-	-			1
PDSL-K164	DI	260	-	-			1
YS-K1-1	DI	261	-	-			1
RSH-ASSUR	DI	262	-	-			1
RSH-ASCON	DI	263	-	-			1
RSHARHVAC	DI	264	-	-			1
RSH-ARC4	DI	265	-	-			1
RSH-ARCB	DI	266	-	-			1
RSHAREVAP	DI	267	-	-			1
RSH-ARHES	DI	268	-	-			1
RXS-ASCON	DI	269	-	-			1
RXS-ASAMU	DI	270	-	-			1
RXS-ASSUR	DI	271	-	-			1
RXSAREVAP	DI	272	-	-			1
RXS-ARAMU	DI	273	-	-			1
RXSARHVAC	DI	274	-	-			1
PSH-SN215	DI	15	35	-	321	AFRM	0
PSH-SN216	DI	16	36	-	321	AFRM	0
II-AY-101	AI	60	-	-	322	AYFRM	
II-AY-102	AI	61	-	-	322	AYFRM	
II-AZ-101	AI	62	-	-	323	AZFRM	
II-AZ-102	AI	63	-	-	323	AZFRM	
KY-ARMAAX	AI	67	-	-	320	AXYZFRM	
55 MORE AI'S		68 - 119		-			

MCS EPN	SIG TYPE	IPN	LOOP INT	WHITE INT	GRAPH	ACAD	PCM
6 - B L A N K 1				5-DEV; 0-AIN; 0-DIN			
GBIT1-0/1	DEV	85	-	-	-	-	
CASSIAL1A	DEV	86	-	-	090	CASS	
CASSIAL2A	DEV	87	-	-	090	CASS	
WHITE	DEV	88	-	-	-	-	
SPARE	DEV	89	-	-	-	-	
** - - - M I S C E L L A N E O U S - - -							
SIMDEV1	DEV	14	-	-	-	-	
HS-AB2ACK	DEV	15	-	-	-	-	
CNDSR-RM	DEV	77	-	-	-	-	
QS-204AR1	DI	240	-	-	FP-81	-	1
QS-204AR2	DI	241	-	-	FP-81	-	1
YS-241A1A	DI	244	-	-	FP-81	-	1

APPENDIX B -SEQUENCE DATABASE (SDB)

PCMOASDB

WHC-SD-534-CSNL-001 REV 3

PCM name for which to create the SDB?> = pcm0a
 Number of sequence units for PCM?> = 26

Unit 0

Enter unit 0 sequence unit name?> = whole0
 Enter unit 0 console enable index?> = 0
 Read/Write unit 0 from other sequence units[Y/N]?> = y
 Enter unit 0 UPF name?> = total0
 Enter number of DEV for unit 0?> = 217
     Enter EPN of first DEV?> = 244alarm
 Enter number of AIN for unit 0?> = 379
     Enter EPN of first AIN?> = ii-p244a1
 Enter number of DIN for unit 0?> = 348
     Enter EPN of first DIN?> = ys244a1i1
 Enter number of DOT for unit 0?> = 0
 Enter number of DGR for unit 0?> = 5
     Enter EPN of first DGR?> = dgr0

Unit 1

Enter unit 1 sequence unit name?> = a244
 Enter unit 1 console enable index?> = 0
 Read/Write unit 1 from other sequence units[Y/N]?> = y
 Enter unit 1 UPF name?> = short0
 Enter number of DEV for unit 1?> = 15
     Enter EPN of first DEV?> = 244alarm
 Enter number of AIN for unit 1?> = 30
     Enter EPN of first AIN?> = ii-p244a1
 Enter number of DIN for unit 1?> = 50
     Enter EPN of first DIN?> = ys244a1i1
 Enter number of DOT for unit 1?> = 0
 Enter number of DGR for unit 1?> = 0

Unit 2

Enter unit 2 sequence unit name?> = hvc244
 Enter unit 2 console enable index?> = 0
 Read/Write unit 2 from other sequence units[Y/N]?> = y
 Enter unit 2 UPF name?> = short0
 Enter number of DEV for unit 2?> = 1
     Enter EPN of first DEV?> = hvac244
 Enter number of AIN for unit 2?> = 5
     Enter EPN of first AIN?> = ri-244ann
 Enter number of DIN for unit 2?> = 5
     Enter EPN of first DIN?> = pds1-ef-2
 Enter number of DOT for unit 2?> = 0
 Enter number of DGR for unit 2?> = 0

Unit 3

```

Enter unit 3 sequence unit name?> = waste
Enter unit 3 console enable index?> = 0
Read/Write unit 3 from other sequence units[Y/N]?> = y
Enter unit 3 UPF name?> = short0
Enter number of DEV for unit 3?> = 1
    Enter EPN of first DEV?> = wstxfr
Enter number of AIN for unit 3?> = 0
Enter number of DIN for unit 3?> = 18
    Enter EPN of first DIN?> = lds-ar151
Enter number of DOT for unit 3?> = 0
Enter number of DGR for unit 3?> = 0
    
```

Unit 4

```

Enter unit 4 sequence unit name?> = a350
Enter unit 4 console enable index?> = 0
Read/Write unit 4 from other sequence units[Y/N]?> = y
Enter unit 4 UPF name?> = short0
Enter number of DEV for unit 4?> = 3
    Enter EPN of first DEV?> = 350alarm
Enter number of AIN for unit 4?> = 17
    Enter EPN of first AIN?> = ii-350-1
Enter number of DIN for unit 4?> = 5
    Enter EPN of first DIN?> = lsh-350-2
Enter number of DOT for unit 4?> = 0
Enter number of DGR for unit 4?> = 0
    
```

Unit 5

```

Enter unit 5 sequence unit name?> = feed
Enter unit 5 console enable index?> = 0
Read/Write unit 5 from other sequence units[Y/N]?> = y
Enter unit 5 UPF name?> = prcess
Enter number of DEV for unit 5?> = 10
    Enter EPN of first DEV?> = paw102alm
Enter number of AIN for unit 5?> = 15
    Enter EPN of first AIN?> = ii-aw-102
Enter number of DIN for unit 5?> = 10
    Enter EPN of first DIN?> = ys-aw102
Enter number of DOT for unit 5?> = 0
Enter number of DGR for unit 5?> = 0
    
```

Unit 6

```

Enter unit 6 sequence unit name?> = cal
Enter unit 6 console enable index?> = 0
Read/Write unit 6 from other sequence units[Y/N]?> = y
Enter unit 6 UPF name?> = prcess
Enter number of DEV for unit 6?> = 10
    Enter EPN of first DEV?> = cal-alarm
Enter number of AIN for unit 6?> = 35
    Enter EPN of first AIN?> = pi-cal-11
Enter number of DIN for unit 6?> = 10
    Enter EPN of first DIN?> = psh-cal11
Enter number of DOT for unit 6?> = 0
Enter number of DGR for unit 6?> = 0
    
```

Unit 7

```

Enter unit 7 sequence unit name?> = pb1
Enter unit 7 console enable index?> = 0
Read/Write unit 7 from other sequence units[Y/N]?> = y
Enter unit 7 UPF name?> = prcess
Enter number of DEV for unit 7?> = 15
    Enter EPN of first DEV?> = pb1-alarm
Enter number of AIN for unit 7?> = 20
    Enter EPN of first AIN?> = ky-pb1-1
Enter number of DIN for unit 7?> = 15
    Enter EPN of first DIN?> = ys-pc-105
Enter number of DOT for unit 7?> = 0
Enter number of DGR for unit 7?> = 0
    
```

Unit 8

```

Enter unit 8 sequence unit name?> = pb2
Enter unit 8 console enable index?> = 0
Read/Write unit 8 from other sequence units[Y/N]?> = y
Enter unit 8 UPF name?> = prcess
Enter number of DEV for unit 8?> = 15
    Enter EPN of first DEV?> = pb2-alarm
Enter number of AIN for unit 8?> = 20
    Enter EPN of first AIN?> = hi-wvr-1
Enter number of DIN for unit 8?> = 20
    Enter EPN of first DIN?> = ys-pb2-1
Enter number of DOT for unit 8?> = 0
Enter number of DGR for unit 8?> = 0
    
```

Unit 9

```

Enter unit 9 sequence unit name?> = eal
Enter unit 9 console enable index?> = 0
Read/Write unit 9 from other sequence units[Y/N]?> = y
Enter unit 9 UPF name?> = prcess
Enter number of DEV for unit 9?> = 15
    Enter EPN of first DEV?> = eal-alarm
Enter number of AIN for unit 9?> = 30
    Enter EPN of first AIN?> = ti-eal-7
Enter number of DIN for unit 9?> = 5
    Enter EPN of first DIN?> = ps1-eal-1
Enter number of DOT for unit 9?> = 0
Enter number of DGR for unit 9?> = 0
    
```

Unit 10

```

Enter unit 10 sequence unit name?> = ecl
Enter unit 10 console enable index?> = 0
Read/Write unit 10 from other sequence units[Y/N]?> = y
Enter unit 10 UPF name?> = prcess
Enter number of DEV for unit 10?> = 10
    Enter EPN of first DEV?> = ecl-alarm
Enter number of AIN for unit 10?> = 30
    Enter EPN of first AIN?> = ti-cal-9
Enter number of DIN for unit 10?> = 10
    Enter EPN of first DIN?> = rsh-cal-1
Enter number of DOT for unit 10?> = 0
Enter number of DGR for unit 10?> = 0
    
```

Unit 11

```

Enter unit 11 sequence unit name?> = rwater
Enter unit 11 console enable index?> = 0
Read/Write unit 11 from other sequence units[Y/N]?> = y
Enter unit 11 UPF name?> = prcess
Enter number of DEV for unit 11?> = 10
    Enter EPN of first DEV?> = rw-alarm
Enter number of AIN for unit 11?> = 10
    Enter EPN of first AIN?> = pdi-rw-1
Enter number of DIN for unit 11?> = 10
    Enter EPN of first DIN?> = lds-bfp-1
Enter number of DOT for unit 11?> = 0
Enter number of DGR for unit 11?> = 0
    
```

Unit 12

```

Enter unit 12 sequence unit name?> = urw
Enter unit 12 console enable index?> = 0
Read/Write unit 12 from other sequence units[Y/N]?> = y
Enter unit 12 UPF name?> = prcess
Enter number of DEV for unit 12?> = 10
Enter EPN of first DEV?> = urw-alarm
Enter number of AIN for unit 12?> = 15
Enter EPN of first AIN?> = fic-ec1-1
Enter number of DIN for unit 12?> = 10
Enter EPN of first DIN?> = fsl-rc2-1
Enter number of DOT for unit 12?> = 0
Enter number of DGR for unit 12?> = 0
    
```

Unit 13

```

Enter unit 13 sequence unit name?> = vvent
Enter unit 13 console enable index?> = 0
Read/Write unit 13 from other sequence units[Y/N]?> = y
Enter unit 13 UPF name?> = prcess
Enter number of DEV for unit 13?> = 10
Enter EPN of first DEV?> = vv-alarm
Enter number of AIN for unit 13?> = 20
Enter EPN of first AIN?> = pdi-fc5-2
Enter number of DIN for unit 13?> = 20
Enter EPN of first DIN?> = pdsh-fc51
Enter number of DOT for unit 13?> = 0
Enter number of DGR for unit 13?> = 0
    
```

Unit 14

```

Enter unit 14 sequence unit name?> = pc
Enter unit 14 console enable index?> = 0
Read/Write unit 14 from other sequence units[Y/N]?> = y
Enter unit 14 UPF name?> = prcess
Enter number of DEV for unit 14?> = 10
Enter EPN of first DEV?> = pc-alarm
Enter number of AIN for unit 14?> = 15
Enter EPN of first AIN?> = ii-a-c100
Enter number of DIN for unit 14?> = 10
Enter EPN of first DIN?> = ys-a-c100
Enter number of DOT for unit 14?> = 0
Enter number of DGR for unit 14?> = 0
    
```

Unit 15

```

Enter unit 15 sequence unit name?> = ix
Enter unit 15 console enable index?> = 0
Read/Write unit 15 from other sequence units[Y/N]?> = y
Enter unit 15 UPF name?> = prcess
Enter number of DEV for unit 15?> = 10
Enter EPN of first DEV?> = ix-alarm
Enter number of AIN for unit 15?> = 25
Enter EPN of first AIN?> = pi-idx-13
Enter number of DIN for unit 15?> = 20
Enter EPN of first DIN?> = zs-idx-2
Enter number of DOT for unit 15?> = 0
Enter number of DGR for unit 15?> = 0
    
```

Unit 16

```

Enter unit 16 sequence unit name?> = sc
Enter unit 16 console enable index?> = 0
Read/Write unit 16 from other sequence units[Y/N]?> = y
Enter unit 16 UPF name?> = prcess
Enter number of DEV for unit 16?> = 10
Enter EPN of first DEV?> = sc-alarm
Enter number of AIN for unit 16?> = 15
Enter EPN of first AIN?> = fi-eal-3
Enter number of DIN for unit 16?> = 20
Enter EPN of first DIN?> = rsh-eal-1
Enter number of DOT for unit 16?> = 0
Enter number of DGR for unit 16?> = 0
    
```

Unit 17

```

Enter unit 17 sequence unit name?> = basins
Enter unit 17 console enable index?> = 0
Read/Write unit 17 from other sequence units[Y/N]?> = y
Enter unit 17 UPF name?> = prcess
Enter number of DEV for unit 17?> = 10
Enter EPN of first DEV?> = basin-alm
Enter number of AIN for unit 17?> = 10
Enter EPN of first AIN?> = ii-sumpp1
Enter number of DIN for unit 17?> = 20
Enter EPN of first DIN?> = ys-scp3il
Enter number of DOT for unit 17?> = 0
Enter number of DGR for unit 17?> = 0
    
```

Unit 18

```

Enter unit 18 sequence unit name?> = util
Enter unit 18 console enable index?> = 0
Read/Write unit 18 from other sequence units[Y/N]?> = y
Enter unit 18 UPF name?> = prcess
Enter number of DEV for unit 18?> = 10
    Enter EPN of first DEV?> = utilities
Enter number of AIN for unit 18?> = 20
    Enter EPN of first AIN?> = pi-ia-1
Enter number of DIN for unit 18?> = 20
    Enter EPN of first DIN?> = pslcpel/2
Enter number of DOT for unit 18?> = 0
Enter number of DGR for unit 18?> = 0
    
```

Unit 19

```

Enter unit 19 sequence unit name?> = rooms
Enter unit 19 console enable index?> = 0
Read/Write unit 19 from other sequence units[Y/N]?> = y
Enter unit 19 UPF name?> = prcess
Enter number of DEV for unit 19?> = 10
    Enter EPN of first DEV?> = room-alm
Enter number of AIN for unit 19?> = 10
    Enter EPN of first AIN?> = ri-kl-1-1
Enter number of DIN for unit 19?> = 10
    Enter EPN of first DIN?> = simdin1
Enter number of DOT for unit 19?> = 0
Enter number of DGR for unit 19?> = 0
    
```

Unit 20

```

Enter unit 20 sequence unit name?> = amu
Enter unit 20 console enable index?> = 0
Read/Write unit 20 from other sequence units[Y/N]?> = y
Enter unit 20 UPF name?> = prcess
Enter number of DEV for unit 20?> = 10
    Enter EPN of first DEV?> = amu-alarm
Enter number of AIN for unit 20?> = 20
    Enter EPN of first AIN?> = ti-e101-2
Enter number of DIN for unit 20?> = 10
    Enter EPN of first DIN?> = ys-a-e101
Enter number of DOT for unit 20?> = 0
Enter number of DGR for unit 20?> = 0
    
```

Unit 21

```

Enter unit 21 sequence unit name?> = 1erf
Enter unit 21 console enable index?> = 0
Read/Write unit 21 from other sequence units[Y/N]?> = y
Enter unit 21 UPF name?> = prcess
Enter number of DEV for unit 21?> = 10
    Enter EPN of first DEV?> = 1erf-alm
Enter number of AIN for unit 21?> = 20
    Enter EPN of first AIN?> = 1di-a1
Enter number of DIN for unit 21?> = 20
    Enter EPN of first DIN?> = ys-p-43
Enter number of DOT for unit 21?> = 0
Enter number of DGR for unit 21?> = 0
    
```

Unit 22

```

Enter unit 22 sequence unit name?> = evap
Enter unit 22 console enable index?> = 0
Read/Write unit 22 from other sequence units[Y/N]?> = y
Enter unit 22 UPF name?> = ovrview
Enter number of DEV for unit 22?> = 80
    Enter EPN of first DEV?> = paw102alm
Enter number of AIN for unit 22?> = 155
    Enter EPN of first AIN?> = ii-aw-102
Enter number of DIN for unit 22?> = 110
    Enter EPN of first DIN?> = ys-aw102
Enter number of DOT for unit 22?> = 0
Enter number of DGR for unit 22?> = 0
    
```

Unit 23

```

Enter unit 23 sequence unit name?> = vapor
Enter unit 23 console enable index?> = 0
Read/Write unit 23 from other sequence units[Y/N]?> = y
Enter unit 23 UPF name?> = ovrview
Enter number of DEV for unit 23?> = 30
    Enter EPN of first DEV?> = ecl-alarm
Enter number of AIN for unit 23?> = 90
    Enter EPN of first AIN?> = ti-cal-9
Enter number of DIN for unit 23?> = 60
    Enter EPN of first DIN?> = rsh-cal-1
Enter number of DOT for unit 23?> = 0
Enter number of DGR for unit 23?> = 0
    
```


Unit 24

Enter unit 24 sequence unit name?> = cond
 Enter unit 24 console enable index?> = 0
 Read/Write unit 24 from other sequence units[Y/N]?> = y
 Enter unit 24 UPF name?> = ovrview
 Enter number of DEV for unit 24?> = 85
     Enter EPN of first DEV?> = pc-alarm
 Enter number of AIN for unit 24?> = 135
     Enter EPN of first AIN?> = ii-a-c100
 Enter number of DIN for unit 24?> = 100
     Enter EPN of first DIN?> = ys-a-c100
 Enter number of DOT for unit 24?> = 0
 Enter number of DGR for unit 24?> = 0

Unit 25

Enter unit 25 sequence unit name?> = blank0
 Enter unit 25 console enable index?> = 0
 Read/Write unit 25 from other sequence units[Y/N]?> = y
 Enter unit 25 UPF name?> = blankk
 Enter number of DEV for unit 25?> = 0
 Enter number of AIN for unit 25?> = 0
 Enter number of DIN for unit 25?> = 0
 Enter number of DOT for unit 25?> = 0
 Enter number of DGR for unit 25?> = 0

Build PCMOA.SDB? [Y/N]> = y

Build Completed -- Database requires 3295 pages of memory

Total Errors = 0

WHC-SD-534-CSWL-001 REV 3

PCM1ASDB

PAGE 70

WHC-SD-534-CSWL-001 REV 3

PCM name for which to create the SDB?> = pcm1a
Number of sequence units for PCM?> = 7

Unit 0

Enter unit 0 sequence unit name?> = whole1
Enter unit 0 console enable index?> = 0
Read/Write unit 0 from other sequence units[Y/N]?> = y
Enter unit 0 UPF name?> = totall
Enter number of DEV for unit 0?> = 90
Enter EPN of first DEV?> = p-an-101
Enter number of AIN for unit 0?> = 130
Enter EPN of first AIN?> = ii-an1011
Enter number of DIN for unit 0?> = 280
Enter EPN of first DIN?> = ys-an-101
Enter number of DOT for unit 0?> = 0
Enter number of DGR for unit 0?> = 0

Unit 1

Enter unit 1 sequence unit name?> = anfarm
Enter unit 1 console enable index?> = 0
Read/Write unit 1 from other sequence units[Y/N]?> = y
Enter unit 1 UPF name?> = tnkfrm
Enter number of DEV for unit 1?> = 15
Enter EPN of first DEV?> = p-an-101
Enter number of AIN for unit 1?> = 20
Enter EPN of first AIN?> = ii-an1011
Enter number of DIN for unit 1?> = 60
Enter EPN of first DIN?> = ys-an-101
Enter number of DOT for unit 1?> = 0
Enter number of DGR for unit 1?> = 0

Unit 2

Enter unit 2 sequence unit name?> = apfarm
Enter unit 2 console enable index?> = 0
Read/Write unit 2 from other sequence units[Y/N]?> = y
Enter unit 2 UPF name?> = tnkfrm
Enter number of DEV for unit 2?> = 20
Enter EPN of first DEV?> = p-ap-101
Enter number of AIN for unit 2?> = 20
Enter EPN of first AIN?> = ii-ap1011
Enter number of DIN for unit 2?> = 90
Enter EPN of first DIN?> = ys-ap-101
Enter number of DOT for unit 2?> = 0
Enter number of DGR for unit 2?> = 0

Unit 3

```

Enter unit 3 sequence unit name?> = awfarm
Enter unit 3 console enable index?> = 0
Read/Write unit 3 from other sequence units[Y/N]?> = y
Enter unit 3 UPF name?> = tnkfrm
Enter number of DEV for unit 3?> = 20
    Enter EPN of first DEV?> = p-aw-101
Enter number of AIN for unit 3?> = 20
    Enter EPN of first AIN?> = ii-aw1011
Enter number of DIN for unit 3?> = 60
    Enter EPN of first DIN?> = ys-aw-101
Enter number of DOT for unit 3?> = 0
Enter number of DGR for unit 3?> = 0
    
```

Unit 4

```

Enter unit 4 sequence unit name?> = xyzfrm
Enter unit 4 console enable index?> = 0
Read/Write unit 4 from other sequence units[Y/N]?> = y
Enter unit 4 UPF name?> = tnkfrm
Enter number of DEV for unit 4?> = 15
    Enter EPN of first DEV?> = p-ay-101
Enter number of AIN for unit 4?> = 60
    Enter EPN of first AIN?> = ii-ay-101
Enter number of DIN for unit 4?> = 60
    Enter EPN of first DIN?> = psh-s1100
Enter number of DOT for unit 4?> = 0
Enter number of DGR for unit 4?> = 0
    
```

Unit 5

```

Enter unit 5 sequence unit name?> = bldng
Enter unit 5 console enable index?> = 0
Read/Write unit 5 from other sequence units[Y/N]?> = y
Enter unit 5 UPF name?> = tnkfrm
Enter number of DEV for unit 5?> = 5
    Enter EPN of first DEV?> = building
Enter number of AIN for unit 5?> = 20
    Enter EPN of first AIN?> = ri-ar-amu
Enter number of DIN for unit 5?> = 30
    Enter EPN of first DIN?> = rsh-aramu
Enter number of DOT for unit 5?> = 0
Enter number of DGR for unit 5?> = 0
    
```

WHC-SD-534-CSWL-001 REV 3

Unit 6

Enter unit 6 sequence unit name?> = blank1
Enter unit 6 console enable index?> = 0
Read/Write unit 6 from other sequence units[Y/N]?> = y
Enter unit 6 UPF name?> = blankk
Enter number of DEV for unit 6?> = 5
    Enter EPN of first DEV?> = gbit1-0/1
Enter number of AIN for unit 6?> = 0
Enter number of DIN for unit 6?> = 0
Enter number of DOT for unit 6?> = 0
Enter number of DGR for unit 6?> = 0

Build PCM1A.SDB? [Y/N]> = y

Build Completed -- Database requires 1791 pages of memory

Total Errors = 0

APPENDIX C - RECIPE PROTOTYPE FILES (RPF)

ALLO

RMS Version 6.3-2 05-Dec-94 09:31:35

RMS?> = rpf de all0

RPF?> = rpf de all0

Invalid Command

Open error file SVO:[300,1]U.UPF ,error = -1

RPF?> = cr all0,*

MRPF?> = add 32 fg

Enter new 6 char. name of fg 0?> = flag0

Enter new 18 char. descriptor of fg 0?> = flag 0

Enter new 6 char. name of fg 1?> = flag1

Enter new 18 char. descriptor of fg 1?> = flag 1

Enter new 6 char. name of fg 2?> = flag2

Enter new 18 char. descriptor of fg 2?> = flag 2

Enter new 6 char. name of fg 3?> = flag3

Enter new 18 char. descriptor of fg 3?> = flag 3

Enter new 6 char. name of fg 4?> = flag4

Enter new 18 char. descriptor of fg 4?> = flag 4

Enter new 6 char. name of fg 5?> = flag5

Enter new 18 char. descriptor of fg 5?> = flag 5

Enter new 6 char. name of fg 6?> = flag6

Enter new 18 char. descriptor of fg 6?> = flag 6

Enter new 6 char. name of fg 7?> = flag7

Enter new 18 char. descriptor of fg 7?> = flag 7

Enter new 6 char. name of fg 8?> = flag8

Enter new 18 char. descriptor of fg 8?> = flag 8

Enter new 6 char. name of fg 9?> = flag9

Enter new 18 char. descriptor of fg 9?> = flag 9

Enter new 6 char. name of fg 10?> = flag10

Enter new 18 char. descriptor of fg 10?> = flag 10

Enter new 6 char. name of fg 11?> = flag11

Enter new 18 char. descriptor of fg 11?> = flag 11

Enter new 6 char. name of fg 12?> = flag12

Enter new 18 char. descriptor of fg 12?> = flag 12

Enter new 6 char. name of fg 13?> = flag13

Enter new 18 char. descriptor of fg 13?> = flag 13

Enter new 6 char. name of fg 14?> = flag14
Enter new 18 char. descriptor of fg 14?> = flag 14

Enter new 6 char. name of fg 15?> = flag15
Enter new 18 char. descriptor of fg 15?> = flag 15

Enter new 6 char. name of fg 16?> = flag16
Enter new 18 char. descriptor of fg 16?> = flag 16

Enter new 6 char. name of fg 17?> = flag17
Enter new 18 char. descriptor of fg 17?> = flag 17

Enter new 6 char. name of fg 18?> = flag18
Enter new 18 char. descriptor of fg 18?> = flag 18

Enter new 6 char. name of fg 19?> = flag19
Enter new 18 char. descriptor of fg 19?> = flag 19

Enter new 6 char. name of fg 20?> = flag20
Enter new 18 char. descriptor of fg 20?> = flag 20

Enter new 6 char. name of fg 21?> = flag21
Enter new 18 char. descriptor of fg 21?> = flag 21

Enter new 6 char. name of fg 22?> = flag22
Enter new 18 char. descriptor of fg 22?> = flag 22

Enter new 6 char. name of fg 23?> = flag23
Enter new 18 char. descriptor of fg 23?> = flag 23

Enter new 6 char. name of fg 24?> = flag24
Enter new 18 char. descriptor of fg 24?> = flag 24

Enter new 6 char. name of fg 25?> = flag25
Enter new 18 char. descriptor of fg 25?> = flag 25

Enter new 6 char. name of fg 26?> = flag26
Enter new 18 char. descriptor of fg 26?> = flag 26

Enter new 6 char. name of fg 27?> = flag27
Enter new 18 char. descriptor of fg 27?> = flag 27

Enter new 6 char. name of fg 28?> = flag28
Enter new 18 char. descriptor of fg 28?> = flag 28

Enter new 6 char. name of fg 29?> = flag29
Enter new 18 char. descriptor of fg 29?> = flag 29

Enter new 6 char. name of fg 30?> = flag30
Enter new 18 char. descriptor of fg 30?> = flag 30

Enter new 6 char. name of fg 31?> = flag31
Enter new 18 char. descriptor of fg 31?> = 31
MRPF?> = add 40 fl

Enter new 6 char. name of fl 0?> = val0
Enter new 18 char. descriptor of fl 0?> = amps
Enter low limit for fl 0?> = 0
Enter high limit for fl 0?> = 13.6

Enter new 6 char. name of fl 1?> = val1
Enter new 18 char. descriptor of fl 1?> = pmp amps
Enter low limit for fl 1?> = 0
Enter high limit for fl 1?> = 13.6

Enter new 6 char. name of fl 2?> = val2
Enter new 18 char. descriptor of fl 2?> = pmp amps
Enter low limit for fl 2?> = 0
Enter high limit for fl 2?> = 13.6

Enter new 6 char. name of fl 3?> = val3
Enter new 18 char. descriptor of fl 3?> = agt amps
Enter low limit for fl 3?> = 0
Enter high limit for fl 3?> = 14.0

Enter new 6 char. name of fl 4?> = val4
Enter new 18 char. descriptor of fl 4?> = temp
Enter low limit for fl 4?> = 0
Enter high limit for fl 4?> = 200.0

Enter new 6 char. name of fl 5?> = val5
Enter new 18 char. descriptor of fl 5?> = flow
Enter low limit for fl 5?> = 375
Enter high limit for fl 5?> = 500

Enter new 6 char. name of fl 6?> = val6
Enter new 18 char. descriptor of fl 6?> = pmp amps
Enter low limit for fl 6?> = 0.0
Enter high limit for fl 6?> = 50.00

Enter new 6 char. name of fl 7?> = val7
Enter new 18 char. descriptor of fl 7?> = pmp amps
Enter low limit for fl 7?> = 0
Enter high limit for fl 7?> = 34.00

Enter new 6 char. name of fl 8?> = val8
 Enter new 18 char. descriptor of fl 8?> = dlta spg
 Enter low limit for fl 8?> = -0.1
 Enter high limit for fl 8?> = 00.1

Enter new 6 char. name of fl 9?> = val9
 Enter new 18 char. descriptor of fl 9?> = rw dp
 Enter low limit for fl 9?> = 0
 Enter high limit for fl 9?> = 30.00

Enter new 6 char. name of fl 10?> = val10
 Enter new 18 char. descriptor of fl 10?> = rw dp
 Enter low limit for fl 10?> = 0
 Enter high limit for fl 10?> = 30.00

Enter new 6 char. name of fl 11?> = val11
 Enter new 18 char. descriptor of fl 11?> = rw dp
 Enter low limit for fl 11?> = 0
 Enter high limit for fl 11?> = 30.00

Enter new 6 char. name of fl 12?> = val12
 Enter new 18 char. descriptor of fl 12?> = slry spd
 Enter low limit for fl 12?> = 0
 Enter high limit for fl 12?> = 100.0

Enter new 6 char. name of fl 13?> = val13
 Enter new 18 char. descriptor of fl 13?> = 90# stm
 Enter low limit for fl 13?> = 0
 Enter high limit for fl 13?> = 100.00

Enter new 6 char. name of fl 14?> = val14
 Enter new 18 char. descriptor of fl 14?> = stm flw
 Enter low limit for fl 14?> = 0.0
 Enter high limit for fl 14?> = 30000.0

Enter new 6 char. name of fl 15?> = val15
 Enter new 18 char. descriptor of fl 15?> = stm inlet
 Enter low limit for fl 15?> = 15.00
 Enter high limit for fl 15?> = 40.00

Enter new 6 char. name of fl 16?> = val16
 Enter new 18 char. descriptor of fl 16?> = sc temp
 Enter low limit for fl 16?> = 100.0
 Enter high limit for fl 16?> = 200.0

Enter new 6 char. name of fl 17?> = val17
 Enter new 18 char. descriptor of fl 17?> = rw inlet
 Enter low limit for fl 17?> = 0.0
 Enter high limit for fl 17?> = 100.00

Enter new 6 char. name of fl 18?> = val18
 Enter new 18 char. descriptor of fl 18?> = urw flow
 Enter low limit for fl 18?> = 0.0
 Enter high limit for fl 18?> = 150.0

Enter new 6 char. name of fl 19?> = val19
 Enter new 18 char. descriptor of fl 19?> = filter dp
 Enter low limit for fl 19?> = 0.0
 Enter high limit for fl 19?> = 30.0

Enter new 6 char. name of fl 20?> = val20
 Enter new 18 char. descriptor of fl 20?> = filter dp
 Enter low limit for fl 20?> = 0.0
 Enter high limit for fl 20?> = 30.0

Enter new 6 char. name of fl 21?> = val21
 Enter new 18 char. descriptor of fl 21?> = urw flow
 Enter low limit for fl 21?> = 0.0
 Enter high limit for fl 21?> = 4000.0

Enter new 6 char. name of fl 22?> = val22
 Enter new 18 char. descriptor of fl 22?> = vv amps
 Enter low limit for fl 22?> = 0
 Enter high limit for fl 22?> = 4.8

Enter new 6 char. name of fl 23?> = val23
 Enter new 18 char. descriptor of fl 23?> = vv flow
 Enter low limit for fl 23?> = 0.0
 Enter high limit for fl 23?> = 1000.0

Enter new 6 char. name of fl 24?> = val24
 Enter new 18 char. descriptor of fl 24?> = pmp amps
 Enter low limit for fl 24?> = 0.0
 Enter high limit for fl 24?> = 15.20

Enter new 6 char. name of fl 25?> = val25
 Enter new 18 char. descriptor of fl 25?> = pc amps
 Enter low limit for fl 25?> = 0.0
 Enter high limit for fl 25?> = 18.33

Enter new 6 char. name of fl 26?> = val26
 Enter new 18 char. descriptor of fl 26?> = lo iface
 Enter low limit for fl 26?> = 0.8
 Enter high limit for fl 26?> = 1.00

Enter new 6 char. name of fl 27?> = val27
 Enter new 18 char. descriptor of fl 27?> = pc amps
 Enter low limit for fl 27?> = 0.0
 Enter high limit for fl 27?> = 18.67

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fl 28?> = val28
Enter new 18 char. descriptor of fl 28?> = float value 28
Enter low limit for fl 28?> = -9999
Enter high limit for fl 28?> = 99999

Enter new 6 char. name of fl 29?> = val29
Enter new 18 char. descriptor of fl 29?> = float value 29
Enter low limit for fl 29?> = -9999
Enter high limit for fl 29?> = 99999

Enter new 6 char. name of fl 30?> = val30
Enter new 18 char. descriptor of fl 30?> = float value 30
Enter low limit for fl 30?> = -9999
Enter high limit for fl 30?> = 99999

Enter new 6 char. name of fl 31?> = val31
Enter new 18 char. descriptor of fl 31?> = float value 31
Enter low limit for fl 31?> = -9999
Enter high limit for fl 31?> = 99999

Enter new 6 char. name of fl 32?> = val32
Enter new 18 char. descriptor of fl 32?> = float value 32
Enter low limit for fl 32?> = -9999
Enter high limit for fl 32?> = 99999

Enter new 6 char. name of fl 33?> = val33
Enter new 18 char. descriptor of fl 33?> = float value 33
Enter low limit for fl 33?> = -9999
Enter high limit for fl 33?> = 99999

Enter new 6 char. name of fl 34?> = val34
Enter new 18 char. descriptor of fl 34?> = float value 34
Enter low limit for fl 34?> = -9999
Enter high limit for fl 34?> = 99999

Enter new 6 char. name of fl 35?> = val35
Enter new 18 char. descriptor of fl 35?> = float value 35
Enter low limit for fl 35?> = -9999
Enter high limit for fl 35?> = 99999

Enter new 6 char. name of fl 36?> = val36
Enter new 18 char. descriptor of fl 36?> = float value 36
Enter low limit for fl 36?> = -9999
Enter high limit for fl 36?> = 99999

Enter new 6 char. name of fl 37?> = val37
Enter new 18 char. descriptor of fl 37?> = float value 37
Enter low limit for fl 37?> = -9999
Enter high limit for fl 37?> = 99999

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fl 38?> = val38
Enter new 18 char. descriptor of fl 38?> = float value 38
Enter low limit for fl 38?> = -9999
Enter high limit for fl 38?> = 99999

Enter new 6 char. name of fl 39?> = val39
Enter new 18 char. descriptor of fl 39?> = float value 39
Enter low limit for fl 39?> = -9999
Enter high limit for fl 39?> = 99999
MRPF?> = add 50 in

Enter new 6 char. name of in 0?> = in0
Enter new 18 char. descriptor of in 0?> = integer number 0
Enter low limit for in 0?> = 0
Enter high limit for in 0?> = 100

Enter new 6 char. name of in 1?> = in1
Enter new 18 char. descriptor of in 1?> = integer number 1
Enter low limit for in 1?> = 0
Enter high limit for in 1?> = 100

Enter new 6 char. name of in 2?> = in2
Enter new 18 char. descriptor of in 2?> = integer number 2
Enter low limit for in 2?> = 0
Enter high limit for in 2?> = 100

Enter new 6 char. name of in 3?> = in3
Enter new 18 char. descriptor of in 3?> = integer number 3
Enter low limit for in 3?> = 0
Enter high limit for in 3?> = 100

Enter new 6 char. name of in 4?> = in4
Enter new 18 char. descriptor of in 4?> = integer number 4
Enter low limit for in 4?> = 0
Enter high limit for in 4?> = 100

Enter new 6 char. name of in 5?> = in5
Enter new 18 char. descriptor of in 5?> = integer number 5
Enter low limit for in 5?> = 0
Enter high limit for in 5?> = 100

Enter new 6 char. name of in 6?> = in6
Enter new 18 char. descriptor of in 6?> = integer number 6
Enter low limit for in 6?> = 0
Enter high limit for in 6?> = 100

Enter new 6 char. name of in 7?> = in7
Enter new 18 char. descriptor of in 7?> = integer number 7
Enter low limit for in 7?> = 0
Enter high limit for in 7?> = 100

Enter new 6 char. name of in 8?> = in8
 Enter new 18 char. descriptor of in 8?> = integer number 8
 Enter low limit for in 8?> = 0
 Enter high limit for in 8?> = 7

Enter new 6 char. name of in 9?> = in9
 Enter new 18 char. descriptor of in 9?> = integer number 9
 Enter low limit for in 9?> = 0
 Enter high limit for in 9?> = 7

Enter new 6 char. name of in 10?> = in10
 Enter new 18 char. descriptor of in 10?> = integer number 10
 Enter low limit for in 10?> = 0
 Enter high limit for in 10?> = 7

Enter new 6 char. name of in 11?> = in11
 Enter new 18 char. descriptor of in 11?> = integer number 11
 Enter low limit for in 11?> = 0
 Enter high limit for in 11?> = 7

Enter new 6 char. name of in 12?> = in12
 Enter new 18 char. descriptor of in 12?> = integer number 12
 Enter low limit for in 12?> = 0
 Enter high limit for in 12?> = 7

Enter new 6 char. name of in 13?> = in13
 Enter new 18 char. descriptor of in 13?> = integer number 13
 Enter low limit for in 13?> = 0
 Enter high limit for in 13?> = 7

Enter new 6 char. name of in 14?> = in14
 Enter new 18 char. descriptor of in 14?> = integer number 14
 Enter low limit for in 14?> = 0
 Enter high limit for in 14?> = 7

Enter new 6 char. name of in 15?> = in15
 Enter new 18 char. descriptor of in 15?> = integer number 15
 Enter low limit for in 15?> = 0
 Enter high limit for in 15?> = 7

Enter new 6 char. name of in 16?> = in16
 Enter new 18 char. descriptor of in 16?> = integer number 16
 Enter low limit for in 16?> = 0
 Enter high limit for in 16?> = 7

Enter new 6 char. name of in 17?> = in17
 Enter new 18 char. descriptor of in 17?> = integer number 17
 Enter low limit for in 17?> = 0
 Enter high limit for in 17?> = 7

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 18?> = in18
Enter new 18 char. descriptor of in 18?> = integer number 18
Enter low limit for in 18?> = 0
Enter high limit for in 18?> = 7

Enter new 6 char. name of in 19?> = in19
Enter new 18 char. descriptor of in 19?> = integer number 19
Enter low limit for in 19?> = 0
Enter high limit for in 19?> = 7

Enter new 6 char. name of in 20?> = in20
Enter new 18 char. descriptor of in 20?> = integer number 20
Enter low limit for in 20?> = 0
Enter high limit for in 20?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 21?> = in21
Enter new 18 char. descriptor of in 21?> = integer number 21
Enter low limit for in 21?> = 0
Enter high limit for in 21?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 22?> = in22
Enter new 18 char. descriptor of in 22?> = integer number 22
Enter low limit for in 22?> = 0
Enter high limit for in 22?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 23?> = in23
Enter new 18 char. descriptor of in 23?> = integer number 23
Enter low limit for in 23?> = 0
Enter high limit for in 23?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 24?> = in24
Enter new 18 char. descriptor of in 24?> = integer number 24
Enter low limit for in 24?> = 0
Enter high limit for in 24?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 25?> = in25
Enter new 18 char. descriptor of in 25?> = integer number 25
Enter low limit for in 25?> = 0
Enter high limit for in 25?> = 100000
High limit is less than low - setting high to 32767

WMC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 26?> = in26
Enter new 18 char. descriptor of in 26?> = integer number 26
Enter low limit for in 26?> = 0
Enter high limit for in 26?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 27?> = in27
Enter new 18 char. descriptor of in 27?> = integer number 27
Enter low limit for in 27?> = 0
Enter high limit for in 27?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 28?> = in28
Enter new 18 char. descriptor of in 28?> = integer number 28
Enter low limit for in 28?> = 0
Enter high limit for in 28?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 29?> = in29
Enter new 18 char. descriptor of in 29?> = integer number 29
Enter low limit for in 29?> = 0
Enter high limit for in 29?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 30?> = in30
Enter new 18 char. descriptor of in 30?> = integer number 30
Enter low limit for in 30?> = 0
Enter high limit for in 30?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 31?> = in31
Enter new 18 char. descriptor of in 31?> = integer number 31
Enter low limit for in 31?> = 0
Enter high limit for in 31?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 32?> = in32
Enter new 18 char. descriptor of in 32?> = integer number 32
Enter low limit for in 32?> = 0
Enter high limit for in 32?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 33?> = in33
Enter new 18 char. descriptor of in 33?> = integer number 33
Enter low limit for in 33?> = 0
Enter high limit for in 33?> = 100000
High limit is less than low - setting high to 32767

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 34?> = in34
Enter new 18 char. descriptor of in 34?> = integer number 34
Enter low limit for in 34?> = 0
Enter high limit for in 34?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 35?> = in35
Enter new 18 char. descriptor of in 35?> = integer number 35
Enter low limit for in 35?> = 0
Enter high limit for in 35?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 36?> = in36
Enter new 18 char. descriptor of in 36?> = integer number 36
Enter low limit for in 36?> = 0
Enter high limit for in 36?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 37?> = in37
Enter new 18 char. descriptor of in 37?> = integer number 37
Enter low limit for in 37?> = 0
Enter high limit for in 37?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 38?> = in38
Enter new 18 char. descriptor of in 38?> = integer number 38
Enter low limit for in 38?> = 0
Enter high limit for in 38?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 39?> = in39
Enter new 18 char. descriptor of in 39?> = integer number 39
Enter low limit for in 39?> = 0
Enter high limit for in 39?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 40?> = imisc1
Enter new 18 char. descriptor of in 40?> = misc1 val1
Enter low limit for in 40?> = 0
Enter high limit for in 40?> = 128

Enter new 6 char. name of in 41?> = imisc2
Enter new 18 char. descriptor of in 41?> = misc2 val2
Enter low limit for in 41?> = 0
Enter high limit for in 41?> = 128

Enter new 6 char. name of in 42?> = adi
Enter new 18 char. descriptor of in 42?> = alm des index
Enter low limit for in 42?> = 0
Enter high limit for in 42?> = 80

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 43?> = lmpvar
 Enter new 18 char. descriptor of in 43?> = lamp var
 Enter low limit for in 43?> = 0
 Enter high limit for in 43?> = 4

Enter new 6 char. name of in 44?> = led0
 Enter new 18 char. descriptor of in 44?> = led 0
 Enter low limit for in 44?> = 0
 Enter high limit for in 44?> = 63

Enter new 6 char. name of in 45?> = led1
 Enter new 18 char. descriptor of in 45?> = led 1
 Enter low limit for in 45?> = 0
 Enter high limit for in 45?> = 63

Enter new 6 char. name of in 46?> = led2
 Enter new 18 char. descriptor of in 46?> = led 2
 Enter low limit for in 46?> = 0
 Enter high limit for in 46?> = 63

Enter new 6 char. name of in 47?> = ispar0
 Enter new 18 char. descriptor of in 47?> = spare int 0
 Enter low limit for in 47?> = 0
 Enter high limit for in 47?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 48?> = ispar1
 Enter new 18 char. descriptor of in 48?> = spare int 1
 Enter low limit for in 48?> = 0
 Enter high limit for in 48?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 49?> = ispar2
 Enter new 18 char. descriptor of in 49?> = spare int 2
 Enter low limit for in 49?> = 0
 Enter high limit for in 49?> = 37000
 High limit is less than low - setting high to 32767
 MRPF?> = add 5 st

Enter new 6 char. name of st 0?> = stmr0
 Enter new 18 char. descriptor of st 0?> = second timer 0
 Enter low limit for st 0?> = 0
 Enter high limit for st 0?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of st 1?> = stmr1
 Enter new 18 char. descriptor of st 1?> = second timer 1
 Enter low limit for st 1?> = 0
 Enter high limit for st 1?> = 300

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of st 2?> = stmr2
Enter new 18 char. descriptor of st 2?> = second timer 2
Enter low limit for st 2?> = 0
Enter high limit for st 2?> = 255

Enter new 6 char. name of st 3?> = stmr3
Enter new 18 char. descriptor of st 3?> = second timer 3
Enter low limit for st 3?> = 0
Enter high limit for st 3?> = 75

Enter new 6 char. name of st 4?> = stmr4
Enter new 18 char. descriptor of st 4?> = second timer 4
Enter low limit for st 4?> = 0
Enter high limit for st 4?> = 600
MRPF?> = add 5 sg

Enter new 6 char. name of sg 0?> = note0
Enter new 18 char. descriptor of sg 0?> = total note 0
Enter maximum size for sg 0?> = 30

Enter new 6 char. name of sg 1?> = note1
Enter new 18 char. descriptor of sg 1?> = total note 1
Enter maximum size for sg 1?> = 30

Enter new 6 char. name of sg 2?> = note2
Enter new 18 char. descriptor of sg 2?> = total note 2
Enter maximum size for sg 2?> = 30

Enter new 6 char. name of sg 3?> = note3
Enter new 18 char. descriptor of sg 3?> = total note 3
Enter maximum size for sg 3?> = 30

Enter new 6 char. name of sg 4?> = note4
Enter new 18 char. descriptor of sg 4?> = total note 4
Enter maximum size for sg 4?> = 30

MRPF?> =
RPF?>
end of file encountered
RMS?>
end of file encountered

242A

RMS Version 6.3-2 05-Dec-94 09:32:03
RMS?> = rpf de 242a
RPF?> = rpf de 242a
Invalid Command
Open error file SV0:[300,1]U.UPF ,error = -1
Open error file SV0:[300,1]U.UPF ,error = -1
RPF?> = cr 242a,*
MRPF?> = add 16 fg

Enter new 6 char. name of fg 0?> = flg0
Enter new 18 char. descriptor of fg 0?> = unit flag 0

Enter new 6 char. name of fg 1?> = flg1
Enter new 18 char. descriptor of fg 1?> = unit flag 1

Enter new 6 char. name of fg 2?> = flg2
Enter new 18 char. descriptor of fg 2?> = unit flag 2

Enter new 6 char. name of fg 3?> = flg3
Enter new 18 char. descriptor of fg 3?> = unit flag 3

Enter new 6 char. name of fg 4?> = flg4
Enter new 18 char. descriptor of fg 4?> = unit flag 4

Enter new 6 char. name of fg 5?> = flg5
Enter new 18 char. descriptor of fg 5?> = unit flag 5

Enter new 6 char. name of fg 6?> = flg6
Enter new 18 char. descriptor of fg 6?> = unit flag 6

Enter new 6 char. name of fg 7?> = flg7
Enter new 18 char. descriptor of fg 7?> = unit flag 7

Enter new 6 char. name of fg 8?> = flg8
Enter new 18 char. descriptor of fg 8?> = unit flag 8

Enter new 6 char. name of fg 9?> = flg9
Enter new 18 char. descriptor of fg 9?> = unit flag 9

Enter new 6 char. name of fg 10?> = flg10
Enter new 18 char. descriptor of fg 10?> = unit flag 10

Enter new 6 char. name of fg 11?> = flg11
Enter new 18 char. descriptor of fg 11?> = unit flag 11

Enter new 6 char. name of fg 12?> = flg12
Enter new 18 char. descriptor of fg 12?> = unit flag 12

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fg 13?> = flg13
Enter new 18 char. descriptor of fg 13?> = unit flag 13

Enter new 6 char. name of fg 14?> = flg14
Enter new 18 char. descriptor of fg 14?> = unit flag 14

Enter new 6 char. name of fg 15?> = flg15
Enter new 18 char. descriptor of fg 15?> = unit flag 15
MRPF?> = add 20 fl

Enter new 6 char. name of fl 0?> = wf0
Enter new 18 char. descriptor of fl 0?> = wf float 0
Enter low limit for fl 0?> = 0
Enter high limit for fl 0?> = 100

Enter new 6 char. name of fl 1?> = wf1
Enter new 18 char. descriptor of fl 1?> = wf float 1
Enter low limit for fl 1?> = 0
Enter high limit for fl 1?> = 100

Enter new 6 char. name of fl 2?> = wf2
Enter new 18 char. descriptor of fl 2?> = wf float 2
Enter low limit for fl 2?> = 0
Enter high limit for fl 2?> = 100

Enter new 6 char. name of fl 3?> = wf3
Enter new 18 char. descriptor of fl 3?> = wf float 3
Enter low limit for fl 3?> = 0
Enter high limit for fl 3?> = 100

Enter new 6 char. name of fl 4?> = wf4
Enter new 18 char. descriptor of fl 4?> = wf float 4
Enter low limit for fl 4?> = 0
Enter high limit for fl 4?> = 100

Enter new 6 char. name of fl 5?> = psi0
Enter new 18 char. descriptor of fl 5?> = pressure 0
Enter low limit for fl 5?> = 0
Enter high limit for fl 5?> = 100

Enter new 6 char. name of fl 6?> = psi1
Enter new 18 char. descriptor of fl 6?> = pressure 1
Enter low limit for fl 6?> = 0
Enter high limit for fl 6?> = 100

Enter new 6 char. name of fl 7?> = psi2
Enter new 18 char. descriptor of fl 7?> = pressure 2
Enter low limit for fl 7?> = 0
Enter high limit for fl 7?> = 100

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fl 8?> = psi3
Enter new 18 char. descriptor of fl 8?> = pressure 3
Enter low limit for fl 8?> = 0
Enter high limit for fl 8?> = 100

Enter new 6 char. name of fl 9?> = psi4
Enter new 18 char. descriptor of fl 9?> = pressure 4
Enter low limit for fl 9?> = 0
Enter high limit for fl 9?> = 100

Enter new 6 char. name of fl 10?> = di0
Enter new 18 char. descriptor of fl 10?> = density 0
Enter low limit for fl 10?> = 0
Enter high limit for fl 10?> = 100

Enter new 6 char. name of fl 11?> = di1
Enter new 18 char. descriptor of fl 11?> = density 1
Enter low limit for fl 11?> = 0
Enter high limit for fl 11?> = 100

Enter new 6 char. name of fl 12?> = di2
Enter new 18 char. descriptor of fl 12?> = density 2
Enter low limit for fl 12?> = 0
Enter high limit for fl 12?> = 100

Enter new 6 char. name of fl 13?> = di3
Enter new 18 char. descriptor of fl 13?> = density 3
Enter low limit for fl 13?> = 0
Enter high limit for fl 13?> = 100

Enter new 6 char. name of fl 14?> = di4
Enter new 18 char. descriptor of fl 14?> = density 4
Enter low limit for fl 14?> = 0
Enter high limit for fl 14?> = 100

Enter new 6 char. name of fl 15?> = v10
Enter new 18 char. descriptor of fl 15?> = float spare 0
Enter low limit for fl 15?> = 0
Enter high limit for fl 15?> = 100

Enter new 6 char. name of fl 16?> = v11
Enter new 18 char. descriptor of fl 16?> = float spare 1
Enter low limit for fl 16?> = 0
Enter high limit for fl 16?> = 100

Enter new 6 char. name of fl 17?> = v12
Enter new 18 char. descriptor of fl 17?> = float spare 2
Enter low limit for fl 17?> = 0
Enter high limit for fl 17?> = 100

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fl 18?> = v13
Enter new 18 char. descriptor of fl 18?> = float spare 3
Enter low limit for fl 18?> = 0
Enter high limit for fl 18?> = 100

Enter new 6 char. name of fl 19?> = v14
Enter new 18 char. descriptor of fl 19?> = float spare 4
Enter low limit for fl 19?> = 0
Enter high limit for fl 19?> = 100
MRPF?> = add 20 in

Enter new 6 char. name of in 0?> = in0
Enter new 18 char. descriptor of in 0?> = integer number 0
Enter low limit for in 0?> = 0
Enter high limit for in 0?> = 1000

Enter new 6 char. name of in 1?> = in1
Enter new 18 char. descriptor of in 1?> = integer number 1
Enter low limit for in 1?> = 0
Enter high limit for in 1?> = 1000

Enter new 6 char. name of in 2?> = in2
Enter new 18 char. descriptor of in 2?> = integer number 2
Enter low limit for in 2?> = 0
Enter high limit for in 2?> = 1000

Enter new 6 char. name of in 3?> = in3
Enter new 18 char. descriptor of in 3?> = integer number 3
Enter low limit for in 3?> = 0
Enter high limit for in 3?> = 1000

Enter new 6 char. name of in 4?> = in4
Enter new 18 char. descriptor of in 4?> = integer number 4
Enter low limit for in 4?> = 0
Enter high limit for in 4?> = 1000

Enter new 6 char. name of in 5?> = in5
Enter new 18 char. descriptor of in 5?> = integer number 5
Enter low limit for in 5?> = 0
Enter high limit for in 5?> = 100

Enter new 6 char. name of in 6?> = in6
Enter new 18 char. descriptor of in 6?> = integer number 6
Enter low limit for in 6?> = 0
Enter high limit for in 6?> = 100

Enter new 6 char. name of in 7?> = in7
Enter new 18 char. descriptor of in 7?> = integer number 7
Enter low limit for in 7?> = 0
Enter high limit for in 7?> = 100

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 8?> = in8
Enter new 18 char. descriptor of in 8?> = integer number 8
Enter low limit for in 8?> = 0
Enter high limit for in 8?> = 100

Enter new 6 char. name of in 9?> = in9
Enter new 18 char. descriptor of in 9?> = integer number 9
Enter low limit for in 9?> = 0
Enter high limit for in 9?> = 100

Enter new 6 char. name of in 10?> = in10
Enter new 18 char. descriptor of in 10?> = integer number 10
Enter low limit for in 10?> = 0
Enter high limit for in 10?> = 7

Enter new 6 char. name of in 11?> = in11
Enter new 18 char. descriptor of in 11?> = integer number 11
Enter low limit for in 11?> = 0
Enter high limit for in 11?> = 7

Enter new 6 char. name of in 12?> = in12
Enter new 18 char. descriptor of in 12?> = integer number 12
Enter low limit for in 12?> = 0
Enter high limit for in 12?> = 7

Enter new 6 char. name of in 13?> = in13
Enter new 18 char. descriptor of in 13?> = integer number 13
Enter low limit for in 13?> = 0
Enter high limit for in 13?> = 7

Enter new 6 char. name of in 14?> = in14
Enter new 18 char. descriptor of in 14?> = integer number 14
Enter low limit for in 14?> = 0
Enter high limit for in 14?> = 7

Enter new 6 char. name of in 15?> = in15
Enter new 18 char. descriptor of in 15?> = integer number 15
Enter low limit for in 15?> = 0
Enter high limit for in 15?> = 7

Enter new 6 char. name of in 16?> = in16
Enter new 18 char. descriptor of in 16?> = integer number 16
Enter low limit for in 16?> = 0
Enter high limit for in 16?> = 7

Enter new 6 char. name of in 17?> = in17
Enter new 18 char. descriptor of in 17?> = integer number 17
Enter low limit for in 17?> = 0
Enter high limit for in 17?> = 7

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 18?> = in18
Enter new 18 char. descriptor of in 18?> = integer number 18
Enter low limit for in 18?> = 0
Enter high limit for in 18?> = 7

Enter new 6 char. name of in 19?> = in19
Enter new 18 char. descriptor of in 19?> = integer number 19
Enter low limit for in 19?> = 0
Enter high limit for in 19?> = 7
MRPF?> = add 8 st

Enter new 6 char. name of st 0?> = minst
Enter new 18 char. descriptor of st 0?> = one minute timer
Enter low limit for st 0?> = 0
Enter high limit for st 0?> = 60

Enter new 6 char. name of st 1?> = fivest
Enter new 18 char. descriptor of st 1?> = five minute timer
Enter low limit for st 1?> = 0
Enter high limit for st 1?> = 300

Enter new 6 char. name of st 2?> = tenst
Enter new 18 char. descriptor of st 2?> = ten minute timer
Enter low limit for st 2?> = 0
Enter high limit for st 2?> = 600

Enter new 6 char. name of st 3?> = pigtmr
Enter new 18 char. descriptor of st 3?> = rc pig timer
Enter low limit for st 3?> = 0
Enter high limit for st 3?> = 255

Enter new 6 char. name of st 4?> = ostmr
Enter new 18 char. descriptor of st 4?> = one shot timer
Enter low limit for st 4?> = 0
Enter high limit for st 4?> = 75

Enter new 6 char. name of st 5?> = stime0
Enter new 18 char. descriptor of st 5?> = pgm error timer
Enter low limit for st 5?> = 0
Enter high limit for st 5?> = 10000

Enter new 6 char. name of st 6?> = stime1
Enter new 18 char. descriptor of st 6?> = pgm utility timer
Enter low limit for st 6?> = 0
Enter high limit for st 6?> = 10000

Enter new 6 char. name of st 7?> = sprtmr
Enter new 18 char. descriptor of st 7?> = spare sec timer
Enter low limit for st 7?> = 0
Enter high limit for st 7?> = 10000

MRPF?> = add 5 sg

Enter new 6 char. name of sg 0?> = note0

Enter new 18 char. descriptor of sg 0?> = total note 0

Enter maximum size for sg 0?> = 30

Enter new 6 char. name of sg 1?> = note1

Enter new 18 char. descriptor of sg 1?> = total note 1

Enter maximum size for sg 1?> = 30

Enter new 6 char. name of sg 2?> = note2

Enter new 18 char. descriptor of sg 2?> = total note 2

Enter maximum size for sg 2?> = 30

Enter new 6 char. name of sg 3?> = note3

Enter new 18 char. descriptor of sg 3?> = total note 3

Enter maximum size for sg 3?> = 30

Enter new 6 char. name of sg 4?> = note4

Enter new 18 char. descriptor of sg 4?> = total note 4

Enter maximum size for sg 4?> = 30

MRPF?> =

RPF?>

end of file encountered

RMS?>

end of file encountered

ALL1

```

RMS Version 6.3-2                                05-Dec-94 09:33:03
RMS?> = rpf de all
RPF?> = rpf de all
Invalid Command
Open error file SV0:[300,1]U.UPF ,error = -1
RPF?> = cr all,*
MRPF?> = add 25 fl

Enter new 6 char. name of fl 0?> = val0
Enter new 18 char. descriptor of fl 0?> = value number 0
Enter low limit for fl 0?> = 0
Enter high limit for fl 0?> = 10000

Enter new 6 char. name of fl 1?> = val1
Enter new 18 char. descriptor of fl 1?> = value number 1
Enter low limit for fl 1?> = 0
Enter high limit for fl 1?> = 10000

Enter new 6 char. name of fl 2?> = val2
Enter new 18 char. descriptor of fl 2?> = value number 2
Enter low limit for fl 2?> = 0
Enter high limit for fl 2?> = 10000

Enter new 6 char. name of fl 3?> = val3
Enter new 18 char. descriptor of fl 3?> = value number 3
Enter low limit for fl 3?> = 0
Enter high limit for fl 3?> = 10000

Enter new 6 char. name of fl 4?> = val4
Enter new 18 char. descriptor of fl 4?> = value number 4
Enter low limit for fl 4?> = 0
Enter high limit for fl 4?> = 10000

Enter new 6 char. name of fl 5?> = val5
Enter new 18 char. descriptor of fl 5?> = value number 5
Enter low limit for fl 5?> = 0
Enter high limit for fl 5?> = 10000

Enter new 6 char. name of fl 6?> = val6
Enter new 18 char. descriptor of fl 6?> = value number 6
Enter low limit for fl 6?> = 0
Enter high limit for fl 6?> = 10000

Enter new 6 char. name of fl 7?> = val7
Enter new 18 char. descriptor of fl 7?> = value number 7
Enter low limit for fl 7?> = 0
Enter high limit for fl 7?> = 10000

```

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fl 8?> = val8
Enter new 18 char. descriptor of fl 8?> = value number 8
Enter low limit for fl 8?> = 0
Enter high limit for fl 8?> = 10000

Enter new 6 char. name of fl 9?> = val9
Enter new 18 char. descriptor of fl 9?> = value number 9
Enter low limit for fl 9?> = 0
Enter high limit for fl 9?> = 10000

Enter new 6 char. name of fl 10?> = val10
Enter new 18 char. descriptor of fl 10?> = value number 10
Enter low limit for fl 10?> = 0
Enter high limit for fl 10?> = 10000

Enter new 6 char. name of fl 11?> = val11
Enter new 18 char. descriptor of fl 11?> = value number 11
Enter low limit for fl 11?> = 0
Enter high limit for fl 11?> = 10000

Enter new 6 char. name of fl 12?> = val12
Enter new 18 char. descriptor of fl 12?> = value number 12
Enter low limit for fl 12?> = 0
Enter high limit for fl 12?> = 10000

Enter new 6 char. name of fl 13?> = val13
Enter new 18 char. descriptor of fl 13?> = value number 13
Enter low limit for fl 13?> = 0
Enter high limit for fl 13?> = 10000

Enter new 6 char. name of fl 14?> = val14
Enter new 18 char. descriptor of fl 14?> = value number 14
Enter low limit for fl 14?> = 0
Enter high limit for fl 14?> = 10000

Enter new 6 char. name of fl 15?> = val15
Enter new 18 char. descriptor of fl 15?> = value number 15
Enter low limit for fl 15?> = 0
Enter high limit for fl 15?> = 10000

Enter new 6 char. name of fl 16?> = val16
Enter new 18 char. descriptor of fl 16?> = value number 16
Enter low limit for fl 16?> = 0
Enter high limit for fl 16?> = 10000

Enter new 6 char. name of fl 17?> = val17
Enter new 18 char. descriptor of fl 17?> = value number 17
Enter low limit for fl 17?> = 0
Enter high limit for fl 17?> = 10000

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fl 18?> = val18
Enter new 18 char. descriptor of fl 18?> = value number 18
Enter low limit for fl 18?> = 0
Enter high limit for fl 18?> = 10000

Enter new 6 char. name of fl 19?> = val19
Enter new 18 char. descriptor of fl 19?> = value number 19
Enter low limit for fl 19?> = 0
Enter high limit for fl 19?> = 10000

Enter new 6 char. name of fl 20?> = val20
Enter new 18 char. descriptor of fl 20?> = value number 20
Enter low limit for fl 20?> = 0
Enter high limit for fl 20?> = 10000

Enter new 6 char. name of fl 21?> = val21
Enter new 18 char. descriptor of fl 21?> = value number 21
Enter low limit for fl 21?> = 0
Enter high limit for fl 21?> = 10000

Enter new 6 char. name of fl 22?> = val22
Enter new 18 char. descriptor of fl 22?> = value number 22
Enter low limit for fl 22?> = 0
Enter high limit for fl 22?> = 10000

Enter new 6 char. name of fl 23?> = val23
Enter new 18 char. descriptor of fl 23?> = value number 23
Enter low limit for fl 23?> = 0
Enter high limit for fl 23?> = 10000

Enter new 6 char. name of fl 24?> = val24
Enter new 18 char. descriptor of fl 24?> = value number 24
Enter low limit for fl 24?> = 0
Enter high limit for fl 24?> = 10000
MRPF?> = add 50 in

Enter new 6 char. name of in 0?> = in0
Enter new 18 char. descriptor of in 0?> = integer number 0
Enter low limit for in 0?> = 0
Enter high limit for in 0?> = 7

Enter new 6 char. name of in 1?> = in1
Enter new 18 char. descriptor of in 1?> = integer number 1
Enter low limit for in 1?> = 0
Enter high limit for in 1?> = 7

Enter new 6 char. name of in 2?> = in2
Enter new 18 char. descriptor of in 2?> = integer number 2
Enter low limit for in 2?> = 0
Enter high limit for in 2?> = 7

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 3?> = in3
Enter new 18 char. descriptor of in 3?> = integer number 3
Enter low limit for in 3?> = 0
Enter high limit for in 3?> = 7

Enter new 6 char. name of in 4?> = in4
Enter new 18 char. descriptor of in 4?> = integer number 4
Enter low limit for in 4?> = 0
Enter high limit for in 4?> = 7

Enter new 6 char. name of in 5?> = in5
Enter new 18 char. descriptor of in 5?> = integer number 5
Enter low limit for in 5?> = 0
Enter high limit for in 5?> = 7

Enter new 6 char. name of in 6?> = in6
Enter new 18 char. descriptor of in 6?> = integer number 6
Enter low limit for in 6?> = 0
Enter high limit for in 6?> = 7

Enter new 6 char. name of in 7?> = in7
Enter new 18 char. descriptor of in 7?> = integer number 7
Enter low limit for in 7?> = 0
Enter high limit for in 7?> = 7

Enter new 6 char. name of in 8?> = in8
Enter new 18 char. descriptor of in 8?> = integer number 8
Enter low limit for in 8?> = 0
Enter high limit for in 8?> = 7

Enter new 6 char. name of in 9?> = in9
Enter new 18 char. descriptor of in 9?> = integer number 9
Enter low limit for in 9?> = 0
Enter high limit for in 9?> = 7

Enter new 6 char. name of in 10?> = in10
Enter new 18 char. descriptor of in 10?> = integer number 10
Enter low limit for in 10?> = 0
Enter high limit for in 10?> = 7

Enter new 6 char. name of in 11?> = in11
Enter new 18 char. descriptor of in 11?> = integer number 11
Enter low limit for in 11?> = 0
Enter high limit for in 11?> = 7

Enter new 6 char. name of in 12?> = in12
Enter new 18 char. descriptor of in 12?> = integer number 12
Enter low limit for in 12?> = 0
Enter high limit for in 12?> = 7

Enter new 6 char. name of in 13?> = in13
 Enter new 18 char. descriptor of in 13?> = integer number 13
 Enter low limit for in 13?> = 0
 Enter high limit for in 13?> = 7

Enter new 6 char. name of in 14?> = in14
 Enter new 18 char. descriptor of in 14?> = integer number 14
 Enter low limit for in 14?> = 0
 Enter high limit for in 14?> = 7

Enter new 6 char. name of in 15?> = in15
 Enter new 18 char. descriptor of in 15?> = integer number 15
 Enter low limit for in 15?> = 0
 Enter high limit for in 15?> = 7

Enter new 6 char. name of in 16?> = in16
 Enter new 18 char. descriptor of in 16?> = integer number 16
 Enter low limit for in 16?> = 0
 Enter high limit for in 16?> = 7

Enter new 6 char. name of in 17?> = in17
 Enter new 18 char. descriptor of in 17?> = integer number 17
 Enter low limit for in 17?> = 0
 Enter high limit for in 17?> = 7

Enter new 6 char. name of in 18?> = in18
 Enter new 18 char. descriptor of in 18?> = integer number 18
 Enter low limit for in 18?> = 0
 Enter high limit for in 18?> = 7

Enter new 6 char. name of in 19?> = in19
 Enter new 18 char. descriptor of in 19?> = integer number 19
 Enter low limit for in 19?> = 0
 Enter high limit for in 19?> = 7

Enter new 6 char. name of in 20?> = in20
 Enter new 18 char. descriptor of in 20?> = integer number 20
 Enter low limit for in 20?> = 0
 Enter high limit for in 20?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 21?> = in21
 Enter new 18 char. descriptor of in 21?> = integer number 21
 Enter low limit for in 21?> = 0
 Enter high limit for in 21?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 22?> = in22
 Enter new 18 char. descriptor of in 22?> = integer number 22
 Enter low limit for in 22?> = 0
 Enter high limit for in 22?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 23?> = in23
 Enter new 18 char. descriptor of in 23?> = integer number 23
 Enter low limit for in 23?> = 0
 Enter high limit for in 23?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 24?> = in24
 Enter new 18 char. descriptor of in 24?> = integer number 24
 Enter low limit for in 24?> = 0
 Enter high limit for in 24?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 25?> = in25
 Enter new 18 char. descriptor of in 25?> = integer number 25
 Enter low limit for in 25?> = 0
 Enter high limit for in 25?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 26?> = in26
 Enter new 18 char. descriptor of in 26?> = integer number 26
 Enter low limit for in 26?> = 0
 Enter high limit for in 26?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 27?> = in27
 Enter new 18 char. descriptor of in 27?> = integer number 27
 Enter low limit for in 27?> = 0
 Enter high limit for in 27?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 28?> = in28
 Enter new 18 char. descriptor of in 28?> = integer number 28
 Enter low limit for in 28?> = 0
 Enter high limit for in 28?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 29?> = in29
 Enter new 18 char. descriptor of in 29?> = integer number 29
 Enter low limit for in 29?> = 0
 Enter high limit for in 29?> = 100000
 High limit is less than low - setting high to 32767

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 30?> = in30
Enter new 18 char. descriptor of in 30?> = integer number 30
Enter low limit for in 30?> = 0
Enter high limit for in 30?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 31?> = in31
Enter new 18 char. descriptor of in 31?> = integer number 31
Enter low limit for in 31?> = 0
Enter high limit for in 31?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 32?> = in32
Enter new 18 char. descriptor of in 32?> = integer number 32
Enter low limit for in 32?> = 0
Enter high limit for in 32?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 33?> = in33
Enter new 18 char. descriptor of in 33?> = integer number 33
Enter low limit for in 33?> = 0
Enter high limit for in 33?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 34?> = in34
Enter new 18 char. descriptor of in 34?> = integer number 34
Enter low limit for in 34?> = 0
Enter high limit for in 34?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 35?> = in35
Enter new 18 char. descriptor of in 35?> = integer number 35
Enter low limit for in 35?> = 0
Enter high limit for in 35?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 36?> = in36
Enter new 18 char. descriptor of in 36?> = integer number 36
Enter low limit for in 36?> = 0
Enter high limit for in 36?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 37?> = in37
Enter new 18 char. descriptor of in 37?> = integer number 37
Enter low limit for in 37?> = 0
Enter high limit for in 37?> = 100000
High limit is less than low - setting high to 32767

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 38?> = in38
Enter new 18 char. descriptor of in 38?> = integer number 38
Enter low limit for in 38?> = 0
Enter high limit for in 38?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 39?> = in39
Enter new 18 char. descriptor of in 39?> = integer number 39
Enter low limit for in 39?> = 0
Enter high limit for in 39?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 40?> = imisc1
Enter new 18 char. descriptor of in 40?> = misc1 val1
Enter low limit for in 40?> = 0
Enter high limit for in 40?> = 128

Enter new 6 char. name of in 41?> = imisc2
Enter new 18 char. descriptor of in 41?> = misc2 val2
Enter low limit for in 41?> = 0
Enter high limit for in 41?> = 128

Enter new 6 char. name of in 42?> = adi
Enter new 18 char. descriptor of in 42?> = alm des index
Enter low limit for in 42?> = 0
Enter high limit for in 42?> = 80

Enter new 6 char. name of in 43?> = lmpvar
Enter new 18 char. descriptor of in 43?> = lamp var
Enter low limit for in 43?> = 0
Enter high limit for in 43?> = 4

Enter new 6 char. name of in 44?> = led0
Enter new 18 char. descriptor of in 44?> = led 0
Enter low limit for in 44?> = 0
Enter high limit for in 44?> = 63

Enter new 6 char. name of in 45?> = led1
Enter new 18 char. descriptor of in 45?> = led 1
Enter low limit for in 45?> = 0
Enter high limit for in 45?> = 63

Enter new 6 char. name of in 46?> = led2
Enter new 18 char. descriptor of in 46?> = led 2
Enter low limit for in 46?> = 0
Enter high limit for in 46?> = 63

Enter new 6 char. name of in 47?> = ispar0
 Enter new 18 char. descriptor of in 47?> = spare int 0
 Enter low limit for in 47?> = 0
 Enter high limit for in 47?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 48?> = ispar1
 Enter new 18 char. descriptor of in 48?> = spare int 1
 Enter low limit for in 48?> = 0
 Enter high limit for in 48?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 49?> = ispar2
 Enter new 18 char. descriptor of in 49?> = spare int 2
 Enter low limit for in 49?> = 0
 Enter high limit for in 49?> = 37000
 High limit is less than low - setting high to 32767
 MRPF?> = add 5 st

Enter new 6 char. name of st 0?> = stmr0
 Enter new 18 char. descriptor of st 0?> = second timer 0
 Enter low limit for st 0?> = 0
 Enter high limit for st 0?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of st 1?> = stmr1
 Enter new 18 char. descriptor of st 1?> = second timer 1
 Enter low limit for st 1?> = 0
 Enter high limit for st 1?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of st 2?> = stmr2
 Enter new 18 char. descriptor of st 2?> = second timer 2
 Enter low limit for st 2?> = 0
 Enter high limit for st 2?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of st 3?> = stmr3
 Enter new 18 char. descriptor of st 3?> = second timer 3
 Enter low limit for st 3?> = 0
 Enter high limit for st 3?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of st 4?> = stmr4
 Enter new 18 char. descriptor of st 4?> = second timer 4
 Enter low limit for st 4?> = 0
 Enter high limit for st 4?> = 37000
 High limit is less than low - setting high to 32767
 MRPF?> = add 5 mt

Enter new 6 char. name of mt 0?> = mtmr0
 Enter new 18 char. descriptor of mt 0?> = minute timer 0
 Enter low limit for mt 0?> = 0
 Enter high limit for mt 0?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 1?> = mtmr1
 Enter new 18 char. descriptor of mt 1?> = minute timer 1
 Enter low limit for mt 1?> = 0
 Enter high limit for mt 1?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 2?> = mtmr2
 Enter new 18 char. descriptor of mt 2?> = minute timer 2
 Enter low limit for mt 2?> = 0
 Enter high limit for mt 2?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 3?> = mtmr3
 Enter new 18 char. descriptor of mt 3?> = minute timer 3
 Enter low limit for mt 3?> = 0
 Enter high limit for mt 3?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 4?> = mtmr4
 Enter new 18 char. descriptor of mt 4?> = minute timer 4
 Enter low limit for mt 4?> = 0
 Enter high limit for mt 4?> = 37000
 High limit is less than low - setting high to 32767

MRPF?>

end of file encountered

RPF?>

end of file encountered

RMS?>

end of file encountered

FARM

WHC-SD-534-CSWL-001 REV 3

RMS Version 6.3-2

05-Dec-94 09:32:21

RMS?> = rpf de farm

RPF?> = rpf de farm

RPF?> = cr farm,*

MRPF?> = add 20 fl

Enter new 6 char. name of fl 0?> = val0

Enter new 18 char. descriptor of fl 0?> = value number 0

Enter low limit for fl 0?> = 0

Enter high limit for fl 0?> = 10000

Enter new 6 char. name of fl 1?> = val1

Enter new 18 char. descriptor of fl 1?> = value number 1

Enter low limit for fl 1?> = 0

Enter high limit for fl 1?> = 10000

Enter new 6 char. name of fl 2?> = val2

Enter new 18 char. descriptor of fl 2?> = value number 2

Enter low limit for fl 2?> = 0

Enter high limit for fl 2?> = 10000

Enter new 6 char. name of fl 3?> = val3

Enter new 18 char. descriptor of fl 3?> = value number 3

Enter low limit for fl 3?> = 0

Enter high limit for fl 3?> = 10000

Enter new 6 char. name of fl 4?> = val4

Enter new 18 char. descriptor of fl 4?> = value number 4

Enter low limit for fl 4?> = 0

Enter high limit for fl 4?> = 10000

Enter new 6 char. name of fl 5?> = val5

Enter new 18 char. descriptor of fl 5?> = value number 5

Enter low limit for fl 5?> = 0

Enter high limit for fl 5?> = 10000

Enter new 6 char. name of fl 6?> = val6

Enter new 18 char. descriptor of fl 6?> = value number 6

Enter low limit for fl 6?> = 0

Enter high limit for fl 6?> = 10000

Enter new 6 char. name of fl 7?> = val7

Enter new 18 char. descriptor of fl 7?> = value number 7

Enter low limit for fl 7?> = 0

Enter high limit for fl 7?> = 10000

Enter new 6 char. name of fl 8?> = val8

Enter new 18 char. descriptor of fl 8?> = value number 8

Enter low limit for fl 8?> = 0

Enter high limit for fl 8?> = 10000

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fl 9?> = val9
Enter new 18 char. descriptor of fl 9?> = value number 9
Enter low limit for fl 9?> = 0
Enter high limit for fl 9?> = 10000

Enter new 6 char. name of fl 10?> = val10
Enter new 18 char. descriptor of fl 10?> = value number 10
Enter low limit for fl 10?> = 0
Enter high limit for fl 10?> = 10000

Enter new 6 char. name of fl 11?> = val11
Enter new 18 char. descriptor of fl 11?> = value number 11
Enter low limit for fl 11?> = 0
Enter high limit for fl 11?> = 10000

Enter new 6 char. name of fl 12?> = val12
Enter new 18 char. descriptor of fl 12?> = value number 12
Enter low limit for fl 12?> = 0
Enter high limit for fl 12?> = 10000

Enter new 6 char. name of fl 13?> = val13
Enter new 18 char. descriptor of fl 13?> = value number 13
Enter low limit for fl 13?> = 0
Enter high limit for fl 13?> = 10000

Enter new 6 char. name of fl 14?> = val14
Enter new 18 char. descriptor of fl 14?> = value number 14
Enter low limit for fl 14?> = 0
Enter high limit for fl 14?> = 10000

Enter new 6 char. name of fl 15?> = val15
Enter new 18 char. descriptor of fl 15?> = value number 15
Enter low limit for fl 15?> = 0
Enter high limit for fl 15?> = 10000

Enter new 6 char. name of fl 16?> = val16
Enter new 18 char. descriptor of fl 16?> = value number 16
Enter low limit for fl 16?> = 0
Enter high limit for fl 16?> = 10000

Enter new 6 char. name of fl 17?> = val17
Enter new 18 char. descriptor of fl 17?> = value number 17
Enter low limit for fl 17?> = 0
Enter high limit for fl 17?> = 10000

Enter new 6 char. name of fl 18?> = val18
Enter new 18 char. descriptor of fl 18?> = value number 18
Enter low limit for fl 18?> = 0
Enter high limit for fl 18?> = 10000

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fl 19?> = val19
Enter new 18 char. descriptor of fl 19?> = value number 19
Enter low limit for fl 19?> = 0
Enter high limit for fl 19?> = 10000
MRPF?> = add 35 in

Enter new 6 char. name of in 0?> = in0
Enter new 18 char. descriptor of in 0?> = integer number 0
Enter low limit for in 0?> = 0
Enter high limit for in 0?> = 7

Enter new 6 char. name of in 1?> = in1
Enter new 18 char. descriptor of in 1?> = integer number 1
Enter low limit for in 1?> = 0
Enter high limit for in 1?> = 7

Enter new 6 char. name of in 2?> = in2
Enter new 18 char. descriptor of in 2?> = integer number 2
Enter low limit for in 2?> = 0
Enter high limit for in 2?> = 7

Enter new 6 char. name of in 3?> = in3
Enter new 18 char. descriptor of in 3?> = integer number 3
Enter low limit for in 3?> = 0
Enter high limit for in 3?> = 7

Enter new 6 char. name of in 4?> = in4
Enter new 18 char. descriptor of in 4?> = integer number 4
Enter low limit for in 4?> = 0
Enter high limit for in 4?> = 7

Enter new 6 char. name of in 5?> = in5
Enter new 18 char. descriptor of in 5?> = integer number 5
Enter low limit for in 5?> = 0
Enter high limit for in 5?> = 7

Enter new 6 char. name of in 6?> = in6
Enter new 18 char. descriptor of in 6?> = integer number 6
Enter low limit for in 6?> = 0
Enter high limit for in 6?> = 7

Enter new 6 char. name of in 7?> = in7
Enter new 18 char. descriptor of in 7?> = integer number 7
Enter low limit for in 7?> = 0
Enter high limit for in 7?> = 7

Enter new 6 char. name of in 8?> = in8
Enter new 18 char. descriptor of in 8?> = integer number 8
Enter low limit for in 8?> = 0
Enter high limit for in 8?> = 7

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 9?> = in9
Enter new 18 char. descriptor of in 9?> = integer number 9
Enter low limit for in 9?> = 0
Enter high limit for in 9?> = 7

Enter new 6 char. name of in 10?> = in10
Enter new 18 char. descriptor of in 10?> = integer number 10
Enter low limit for in 10?> = 0
Enter high limit for in 10?> = 7

Enter new 6 char. name of in 11?> = in11
Enter new 18 char. descriptor of in 11?> = integer number 11
Enter low limit for in 11?> = 0
Enter high limit for in 11?> = 7

Enter new 6 char. name of in 12?> = in12
Enter new 18 char. descriptor of in 12?> = integer number 12
Enter low limit for in 12?> = 0
Enter high limit for in 12?> = 7

Enter new 6 char. name of in 13?> = in13
Enter new 18 char. descriptor of in 13?> = integer number 13
Enter low limit for in 13?> = 0
Enter high limit for in 13?> = 7

Enter new 6 char. name of in 14?> = in14
Enter new 18 char. descriptor of in 14?> = integer number 14
Enter low limit for in 14?> = 0
Enter high limit for in 14?> = 7

Enter new 6 char. name of in 15?> = in15
Enter new 18 char. descriptor of in 15?> = integer number 15
Enter low limit for in 15?> = 0
Enter high limit for in 15?> = 7

Enter new 6 char. name of in 16?> = in16
Enter new 18 char. descriptor of in 16?> = integer number 16
Enter low limit for in 16?> = 0
Enter high limit for in 16?> = 7

Enter new 6 char. name of in 17?> = in17
Enter new 18 char. descriptor of in 17?> = integer number 17
Enter low limit for in 17?> = 0
Enter high limit for in 17?> = 7

Enter new 6 char. name of in 18?> = in18
Enter new 18 char. descriptor of in 18?> = integer number 18
Enter low limit for in 18?> = 0
Enter high limit for in 18?> = 7

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 19?> = in19
Enter new 18 char. descriptor of in 19?> = integer number 19
Enter low limit for in 19?> = 0
Enter high limit for in 19?> = 7

Enter new 6 char. name of in 20?> = in20
Enter new 18 char. descriptor of in 20?> = integer number 20
Enter low limit for in 20?> = 0
Enter high limit for in 20?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 21?> = in21
Enter new 18 char. descriptor of in 21?> = integer number 21
Enter low limit for in 21?> = 0
Enter high limit for in 21?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 22?> = in22
Enter new 18 char. descriptor of in 22?> = integer number 22
Enter low limit for in 22?> = 0
Enter high limit for in 22?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 23?> = in23
Enter new 18 char. descriptor of in 23?> = integer number 23
Enter low limit for in 23?> = 0
Enter high limit for in 23?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 24?> = in24
Enter new 18 char. descriptor of in 24?> = integer number 24
Enter low limit for in 24?> = 0
Enter high limit for in 24?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 25?> = imisc1
Enter new 18 char. descriptor of in 25?> = misc1 val1
Enter low limit for in 25?> = 0
Enter high limit for in 25?> = 128

Enter new 6 char. name of in 26?> = imisc2
Enter new 18 char. descriptor of in 26?> = misc2 val2
Enter low limit for in 26?> = 0
Enter high limit for in 26?> = 128

Enter new 6 char. name of in 27?> = adi
Enter new 18 char. descriptor of in 27?> = alm des index
Enter low limit for in 27?> = 0
Enter high limit for in 27?> = 80

Enter new 6 char. name of in 28?> = lmpvar
 Enter new 18 char. descriptor of in 28?> = lamp var
 Enter low limit for in 28?> = 0
 Enter high limit for in 28?> = 4

Enter new 6 char. name of in 29?> = led0
 Enter new 18 char. descriptor of in 29?> = led 0
 Enter low limit for in 29?> = 0
 Enter high limit for in 29?> = 63

Enter new 6 char. name of in 30?> = led1
 Enter new 18 char. descriptor of in 30?> = led 1
 Enter low limit for in 30?> = 0
 Enter high limit for in 30?> = 63

Enter new 6 char. name of in 31?> = led2
 Enter new 18 char. descriptor of in 31?> = led 2
 Enter low limit for in 31?> = 0
 Enter high limit for in 31?> = 63

Enter new 6 char. name of in 32?> = ispar0
 Enter new 18 char. descriptor of in 32?> = spare int 0
 Enter low limit for in 32?> = 0
 Enter high limit for in 32?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 33?> = ispar1
 Enter new 18 char. descriptor of in 33?> = spare int 1
 Enter low limit for in 33?> = 0
 Enter high limit for in 33?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 34?> = ispar2
 Enter new 18 char. descriptor of in 34?> = spare int 2
 Enter low limit for in 34?> = 0
 Enter high limit for in 34?> = 37000
 High limit is less than low - setting high to 32767
 MRPF?> = add 5 st

Enter new 6 char. name of st 0?> = stmr0
 Enter new 18 char. descriptor of st 0?> = second timer 0
 Enter low limit for st 0?> = 0
 Enter high limit for st 0?> = 37000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of st 1?> = stmr1
 Enter new 18 char. descriptor of st 1?> = second timer 1
 Enter low limit for st 1?> = 0
 Enter high limit for st 1?> = 37000
 High limit is less than low - setting high to 32767

WMC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of st 2?> = stmr2
Enter new 18 char. descriptor of st 2?> = second timer 2
Enter low limit for st 2?> = 0
Enter high limit for st 2?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of st 3?> = stmr3
Enter new 18 char. descriptor of st 3?> = second timer 3
Enter low limit for st 3?> = 0
Enter high limit for st 3?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of st 4?> = stmr4
Enter new 18 char. descriptor of st 4?> = second timer 4
Enter low limit for st 4?> = 0
Enter high limit for st 4?> = 37000
High limit is less than low - setting high to 32767
MRPF?> = add 5 mt

Enter new 6 char. name of mt 0?> = mtrm0
Enter new 18 char. descriptor of mt 0?> = minute timer 0
Enter low limit for mt 0?> = 0
Enter high limit for mt 0?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 1?> = mtrm1
Enter new 18 char. descriptor of mt 1?> = minute timer 1
Enter low limit for mt 1?> = 0
Enter high limit for mt 1?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 2?> = mtrm2
Enter new 18 char. descriptor of mt 2?> = minute timer 2
Enter low limit for mt 2?> = 0
Enter high limit for mt 2?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 3?> = mtrm3
Enter new 18 char. descriptor of mt 3?> = minute timer 3
Enter low limit for mt 3?> = 0
Enter high limit for mt 3?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 4?> = mtrm4
Enter new 18 char. descriptor of mt 4?> = minute timer 4
Enter low limit for mt 4?> = 0
Enter high limit for mt 4?> = 37000
High limit is less than low - setting high to 32767
MRPF?> = add 5 sg

WMC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of sg 0?> = note0
Enter new 18 char. descriptor of sg 0?> = recipe string 0
Enter maximum size for sg 0?> = 20

Enter new 6 char. name of sg 1?> = note1
Enter new 18 char. descriptor of sg 1?> = recipe string 1
Enter maximum size for sg 1?> = 20

Enter new 6 char. name of sg 2?> = note2
Enter new 18 char. descriptor of sg 2?> = recipe string 2
Enter maximum size for sg 2?> = 20

Enter new 6 char. name of sg 3?> = note3
Enter new 18 char. descriptor of sg 3?> = recipe string 3
Enter maximum size for sg 3?> = 20

Enter new 6 char. name of sg 4?> = note4
Enter new 18 char. descriptor of sg 4?> = recipe string 4
Enter maximum size for sg 4?> = 20

MRPF?>

end of file encountered

RPF?>

end of file encountered

RMS?>

end of file encountered

EMPTY

RMS Version 6.3-2 05-Dec-94 09:32:42
RMS?> = rpf de emty
RPF?> = rpf de emty
Invalid Command
Open error file SVO:[300,1]U.UPF ,error = -1
RPF?> = cr emty,*
MRPF?> = add 25 fl

Enter new 6 char. name of fl 0?> = val0
Enter new 18 char. descriptor of fl 0?> = value number 0
Enter low limit for fl 0?> = 0
Enter high limit for fl 0?> = 10000

Enter new 6 char. name of fl 1?> = val1
Enter new 18 char. descriptor of fl 1?> = value number 1
Enter low limit for fl 1?> = 0
Enter high limit for fl 1?> = 10000

Enter new 6 char. name of fl 2?> = val2
Enter new 18 char. descriptor of fl 2?> = value number 2
Enter low limit for fl 2?> = 0
Enter high limit for fl 2?> = 10000

Enter new 6 char. name of fl 3?> = val3
Enter new 18 char. descriptor of fl 3?> = value number 3
Enter low limit for fl 3?> = 0
Enter high limit for fl 3?> = 10000

Enter new 6 char. name of fl 4?> = val4
Enter new 18 char. descriptor of fl 4?> = value number 4
Enter low limit for fl 4?> = 0
Enter high limit for fl 4?> = 10000

Enter new 6 char. name of fl 5?> = val5
Enter new 18 char. descriptor of fl 5?> = value number 5
Enter low limit for fl 5?> = 0
Enter high limit for fl 5?> = 10000

Enter new 6 char. name of fl 6?> = val6
Enter new 18 char. descriptor of fl 6?> = value number 6
Enter low limit for fl 6?> = 0
Enter high limit for fl 6?> = 10000

Enter new 6 char. name of fl 7?> = val7
Enter new 18 char. descriptor of fl 7?> = value number 7
Enter low limit for fl 7?> = 0
Enter high limit for fl 7?> = 10000

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fl 8?> = val8
Enter new 18 char. descriptor of fl 8?> = value number 8
Enter low limit for fl 8?> = 0
Enter high limit for fl 8?> = 10000

Enter new 6 char. name of fl 9?> = val9
Enter new 18 char. descriptor of fl 9?> = value number 9
Enter low limit for fl 9?> = 0
Enter high limit for fl 9?> = 10000

Enter new 6 char. name of fl 10?> = val10
Enter new 18 char. descriptor of fl 10?> = value number 10
Enter low limit for fl 10?> = 0
Enter high limit for fl 10?> = 10000

Enter new 6 char. name of fl 11?> = val11
Enter new 18 char. descriptor of fl 11?> = value number 11
Enter low limit for fl 11?> = 0
Enter high limit for fl 11?> = 10000

Enter new 6 char. name of fl 12?> = val12
Enter new 18 char. descriptor of fl 12?> = value number 12
Enter low limit for fl 12?> = 0
Enter high limit for fl 12?> = 10000

Enter new 6 char. name of fl 13?> = val13
Enter new 18 char. descriptor of fl 13?> = value number 13
Enter low limit for fl 13?> = 0
Enter high limit for fl 13?> = 10000

Enter new 6 char. name of fl 14?> = val14
Enter new 18 char. descriptor of fl 14?> = value number 14
Enter low limit for fl 14?> = 0
Enter high limit for fl 14?> = 10000

Enter new 6 char. name of fl 15?> = val15
Enter new 18 char. descriptor of fl 15?> = value number 15
Enter low limit for fl 15?> = 0
Enter high limit for fl 15?> = 10000

Enter new 6 char. name of fl 16?> = val16
Enter new 18 char. descriptor of fl 16?> = value number 16
Enter low limit for fl 16?> = 0
Enter high limit for fl 16?> = 10000

Enter new 6 char. name of fl 17?> = val17
Enter new 18 char. descriptor of fl 17?> = value number 17
Enter low limit for fl 17?> = 0
Enter high limit for fl 17?> = 10000

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fl 18?> = val18
Enter new 18 char. descriptor of fl 18?> = value number 18
Enter low limit for fl 18?> = 0
Enter high limit for fl 18?> = 10000

Enter new 6 char. name of fl 19?> = val19
Enter new 18 char. descriptor of fl 19?> = value number 19
Enter low limit for fl 19?> = 0
Enter high limit for fl 19?> = 10000

Enter new 6 char. name of fl 20?> = val20
Enter new 18 char. descriptor of fl 20?> = value number 20
Enter low limit for fl 20?> = 0
Enter high limit for fl 20?> = 10000

Enter new 6 char. name of fl 21?> = val21
Enter new 18 char. descriptor of fl 21?> = value number 21
Enter low limit for fl 21?> = 0
Enter high limit for fl 21?> = 10000

Enter new 6 char. name of fl 22?> = val22
Enter new 18 char. descriptor of fl 22?> = value number 22
Enter low limit for fl 22?> = 0
Enter high limit for fl 22?> = 10000

Enter new 6 char. name of fl 23?> = val23
Enter new 18 char. descriptor of fl 23?> = value number 23
Enter low limit for fl 23?> = 0
Enter high limit for fl 23?> = 10000

Enter new 6 char. name of fl 24?> = val24
Enter new 18 char. descriptor of fl 24?> = value number 24
Enter low limit for fl 24?> = 0
Enter high limit for fl 24?> = 10000
MRPF?> = add 50 in

Enter new 6 char. name of in 0?> = in0
Enter new 18 char. descriptor of in 0?> = integer number 0
Enter low limit for in 0?> = 0
Enter high limit for in 0?> = 7

Enter new 6 char. name of in 1?> = in1
Enter new 18 char. descriptor of in 1?> = integer number 1
Enter low limit for in 1?> = 0
Enter high limit for in 1?> = 7

Enter new 6 char. name of in 2?> = in2
Enter new 18 char. descriptor of in 2?> = integer number 2
Enter low limit for in 2?> = 0
Enter high limit for in 2?> = 7

WMC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 3?> = in3
Enter new 18 char. descriptor of in 3?> = integer number 3
Enter low limit for in 3?> = 0
Enter high limit for in 3?> = 7

Enter new 6 char. name of in 4?> = in4
Enter new 18 char. descriptor of in 4?> = integer number 4
Enter low limit for in 4?> = 0
Enter high limit for in 4?> = 7

Enter new 6 char. name of in 5?> = in5
Enter new 18 char. descriptor of in 5?> = integer number 5
Enter low limit for in 5?> = 0
Enter high limit for in 5?> = 7

Enter new 6 char. name of in 6?> = in6
Enter new 18 char. descriptor of in 6?> = integer number 6
Enter low limit for in 6?> = 0
Enter high limit for in 6?> = 7

Enter new 6 char. name of in 7?> = in7
Enter new 18 char. descriptor of in 7?> = integer number 7
Enter low limit for in 7?> = 0
Enter high limit for in 7?> = 7

Enter new 6 char. name of in 8?> = in8
Enter new 18 char. descriptor of in 8?> = integer number 8
Enter low limit for in 8?> = 0
Enter high limit for in 8?> = 7

Enter new 6 char. name of in 9?> = in9
Enter new 18 char. descriptor of in 9?> = integer number 9
Enter low limit for in 9?> = 0
Enter high limit for in 9?> = 7

Enter new 6 char. name of in 10?> = in10
Enter new 18 char. descriptor of in 10?> = integer number 10
Enter low limit for in 10?> = 0
Enter high limit for in 10?> = 7

Enter new 6 char. name of in 11?> = in11
Enter new 18 char. descriptor of in 11?> = integer number 11
Enter low limit for in 11?> = 0
Enter high limit for in 11?> = 7

Enter new 6 char. name of in 12?> = in12
Enter new 18 char. descriptor of in 12?> = integer number 12
Enter low limit for in 12?> = 0
Enter high limit for in 12?> = 7

Enter new 6 char. name of in 13?> = in13
 Enter new 18 char. descriptor of in 13?> = integer number 13
 Enter low limit for in 13?> = 0
 Enter high limit for in 13?> = 7

Enter new 6 char. name of in 14?> = in14
 Enter new 18 char. descriptor of in 14?> = integer number 14
 Enter low limit for in 14?> = 0
 Enter high limit for in 14?> = 7

Enter new 6 char. name of in 15?> = in15
 Enter new 18 char. descriptor of in 15?> = integer number 15
 Enter low limit for in 15?> = 0
 Enter high limit for in 15?> = 7

Enter new 6 char. name of in 16?> = in16
 Enter new 18 char. descriptor of in 16?> = integer number 16
 Enter low limit for in 16?> = 0
 Enter high limit for in 16?> = 7

Enter new 6 char. name of in 17?> = in17
 Enter new 18 char. descriptor of in 17?> = integer number 17
 Enter low limit for in 17?> = 0
 Enter high limit for in 17?> = 7

Enter new 6 char. name of in 18?> = in18
 Enter new 18 char. descriptor of in 18?> = integer number 18
 Enter low limit for in 18?> = 0
 Enter high limit for in 18?> = 7

Enter new 6 char. name of in 19?> = in19
 Enter new 18 char. descriptor of in 19?> = integer number 19
 Enter low limit for in 19?> = 0
 Enter high limit for in 19?> = 7

Enter new 6 char. name of in 20?> = in20
 Enter new 18 char. descriptor of in 20?> = integer number 20
 Enter low limit for in 20?> = 0
 Enter high limit for in 20?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 21?> = in21
 Enter new 18 char. descriptor of in 21?> = integer number 21
 Enter low limit for in 21?> = 0
 Enter high limit for in 21?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 22?> = in22
 Enter new 18 char. descriptor of in 22?> = integer number 22
 Enter low limit for in 22?> = 0
 Enter high limit for in 22?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 23?> = in23
 Enter new 18 char. descriptor of in 23?> = integer number 23
 Enter low limit for in 23?> = 0
 Enter high limit for in 23?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 24?> = in24
 Enter new 18 char. descriptor of in 24?> = integer number 24
 Enter low limit for in 24?> = 0
 Enter high limit for in 24?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 25?> = in25
 Enter new 18 char. descriptor of in 25?> = integer number 25
 Enter low limit for in 25?> = 0
 Enter high limit for in 25?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 26?> = in26
 Enter new 18 char. descriptor of in 26?> = integer number 26
 Enter low limit for in 26?> = 0
 Enter high limit for in 26?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 27?> = in27
 Enter new 18 char. descriptor of in 27?> = integer number 27
 Enter low limit for in 27?> = 0
 Enter high limit for in 27?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 28?> = in28
 Enter new 18 char. descriptor of in 28?> = integer number 28
 Enter low limit for in 28?> = 0
 Enter high limit for in 28?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 29?> = in29
 Enter new 18 char. descriptor of in 29?> = integer number 29
 Enter low limit for in 29?> = 0
 Enter high limit for in 29?> = 100000
 High limit is less than low - setting high to 32767

Enter new 6 char. name of in 30?> = in30
Enter new 18 char. descriptor of in 30?> = integer number 30
Enter low limit for in 30?> = 0
Enter high limit for in 30?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 31?> = in31
Enter new 18 char. descriptor of in 31?> = integer number 31
Enter low limit for in 31?> = 0
Enter high limit for in 31?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 32?> = in32
Enter new 18 char. descriptor of in 32?> = integer number 32
Enter low limit for in 32?> = 0
Enter high limit for in 32?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 33?> = in33
Enter new 18 char. descriptor of in 33?> = integer number 33
Enter low limit for in 33?> = 0
Enter high limit for in 33?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 34?> = in34
Enter new 18 char. descriptor of in 34?> = integer number 34
Enter low limit for in 34?> = 0
Enter high limit for in 34?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 35?> = in35
Enter new 18 char. descriptor of in 35?> = integer number 35
Enter low limit for in 35?> = 0
Enter high limit for in 35?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 36?> = in36
Enter new 18 char. descriptor of in 36?> = integer number 36
Enter low limit for in 36?> = 0
Enter high limit for in 36?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 37?> = in37
Enter new 18 char. descriptor of in 37?> = integer number 37
Enter low limit for in 37?> = 0
Enter high limit for in 37?> = 100000
High limit is less than low - setting high to 32767

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 38?> = in38
Enter new 18 char. descriptor of in 38?> = integer number 38
Enter low limit for in 38?> = 0
Enter high limit for in 38?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 39?> = in39
Enter new 18 char. descriptor of in 39?> = integer number 39
Enter low limit for in 39?> = 0
Enter high limit for in 39?> = 100000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 40?> = imisc1
Enter new 18 char. descriptor of in 40?> = misc1 val1
Enter low limit for in 40?> = 0
Enter high limit for in 40?> = 128

Enter new 6 char. name of in 41?> = imisc2
Enter new 18 char. descriptor of in 41?> = misc2 val2
Enter low limit for in 41?> = 0
Enter high limit for in 41?> = 128

Enter new 6 char. name of in 42?> = adi
Enter new 18 char. descriptor of in 42?> = alm des index
Enter low limit for in 42?> = 0
Enter high limit for in 42?> = 80

Enter new 6 char. name of in 43?> = lmpvar
Enter new 18 char. descriptor of in 43?> = lamp var
Enter low limit for in 43?> = 0
Enter high limit for in 43?> = 4

Enter new 6 char. name of in 44?> = led0
Enter new 18 char. descriptor of in 44?> = led 0
Enter low limit for in 44?> = 0
Enter high limit for in 44?> = 63

Enter new 6 char. name of in 45?> = led1
Enter new 18 char. descriptor of in 45?> = led 1
Enter low limit for in 45?> = 0
Enter high limit for in 45?> = 63

Enter new 6 char. name of in 46?> = led2
Enter new 18 char. descriptor of in 46?> = led 2
Enter low limit for in 46?> = 0
Enter high limit for in 46?> = 63

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of in 47?> = ispar0
Enter new 18 char. descriptor of in 47?> = spare int 0
Enter low limit for in 47?> = 0
Enter high limit for in 47?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 48?> = ispar1
Enter new 18 char. descriptor of in 48?> = spare int 1
Enter low limit for in 48?> = 0
Enter high limit for in 48?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of in 49?> = ispar2
Enter new 18 char. descriptor of in 49?> = spare int 2
Enter low limit for in 49?> = 0
Enter high limit for in 49?> = 37000
High limit is less than low - setting high to 32767
MRPF?> = add 5 st

Enter new 6 char. name of st 0?> = stmr0
Enter new 18 char. descriptor of st 0?> = second timer 0
Enter low limit for st 0?> = 0
Enter high limit for st 0?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of st 1?> = stmr1
Enter new 18 char. descriptor of st 1?> = second timer 1
Enter low limit for st 1?> = 0
Enter high limit for st 1?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of st 2?> = stmr2
Enter new 18 char. descriptor of st 2?> = second timer 2
Enter low limit for st 2?> = 0
Enter high limit for st 2?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of st 3?> = stmr3
Enter new 18 char. descriptor of st 3?> = second timer 3
Enter low limit for st 3?> = 0
Enter high limit for st 3?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of st 4?> = stmr4
Enter new 18 char. descriptor of st 4?> = second timer 4
Enter low limit for st 4?> = 0
Enter high limit for st 4?> = 37000
High limit is less than low - setting high to 32767
MRPF?> = add 5 mt

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of mt 0?> = mtmr0
Enter new 18 char. descriptor of mt 0?> = minute timer 0
Enter low limit for mt 0?> = 0
Enter high limit for mt 0?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 1?> = mtmr1
Enter new 18 char. descriptor of mt 1?> = minute timer 1
Enter low limit for mt 1?> = 0
Enter high limit for mt 1?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 2?> = mtmr2
Enter new 18 char. descriptor of mt 2?> = minute timer 2
Enter low limit for mt 2?> = 0
Enter high limit for mt 2?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 3?> = mtmr3
Enter new 18 char. descriptor of mt 3?> = minute timer 3
Enter low limit for mt 3?> = 0
Enter high limit for mt 3?> = 37000
High limit is less than low - setting high to 32767

Enter new 6 char. name of mt 4?> = mtmr4
Enter new 18 char. descriptor of mt 4?> = minute timer 4
Enter low limit for mt 4?> = 0
Enter high limit for mt 4?> = 37000
High limit is less than low - setting high to 32767

MRPF?>

end of file encountered

RPF?>

end of file encountered

RMS?>

end of file encountered

~~XXXX~~ RMIS View/Print Document Cover Sheet ~~XXXX~~

This document was retrieved from the Documentation and Records Management (DRM) ISEARCH System. It is intended for Information only and may not be the most recent or updated version. Contact a Document Service Center (see Hanford Info for locations) if you need additional retrieval information.

Accession #: D196079533

Document #: SD-534-CSWL-001

Title/Desc:

SEQUENCE & BATCH LANGUAGE PROGRAMS & ALARM RELATED
C PROGRAMS FOR THE 242A MCS [SEC 2 OF 6 PAGE 127
TO 252]

Pages: 127

This document was too large to scan as a whole document, therefore it required breaking into smaller sections.

Document number: SD-534-CSWL-001

Section 2 of 6

Title: Sequence and Batch Language programs
and Alarm Related "C" Programs for the
242-A MCS

Date: 4/15/96 Revision: 3

Originator: Burger, J J

Co: _____

Recipient: _____

Co: _____

References: ECN- 627504

APPENDIX D - UNIT PROTOTYPE FILES (UPF)

TOTAL0

```

UPF Version 6.3-2      05-Dec-94 09:33:27
UPF?> = de total0
UPF?> = cr total0,*
Enter 4 character name of RPF to associate?> = all0
MUPF?> = add 432 fg
Enter new 6 char. name of fg 0 ?> = fai0
Enter new 6 char. name of fg 1 ?> = fai1
Enter new 6 char. name of fg 2 ?> = fai2
Enter new 6 char. name of fg 3 ?> = fai3
Enter new 6 char. name of fg 4 ?> = fai4
Enter new 6 char. name of fg 5 ?> = fai5
Enter new 6 char. name of fg 6 ?> = fai6
Enter new 6 char. name of fg 7 ?> = fai7
Enter new 6 char. name of fg 8 ?> = fai8
Enter new 6 char. name of fg 9 ?> = fai9
Enter new 6 char. name of fg 10 ?> = fai10
Enter new 6 char. name of fg 11 ?> = fai11
Enter new 6 char. name of fg 12 ?> = fai12
Enter new 6 char. name of fg 13 ?> = fai13
Enter new 6 char. name of fg 14 ?> = fai14
Enter new 6 char. name of fg 15 ?> = fai15
Enter new 6 char. name of fg 16 ?> = fai16
Enter new 6 char. name of fg 17 ?> = fai17
Enter new 6 char. name of fg 18 ?> = fai18
Enter new 6 char. name of fg 19 ?> = fai19
Enter new 6 char. name of fg 20 ?> = fai20
Enter new 6 char. name of fg 21 ?> = fai21
Enter new 6 char. name of fg 22 ?> = fai22
Enter new 6 char. name of fg 23 ?> = fai23
Enter new 6 char. name of fg 24 ?> = fai24
Enter new 6 char. name of fg 25 ?> = fai25
Enter new 6 char. name of fg 26 ?> = fai26
Enter new 6 char. name of fg 27 ?> = fai27
Enter new 6 char. name of fg 28 ?> = fai28
Enter new 6 char. name of fg 29 ?> = fai29
Enter new 6 char. name of fg 30 ?> = fai30
Enter new 6 char. name of fg 31 ?> = fai31
Enter new 6 char. name of fg 32 ?> = fai32
Enter new 6 char. name of fg 33 ?> = fai33
Enter new 6 char. name of fg 34 ?> = fai34
Enter new 6 char. name of fg 35 ?> = fai35
Enter new 6 char. name of fg 36 ?> = fai36
Enter new 6 char. name of fg 37 ?> = fai37
Enter new 6 char. name of fg 38 ?> = fai38
Enter new 6 char. name of fg 39 ?> = fai39
Enter new 6 char. name of fg 40 ?> = fai40
Enter new 6 char. name of fg 41 ?> = fai41
Enter new 6 char. name of fg 42 ?> = fai42
Enter new 6 char. name of fg 43 ?> = fai43
Enter new 6 char. name of fg 44 ?> = fai44

```

Enter new 6 char. name of fg 45 ?> = fai45
Enter new 6 char. name of fg 46 ?> = fai46
Enter new 6 char. name of fg 47 ?> = fai47
Enter new 6 char. name of fg 48 ?> = fai48
Enter new 6 char. name of fg 49 ?> = fai49
Enter new 6 char. name of fg 50 ?> = fai50
Enter new 6 char. name of fg 51 ?> = fai51
Enter new 6 char. name of fg 52 ?> = fai52
Enter new 6 char. name of fg 53 ?> = fai53
Enter new 6 char. name of fg 54 ?> = fai54
Enter new 6 char. name of fg 55 ?> = fai55
Enter new 6 char. name of fg 56 ?> = fai56
Enter new 6 char. name of fg 57 ?> = fai57
Enter new 6 char. name of fg 58 ?> = fai58
Enter new 6 char. name of fg 59 ?> = fai59
Enter new 6 char. name of fg 60 ?> = fai60
Enter new 6 char. name of fg 61 ?> = fai61
Enter new 6 char. name of fg 62 ?> = fai62
Enter new 6 char. name of fg 63 ?> = fai63
Enter new 6 char. name of fg 64 ?> = fai64
Enter new 6 char. name of fg 65 ?> = fai65
Enter new 6 char. name of fg 66 ?> = fai66
Enter new 6 char. name of fg 67 ?> = fai67
Enter new 6 char. name of fg 68 ?> = fai68
Enter new 6 char. name of fg 69 ?> = fai69
Enter new 6 char. name of fg 70 ?> = fai70
Enter new 6 char. name of fg 71 ?> = fai71
Enter new 6 char. name of fg 72 ?> = fai72
Enter new 6 char. name of fg 73 ?> = fai73
Enter new 6 char. name of fg 74 ?> = fai74
Enter new 6 char. name of fg 75 ?> = fai75
Enter new 6 char. name of fg 76 ?> = fai76
Enter new 6 char. name of fg 77 ?> = fai77
Enter new 6 char. name of fg 78 ?> = fai78
Enter new 6 char. name of fg 79 ?> = fai79
Enter new 6 char. name of fg 80 ?> = fai80
Enter new 6 char. name of fg 81 ?> = fai81
Enter new 6 char. name of fg 82 ?> = fai82
Enter new 6 char. name of fg 83 ?> = fai83
Enter new 6 char. name of fg 84 ?> = fai84
Enter new 6 char. name of fg 85 ?> = fai85
Enter new 6 char. name of fg 86 ?> = fai86
Enter new 6 char. name of fg 87 ?> = fai87
Enter new 6 char. name of fg 88 ?> = fai88
Enter new 6 char. name of fg 89 ?> = fai89
Enter new 6 char. name of fg 90 ?> = fai90
Enter new 6 char. name of fg 91 ?> = fai91
Enter new 6 char. name of fg 92 ?> = fai92
Enter new 6 char. name of fg 93 ?> = fai93
Enter new 6 char. name of fg 94 ?> = fai94

Enter new 6 char. name of fg 95 ?> = fai95
Enter new 6 char. name of fg 96 ?> = fai96
Enter new 6 char. name of fg 97 ?> = fai97
Enter new 6 char. name of fg 98 ?> = fai98
Enter new 6 char. name of fg 99 ?> = fai99
Enter new 6 char. name of fg 100 ?> = fai100
Enter new 6 char. name of fg 101 ?> = fai101
Enter new 6 char. name of fg 102 ?> = fai102
Enter new 6 char. name of fg 103 ?> = fai103
Enter new 6 char. name of fg 104 ?> = fai104
Enter new 6 char. name of fg 105 ?> = fai105
Enter new 6 char. name of fg 106 ?> = fai106
Enter new 6 char. name of fg 107 ?> = fai107
Enter new 6 char. name of fg 108 ?> = fai108
Enter new 6 char. name of fg 109 ?> = fai109
Enter new 6 char. name of fg 110 ?> = fai110
Enter new 6 char. name of fg 111 ?> = fai111
Enter new 6 char. name of fg 112 ?> = fai112
Enter new 6 char. name of fg 113 ?> = fai113
Enter new 6 char. name of fg 114 ?> = fai114
Enter new 6 char. name of fg 115 ?> = fai115
Enter new 6 char. name of fg 116 ?> = fai116
Enter new 6 char. name of fg 117 ?> = fai117
Enter new 6 char. name of fg 118 ?> = fai118
Enter new 6 char. name of fg 119 ?> = fai119
Enter new 6 char. name of fg 120 ?> = fai120
Enter new 6 char. name of fg 121 ?> = fai121
Enter new 6 char. name of fg 122 ?> = fai122
Enter new 6 char. name of fg 123 ?> = fai123
Enter new 6 char. name of fg 124 ?> = fai124
Enter new 6 char. name of fg 125 ?> = fai125
Enter new 6 char. name of fg 126 ?> = fai126
Enter new 6 char. name of fg 127 ?> = fai127
Enter new 6 char. name of fg 128 ?> = fdi0
Enter new 6 char. name of fg 129 ?> = fdi1
Enter new 6 char. name of fg 130 ?> = fdi2
Enter new 6 char. name of fg 131 ?> = fdi3
Enter new 6 char. name of fg 132 ?> = fdi4
Enter new 6 char. name of fg 133 ?> = fdi5
Enter new 6 char. name of fg 134 ?> = fdi6
Enter new 6 char. name of fg 135 ?> = fdi7
Enter new 6 char. name of fg 136 ?> = fdi8
Enter new 6 char. name of fg 137 ?> = fdi9
Enter new 6 char. name of fg 138 ?> = fdi10
Enter new 6 char. name of fg 139 ?> = fdi11
Enter new 6 char. name of fg 140 ?> = fdi12
Enter new 6 char. name of fg 141 ?> = fdi13
Enter new 6 char. name of fg 142 ?> = fdi14
Enter new 6 char. name of fg 143 ?> = fdi15
Enter new 6 char. name of fg 144 ?> = fdi16

Enter new 6 char. name of fg 145 ?> = fdi17
Enter new 6 char. name of fg 146 ?> = fdi18
Enter new 6 char. name of fg 147 ?> = fdi19
Enter new 6 char. name of fg 148 ?> = fdi20
Enter new 6 char. name of fg 149 ?> = fdi21
Enter new 6 char. name of fg 150 ?> = fdi22
Enter new 6 char. name of fg 151 ?> = fdi23
Enter new 6 char. name of fg 152 ?> = fdi24
Enter new 6 char. name of fg 153 ?> = fdi25
Enter new 6 char. name of fg 154 ?> = fdi26
Enter new 6 char. name of fg 155 ?> = fdi27
Enter new 6 char. name of fg 156 ?> = fdi28
Enter new 6 char. name of fg 157 ?> = fdi29
Enter new 6 char. name of fg 158 ?> = fdi30
Enter new 6 char. name of fg 159 ?> = fdi31
Enter new 6 char. name of fg 160 ?> = fdi32
Enter new 6 char. name of fg 161 ?> = fdi33
Enter new 6 char. name of fg 162 ?> = fdi34
Enter new 6 char. name of fg 163 ?> = fdi35
Enter new 6 char. name of fg 164 ?> = fdi36
Enter new 6 char. name of fg 165 ?> = fdi37
Enter new 6 char. name of fg 166 ?> = fdi38
Enter new 6 char. name of fg 167 ?> = fdi39
Enter new 6 char. name of fg 168 ?> = fdi40
Enter new 6 char. name of fg 169 ?> = fdi41
Enter new 6 char. name of fg 170 ?> = fdi42
Enter new 6 char. name of fg 171 ?> = fdi43
Enter new 6 char. name of fg 172 ?> = fdi44
Enter new 6 char. name of fg 173 ?> = fdi45
Enter new 6 char. name of fg 174 ?> = fdi46
Enter new 6 char. name of fg 175 ?> = fdi47
Enter new 6 char. name of fg 176 ?> = fdi48
Enter new 6 char. name of fg 177 ?> = fdi49
Enter new 6 char. name of fg 178 ?> = fdi50
Enter new 6 char. name of fg 179 ?> = fdi51
Enter new 6 char. name of fg 180 ?> = fdi52
Enter new 6 char. name of fg 181 ?> = fdi53
Enter new 6 char. name of fg 182 ?> = fdi54
Enter new 6 char. name of fg 183 ?> = fdi55
Enter new 6 char. name of fg 184 ?> = fdi56
Enter new 6 char. name of fg 185 ?> = fdi57
Enter new 6 char. name of fg 186 ?> = fdi58
Enter new 6 char. name of fg 187 ?> = fdi59
Enter new 6 char. name of fg 188 ?> = fdi60
Enter new 6 char. name of fg 189 ?> = fdi61
Enter new 6 char. name of fg 190 ?> = fdi62
Enter new 6 char. name of fg 191 ?> = fdi63
Enter new 6 char. name of fg 192 ?> = fdi64
Enter new 6 char. name of fg 193 ?> = fdi65
Enter new 6 char. name of fg 194 ?> = fdi66

Enter new 6 char. name of fg 195 ?> = fdi67
Enter new 6 char. name of fg 196 ?> = fdi68
Enter new 6 char. name of fg 197 ?> = fdi69
Enter new 6 char. name of fg 198 ?> = fdi70
Enter new 6 char. name of fg 199 ?> = fdi71
Enter new 6 char. name of fg 200 ?> = fdi72
Enter new 6 char. name of fg 201 ?> = fdi73
Enter new 6 char. name of fg 202 ?> = fdi74
Enter new 6 char. name of fg 203 ?> = fdi75
Enter new 6 char. name of fg 204 ?> = fdi76
Enter new 6 char. name of fg 205 ?> = fdi77
Enter new 6 char. name of fg 206 ?> = fdi78
Enter new 6 char. name of fg 207 ?> = fdi79
Enter new 6 char. name of fg 208 ?> = fdi80
Enter new 6 char. name of fg 209 ?> = fdi81
Enter new 6 char. name of fg 210 ?> = fdi82
Enter new 6 char. name of fg 211 ?> = fdi83
Enter new 6 char. name of fg 212 ?> = fdi84
Enter new 6 char. name of fg 213 ?> = fdi85
Enter new 6 char. name of fg 214 ?> = fdi86
Enter new 6 char. name of fg 215 ?> = fdi87
Enter new 6 char. name of fg 216 ?> = fdi88
Enter new 6 char. name of fg 217 ?> = fdi89
Enter new 6 char. name of fg 218 ?> = fdi90
Enter new 6 char. name of fg 219 ?> = fdi91
Enter new 6 char. name of fg 220 ?> = fdi92
Enter new 6 char. name of fg 221 ?> = fdi93
Enter new 6 char. name of fg 222 ?> = fdi94
Enter new 6 char. name of fg 223 ?> = fdi95
Enter new 6 char. name of fg 224 ?> = fdi96
Enter new 6 char. name of fg 225 ?> = fdi97
Enter new 6 char. name of fg 226 ?> = fdi98
Enter new 6 char. name of fg 227 ?> = fdi99
Enter new 6 char. name of fg 228 ?> = fdi100
Enter new 6 char. name of fg 229 ?> = fdi101
Enter new 6 char. name of fg 230 ?> = fdi102
Enter new 6 char. name of fg 231 ?> = fdi103
Enter new 6 char. name of fg 232 ?> = fdi104
Enter new 6 char. name of fg 233 ?> = fdi105
Enter new 6 char. name of fg 234 ?> = fdi106
Enter new 6 char. name of fg 235 ?> = fdi107
Enter new 6 char. name of fg 236 ?> = fdi108
Enter new 6 char. name of fg 237 ?> = fdi109
Enter new 6 char. name of fg 238 ?> = fdi110
Enter new 6 char. name of fg 239 ?> = fdi111
Enter new 6 char. name of fg 240 ?> = fdi112
Enter new 6 char. name of fg 241 ?> = fdi113
Enter new 6 char. name of fg 242 ?> = fdi114
Enter new 6 char. name of fg 243 ?> = fdi115
Enter new 6 char. name of fg 244 ?> = fdi116

Enter new 6 char. name of fg 245 ?> = fdil17
Enter new 6 char. name of fg 246 ?> = fdil18
Enter new 6 char. name of fg 247 ?> = fdil19
Enter new 6 char. name of fg 248 ?> = fdil20
Enter new 6 char. name of fg 249 ?> = fdil21
Enter new 6 char. name of fg 250 ?> = fdil22
Enter new 6 char. name of fg 251 ?> = fdil23
Enter new 6 char. name of fg 252 ?> = fdil24
Enter new 6 char. name of fg 253 ?> = fdil25
Enter new 6 char. name of fg 254 ?> = fdil26
Enter new 6 char. name of fg 255 ?> = fdil27
Enter new 6 char. name of fg 256 ?> = fdv0
Enter new 6 char. name of fg 257 ?> = fdv1
Enter new 6 char. name of fg 258 ?> = fdv2
Enter new 6 char. name of fg 259 ?> = fdv3
Enter new 6 char. name of fg 260 ?> = fdv4
Enter new 6 char. name of fg 261 ?> = fdv5
Enter new 6 char. name of fg 262 ?> = fdv6
Enter new 6 char. name of fg 263 ?> = fdv7
Enter new 6 char. name of fg 264 ?> = fdv8
Enter new 6 char. name of fg 265 ?> = fdv9
Enter new 6 char. name of fg 266 ?> = fdv10
Enter new 6 char. name of fg 267 ?> = fdv11
Enter new 6 char. name of fg 268 ?> = fdv12
Enter new 6 char. name of fg 269 ?> = fdv13
Enter new 6 char. name of fg 270 ?> = fdv14
Enter new 6 char. name of fg 271 ?> = fdv15
Enter new 6 char. name of fg 272 ?> = fdv16
Enter new 6 char. name of fg 273 ?> = fdv17
Enter new 6 char. name of fg 274 ?> = fdv18
Enter new 6 char. name of fg 275 ?> = fdv19
Enter new 6 char. name of fg 276 ?> = fdv20
Enter new 6 char. name of fg 277 ?> = fdv21
Enter new 6 char. name of fg 278 ?> = fdv22
Enter new 6 char. name of fg 279 ?> = fdv23
Enter new 6 char. name of fg 280 ?> = fdv24
Enter new 6 char. name of fg 281 ?> = fdv25
Enter new 6 char. name of fg 282 ?> = fdv26
Enter new 6 char. name of fg 283 ?> = fdv27
Enter new 6 char. name of fg 284 ?> = fdv28
Enter new 6 char. name of fg 285 ?> = fdv29
Enter new 6 char. name of fg 286 ?> = fdv30
Enter new 6 char. name of fg 287 ?> = fdv31
Enter new 6 char. name of fg 288 ?> = fdv32
Enter new 6 char. name of fg 289 ?> = fdv33
Enter new 6 char. name of fg 290 ?> = fdv34
Enter new 6 char. name of fg 291 ?> = fdv35
Enter new 6 char. name of fg 292 ?> = fdv36
Enter new 6 char. name of fg 293 ?> = fdv37
Enter new 6 char. name of fg 294 ?> = fdv38

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fg 295 ?> = fdv39
Enter new 6 char. name of fg 296 ?> = fdv40
Enter new 6 char. name of fg 297 ?> = fdv41
Enter new 6 char. name of fg 298 ?> = fdv42
Enter new 6 char. name of fg 299 ?> = fdv43
Enter new 6 char. name of fg 300 ?> = fdv44
Enter new 6 char. name of fg 301 ?> = fdv45
Enter new 6 char. name of fg 302 ?> = fdv46
Enter new 6 char. name of fg 303 ?> = fdv47
Enter new 6 char. name of fg 304 ?> = fdv48
Enter new 6 char. name of fg 305 ?> = fdv49
Enter new 6 char. name of fg 306 ?> = fdv50
Enter new 6 char. name of fg 307 ?> = fdv51
Enter new 6 char. name of fg 308 ?> = fdv52
Enter new 6 char. name of fg 309 ?> = fdv53
Enter new 6 char. name of fg 310 ?> = fdv54
Enter new 6 char. name of fg 311 ?> = fdv55
Enter new 6 char. name of fg 312 ?> = fdv56
Enter new 6 char. name of fg 313 ?> = fdv57
Enter new 6 char. name of fg 314 ?> = fdv58
Enter new 6 char. name of fg 315 ?> = fdv59
Enter new 6 char. name of fg 316 ?> = fdv60
Enter new 6 char. name of fg 317 ?> = fdv61
Enter new 6 char. name of fg 318 ?> = fdv62
Enter new 6 char. name of fg 319 ?> = fdv63
Enter new 6 char. name of fg 320 ?> = fdv64
Enter new 6 char. name of fg 321 ?> = fdv65
Enter new 6 char. name of fg 322 ?> = fdv66
Enter new 6 char. name of fg 323 ?> = fdv67
Enter new 6 char. name of fg 324 ?> = fdv68
Enter new 6 char. name of fg 325 ?> = fdv69
Enter new 6 char. name of fg 326 ?> = fdv70
Enter new 6 char. name of fg 327 ?> = fdv71
Enter new 6 char. name of fg 328 ?> = fdv72
Enter new 6 char. name of fg 329 ?> = fdv73
Enter new 6 char. name of fg 330 ?> = fdv74
Enter new 6 char. name of fg 331 ?> = fdv75
Enter new 6 char. name of fg 332 ?> = fdv76
Enter new 6 char. name of fg 333 ?> = fdv77
Enter new 6 char. name of fg 334 ?> = fdv78
Enter new 6 char. name of fg 335 ?> = fdv79
Enter new 6 char. name of fg 336 ?> = fdv80
Enter new 6 char. name of fg 337 ?> = fdv81
Enter new 6 char. name of fg 338 ?> = fdv82
Enter new 6 char. name of fg 339 ?> = fdv83
Enter new 6 char. name of fg 340 ?> = fdv84
Enter new 6 char. name of fg 341 ?> = fdv85
Enter new 6 char. name of fg 342 ?> = fdv86
Enter new 6 char. name of fg 343 ?> = fdv87
Enter new 6 char. name of fg 344 ?> = fdv88

Enter new 6 char. name of fg 345 ?> = fdv89
 Enter new 6 char. name of fg 346 ?> = fdv90
 Enter new 6 char. name of fg 347 ?> = fdv91
 Enter new 6 char. name of fg 348 ?> = fdv92
 Enter new 6 char. name of fg 349 ?> = fdv93
 Enter new 6 char. name of fg 350 ?> = fdv94
 Enter new 6 char. name of fg 351 ?> = fdv95
 Enter new 6 char. name of fg 352 ?> = fdv96
 Enter new 6 char. name of fg 353 ?> = fdv97
 Enter new 6 char. name of fg 354 ?> = fdv98
 Enter new 6 char. name of fg 355 ?> = fdv99
 Enter new 6 char. name of fg 356 ?> = fdv100
 Enter new 6 char. name of fg 357 ?> = fdv101
 Enter new 6 char. name of fg 358 ?> = fdv102
 Enter new 6 char. name of fg 359 ?> = fdv103
 Enter new 6 char. name of fg 360 ?> = fdv104
 Enter new 6 char. name of fg 361 ?> = fdv105
 Enter new 6 char. name of fg 362 ?> = fdv106
 Enter new 6 char. name of fg 363 ?> = fdv107
 Enter new 6 char. name of fg 364 ?> = fdv108
 Enter new 6 char. name of fg 365 ?> = fdv109
 Enter new 6 char. name of fg 366 ?> = fdv110
 Enter new 6 char. name of fg 367 ?> = fdv111
 Enter new 6 char. name of fg 368 ?> = fdv112
 Enter new 6 char. name of fg 369 ?> = fdv113
 Enter new 6 char. name of fg 370 ?> = fdv114
 Enter new 6 char. name of fg 371 ?> = fdv115
 Enter new 6 char. name of fg 372 ?> = fdv116
 Enter new 6 char. name of fg 373 ?> = fdv117
 Enter new 6 char. name of fg 374 ?> = fdv118
 Enter new 6 char. name of fg 375 ?> = fdv119
 Enter new 6 char. name of fg 376 ?> = fdv120
 Enter new 6 char. name of fg 377 ?> = fdv121
 Enter new 6 char. name of fg 378 ?> = fdv122
 Enter new 6 char. name of fg 379 ?> = fdv123
 Enter new 6 char. name of fg 380 ?> = fdv124
 Enter new 6 char. name of fg 381 ?> = fdv125
 Enter new 6 char. name of fg 382 ?> = fdv126
 Enter new 6 char. name of fg 383 ?> = fdv127
 Enter new 6 char. name of fg 384 ?> = fx0
 Enter new 6 char. name of fg 385 ?> = fx1
 Enter new 6 char. name of fg 386 ?> = fx2
 Enter new 6 char. name of fg 387 ?> = fx3
 Enter new 6 char. name of fg 388 ?> = fx4
 Enter new 6 char. name of fg 389 ?> = fx5
 Enter new 6 char. name of fg 390 ?> = fx6
 Enter new 6 char. name of fg 391 ?> = fx7
 Enter new 6 char. name of fg 392 ?> = fx8
 Enter new 6 char. name of fg 393 ?> = fx9
 Enter new 6 char. name of fg 394 ?> = fx10

```

Enter new 6 char. name of fg 395 ?> = fx11
Enter new 6 char. name of fg 396 ?> = fx12
Enter new 6 char. name of fg 397 ?> = fx13
Enter new 6 char. name of fg 398 ?> = fx14
Enter new 6 char. name of fg 399 ?> = fx15
Enter new 6 char. name of fg 400 ?> = fx16
Enter new 6 char. name of fg 401 ?> = fx17
Enter new 6 char. name of fg 402 ?> = fx18
Enter new 6 char. name of fg 403 ?> = fx19
Enter new 6 char. name of fg 404 ?> = fx20
Enter new 6 char. name of fg 405 ?> = fx21
Enter new 6 char. name of fg 406 ?> = fx22
Enter new 6 char. name of fg 407 ?> = fx23
Enter new 6 char. name of fg 408 ?> = fx24
Enter new 6 char. name of fg 409 ?> = fx25
Enter new 6 char. name of fg 410 ?> = fx26
Enter new 6 char. name of fg 411 ?> = fx27
Enter new 6 char. name of fg 412 ?> = fx28
Enter new 6 char. name of fg 413 ?> = fx29
Enter new 6 char. name of fg 414 ?> = fx30
Enter new 6 char. name of fg 415 ?> = fx31
Enter new 6 char. name of fg 416 ?> = fx32
Enter new 6 char. name of fg 417 ?> = fx33
Enter new 6 char. name of fg 418 ?> = fx34
Enter new 6 char. name of fg 419 ?> = fx35
Enter new 6 char. name of fg 420 ?> = fx36
Enter new 6 char. name of fg 421 ?> = fdisp
Enter new 6 char. name of fg 422 ?> = fhold
Enter new 6 char. name of fg 423 ?> = fning
Enter new 6 char. name of fg 424 ?> = fabort
Enter new 6 char. name of fg 425 ?> = fstart
Enter new 6 char. name of fg 426 ?> = fstop
Enter new 6 char. name of fg 427 ?> = ftrip0
Enter new 6 char. name of fg 428 ?> = ftrip1
Enter new 6 char. name of fg 429 ?> = ftrip2
Enter new 6 char. name of fg 430 ?> = ftrip3
Enter new 6 char. name of fg 431 ?> = ftrip4
MUPF?> = add 89 in
Enter new 6 char. name of in 0 ?> = bitval
Enter new 6 char. name of in 1 ?> = iloop
Enter new 6 char. name of in 2 ?> = jloop
Enter new 6 char. name of in 3 ?> = kloop
Enter new 6 char. name of in 4 ?> = lloop
Enter new 6 char. name of in 5 ?> = mloop
Enter new 6 char. name of in 6 ?> = nloop
Enter new 6 char. name of in 7 ?> = oloop
Enter new 6 char. name of in 8 ?> = ploop
Enter new 6 char. name of in 9 ?> = index
Enter new 6 char. name of in 10 ?> = trip0
Enter new 6 char. name of in 11 ?> = trip1

```

Enter new 6 char. name of in 12 ?> = trip2
Enter new 6 char. name of in 13 ?> = trip3
Enter new 6 char. name of in 14 ?> = trip4
Enter new 6 char. name of in 15 ?> = trip5
Enter new 6 char. name of in 16 ?> = trip6
Enter new 6 char. name of in 17 ?> = trip7
Enter new 6 char. name of in 18 ?> = nadi
Enter new 6 char. name of in 19 ?> = blkval
Enter new 6 char. name of in 20 ?> = sort
Enter new 6 char. name of in 21 ?> = bval
Enter new 6 char. name of in 22 ?> = hldstp
Enter new 6 char. name of in 23 ?> = errn
Enter new 6 char. name of in 24 ?> = errl
Enter new 6 char. name of in 25 ?> = int0
Enter new 6 char. name of in 26 ?> = int1
Enter new 6 char. name of in 27 ?> = int2
Enter new 6 char. name of in 28 ?> = int3
Enter new 6 char. name of in 29 ?> = int4
Enter new 6 char. name of in 30 ?> = int5
Enter new 6 char. name of in 31 ?> = int6
Enter new 6 char. name of in 32 ?> = int7
Enter new 6 char. name of in 33 ?> = int8
Enter new 6 char. name of in 34 ?> = int9
Enter new 6 char. name of in 35 ?> = int10
Enter new 6 char. name of in 36 ?> = int11
Enter new 6 char. name of in 37 ?> = int12
Enter new 6 char. name of in 38 ?> = int13
Enter new 6 char. name of in 39 ?> = int14
Enter new 6 char. name of in 40 ?> = int15
Enter new 6 char. name of in 41 ?> = int16
Enter new 6 char. name of in 42 ?> = int17
Enter new 6 char. name of in 43 ?> = int18
Enter new 6 char. name of in 44 ?> = int19
Enter new 6 char. name of in 45 ?> = int20
Enter new 6 char. name of in 46 ?> = int21
Enter new 6 char. name of in 47 ?> = int22
Enter new 6 char. name of in 48 ?> = int23
Enter new 6 char. name of in 49 ?> = int24
Enter new 6 char. name of in 50 ?> = int25
Enter new 6 char. name of in 51 ?> = int26
Enter new 6 char. name of in 52 ?> = int27
Enter new 6 char. name of in 53 ?> = int28
Enter new 6 char. name of in 54 ?> = int29
Enter new 6 char. name of in 55 ?> = int30
Enter new 6 char. name of in 56 ?> = int31
Enter new 6 char. name of in 57 ?> = int32
Enter new 6 char. name of in 58 ?> = int33
Enter new 6 char. name of in 59 ?> = int34
Enter new 6 char. name of in 60 ?> = int35
Enter new 6 char. name of in 61 ?> = int36

```

Enter new 6 char. name of in 62 ?> = int37
Enter new 6 char. name of in 63 ?> = int38
Enter new 6 char. name of in 64 ?> = int39
Enter new 6 char. name of in 65 ?> = int40
Enter new 6 char. name of in 66 ?> = int41
Enter new 6 char. name of in 67 ?> = int42
Enter new 6 char. name of in 68 ?> = int43
Enter new 6 char. name of in 69 ?> = int44
Enter new 6 char. name of in 70 ?> = int45
Enter new 6 char. name of in 71 ?> = int46
Enter new 6 char. name of in 72 ?> = int47
Enter new 6 char. name of in 73 ?> = int48
Enter new 6 char. name of in 74 ?> = int49
Enter new 6 char. name of in 75 ?> = int50
Enter new 6 char. name of in 76 ?> = int51
Enter new 6 char. name of in 77 ?> = int52
Enter new 6 char. name of in 78 ?> = int53
Enter new 6 char. name of in 79 ?> = int54
Enter new 6 char. name of in 80 ?> = int55
Enter new 6 char. name of in 81 ?> = int56
Enter new 6 char. name of in 82 ?> = int57
Enter new 6 char. name of in 83 ?> = int58
Enter new 6 char. name of in 84 ?> = int59
Enter new 6 char. name of in 85 ?> = int60
Enter new 6 char. name of in 86 ?> = int61
Enter new 6 char. name of in 87 ?> = int62
Enter new 6 char. name of in 88 ?> = int63
MUPF?> = add 16 pt
Enter new 6 char. name of pt 0 ?> = pepn
Enter new 6 char. name of pt 1 ?> = pdev
Enter new 6 char. name of pt 2 ?> = pai
Enter new 6 char. name of pt 3 ?> = pdin
Enter new 6 char. name of pt 4 ?> = ptrip
Enter new 6 char. name of pt 5 ?> = pdeadb
Enter new 6 char. name of pt 6 ?> = palarm
Enter new 6 char. name of pt 7 ?> = pack
Enter new 6 char. name of pt 8 ?> = pmessg
Enter new 6 char. name of pt 9 ?> = pdsabl
Enter new 6 char. name of pt 10 ?> = pback
Enter new 6 char. name of pt 11 ?> = preps
Enter new 6 char. name of pt 12 ?> = plast
Enter new 6 char. name of pt 13 ?> = pskid
Enter new 6 char. name of pt 14 ?> = pstat
Enter new 6 char. name of pt 15 ?> = pstrng
MUPF?> = add 202 sg
Enter new 6 char. name of sg 0 ?> = epns0
Enter max string length for sg 0 ?> = 10
Enter new 6 char. name of sg 1 ?> = epns1
Enter max string length for sg 1 ?> = 10
Enter new 6 char. name of sg 2 ?> = epns2

```


Enter max string length for sg 2 ?> = 10
 Enter new 6 char. name of sg 3 ?> = epns3
 Enter max string length for sg 3 ?> = 10
 Enter new 6 char. name of sg 4 ?> = epns4
 Enter max string length for sg 4 ?> = 10
 Enter new 6 char. name of sg 5 ?> = epns5
 Enter max string length for sg 5 ?> = 10
 Enter new 6 char. name of sg 6 ?> = epns6
 Enter max string length for sg 6 ?> = 10
 Enter new 6 char. name of sg 7 ?> = epns7
 Enter max string length for sg 7 ?> = 10
 Enter new 6 char. name of sg 8 ?> = epns8
 Enter max string length for sg 8 ?> = 10
 Enter new 6 char. name of sg 9 ?> = epns9
 Enter max string length for sg 9 ?> = 10
 Enter new 6 char. name of sg 10 ?> = epns10
 Enter max string length for sg 10 ?> = 10
 Enter new 6 char. name of sg 11 ?> = epns11
 Enter max string length for sg 11 ?> = 10
 Enter new 6 char. name of sg 12 ?> = epns12
 Enter max string length for sg 12 ?> = 10
 Enter new 6 char. name of sg 13 ?> = epns13
 Enter max string length for sg 13 ?> = 10
 Enter new 6 char. name of sg 14 ?> = epns14
 Enter max string length for sg 14 ?> = 10
 Enter new 6 char. name of sg 15 ?> = epns15
 Enter max string length for sg 15 ?> = 10
 Enter new 6 char. name of sg 16 ?> = epns16
 Enter max string length for sg 16 ?> = 10
 Enter new 6 char. name of sg 17 ?> = epns17
 Enter max string length for sg 17 ?> = 10
 Enter new 6 char. name of sg 18 ?> = epns18
 Enter max string length for sg 18 ?> = 10
 Enter new 6 char. name of sg 19 ?> = epns19
 Enter max string length for sg 19 ?> = 10
 Enter new 6 char. name of sg 20 ?> = epns20
 Enter max string length for sg 20 ?> = 10
 Enter new 6 char. name of sg 21 ?> = epns21
 Enter max string length for sg 21 ?> = 10
 Enter new 6 char. name of sg 22 ?> = epns22
 Enter max string length for sg 22 ?> = 10
 Enter new 6 char. name of sg 23 ?> = epns23
 Enter max string length for sg 23 ?> = 10
 Enter new 6 char. name of sg 24 ?> = epns24
 Enter max string length for sg 24 ?> = 10
 Enter new 6 char. name of sg 25 ?> = epns25
 Enter max string length for sg 25 ?> = 10
 Enter new 6 char. name of sg 26 ?> = epns26
 Enter max string length for sg 26 ?> = 10
 Enter new 6 char. name of sg 27 ?> = epns27

Enter max string length for sg 27 ?> = 10
 Enter new 6 char. name of sg 28 ?> = epns28
 Enter max string length for sg 28 ?> = 10
 Enter new 6 char. name of sg 29 ?> = epns29
 Enter max string length for sg 29 ?> = 10
 Enter new 6 char. name of sg 30 ?> = epns30
 Enter max string length for sg 30 ?> = 10
 Enter new 6 char. name of sg 31 ?> = epns31
 Enter max string length for sg 31 ?> = 10
 Enter new 6 char. name of sg 32 ?> = epns32
 Enter max string length for sg 32 ?> = 10
 Enter new 6 char. name of sg 33 ?> = epns33
 Enter max string length for sg 33 ?> = 10
 Enter new 6 char. name of sg 34 ?> = epns34
 Enter max string length for sg 34 ?> = 10
 Enter new 6 char. name of sg 35 ?> = epns35
 Enter max string length for sg 35 ?> = 10
 Enter new 6 char. name of sg 36 ?> = epns36
 Enter max string length for sg 36 ?> = 10
 Enter new 6 char. name of sg 37 ?> = epns37
 Enter max string length for sg 37 ?> = 10
 Enter new 6 char. name of sg 38 ?> = epns38
 Enter max string length for sg 38 ?> = 10
 Enter new 6 char. name of sg 39 ?> = epns39
 Enter max string length for sg 39 ?> = 10
 Enter new 6 char. name of sg 40 ?> = epns40
 Enter max string length for sg 40 ?> = 10
 Enter new 6 char. name of sg 41 ?> = epns41
 Enter max string length for sg 41 ?> = 10
 Enter new 6 char. name of sg 42 ?> = epns42
 Enter max string length for sg 42 ?> = 10
 Enter new 6 char. name of sg 43 ?> = epns43
 Enter max string length for sg 43 ?> = 10
 Enter new 6 char. name of sg 44 ?> = epns44
 Enter max string length for sg 44 ?> = 10
 Enter new 6 char. name of sg 45 ?> = epns45
 Enter max string length for sg 45 ?> = 10
 Enter new 6 char. name of sg 46 ?> = epns46
 Enter max string length for sg 46 ?> = 10
 Enter new 6 char. name of sg 47 ?> = epns47
 Enter max string length for sg 47 ?> = 10
 Enter new 6 char. name of sg 48 ?> = epns48
 Enter max string length for sg 48 ?> = 10
 Enter new 6 char. name of sg 49 ?> = epns49
 Enter max string length for sg 49 ?> = 10
 Enter new 6 char. name of sg 50 ?> = epns50
 Enter max string length for sg 50 ?> = 10
 Enter new 6 char. name of sg 51 ?> = epns51
 Enter max string length for sg 51 ?> = 10
 Enter new 6 char. name of sg 52 ?> = epns52

Enter max string length for sg 52 ?> = 10
 Enter new 6 char. name of sg 53 ?> = epns53
 Enter max string length for sg 53 ?> = 10
 Enter new 6 char. name of sg 54 ?> = epns54
 Enter max string length for sg 54 ?> = 10
 Enter new 6 char. name of sg 55 ?> = epns55
 Enter max string length for sg 55 ?> = 10
 Enter new 6 char. name of sg 56 ?> = epns56
 Enter max string length for sg 56 ?> = 10
 Enter new 6 char. name of sg 57 ?> = epns57
 Enter max string length for sg 57 ?> = 10
 Enter new 6 char. name of sg 58 ?> = epns58
 Enter max string length for sg 58 ?> = 10
 Enter new 6 char. name of sg 59 ?> = epns59
 Enter max string length for sg 59 ?> = 10
 Enter new 6 char. name of sg 60 ?> = epns60
 Enter max string length for sg 60 ?> = 10
 Enter new 6 char. name of sg 61 ?> = epns61
 Enter max string length for sg 61 ?> = 10
 Enter new 6 char. name of sg 62 ?> = epns62
 Enter max string length for sg 62 ?> = 10
 Enter new 6 char. name of sg 63 ?> = epns63
 Enter max string length for sg 63 ?> = 10
 Enter new 6 char. name of sg 64 ?> = epns64
 Enter max string length for sg 64 ?> = 10
 Enter new 6 char. name of sg 65 ?> = epns65
 Enter max string length for sg 65 ?> = 10
 Enter new 6 char. name of sg 66 ?> = epns66
 Enter max string length for sg 66 ?> = 10
 Enter new 6 char. name of sg 67 ?> = epns67
 Enter max string length for sg 67 ?> = 10
 Enter new 6 char. name of sg 68 ?> = epns68
 Enter max string length for sg 68 ?> = 10
 Enter new 6 char. name of sg 69 ?> = epns69
 Enter max string length for sg 69 ?> = 10
 Enter new 6 char. name of sg 70 ?> = epns70
 Enter max string length for sg 70 ?> = 10
 Enter new 6 char. name of sg 71 ?> = epns71
 Enter max string length for sg 71 ?> = 10
 Enter new 6 char. name of sg 72 ?> = epns72
 Enter max string length for sg 72 ?> = 10
 Enter new 6 char. name of sg 73 ?> = epns73
 Enter max string length for sg 73 ?> = 10
 Enter new 6 char. name of sg 74 ?> = epns74
 Enter max string length for sg 74 ?> = 10
 Enter new 6 char. name of sg 75 ?> = epns75
 Enter max string length for sg 75 ?> = 10
 Enter new 6 char. name of sg 76 ?> = epns76
 Enter max string length for sg 76 ?> = 10
 Enter new 6 char. name of sg 77 ?> = epns77

Enter max string length for sg 77 ?> = 10
Enter new 6 char. name of sg 78 ?> = epns78
Enter max string length for sg 78 ?> = 10
Enter new 6 char. name of sg 79 ?> = epns79
Enter max string length for sg 79 ?> = 10
Enter new 6 char. name of sg 80 ?> = epns80
Enter max string length for sg 80 ?> = 10
Enter new 6 char. name of sg 81 ?> = epns81
Enter max string length for sg 81 ?> = 10
Enter new 6 char. name of sg 82 ?> = epns82
Enter max string length for sg 82 ?> = 10
Enter new 6 char. name of sg 83 ?> = epns83
Enter max string length for sg 83 ?> = 10
Enter new 6 char. name of sg 84 ?> = epns84
Enter max string length for sg 84 ?> = 10
Enter new 6 char. name of sg 85 ?> = epns85
Enter max string length for sg 85 ?> = 10
Enter new 6 char. name of sg 86 ?> = epns86
Enter max string length for sg 86 ?> = 10
Enter new 6 char. name of sg 87 ?> = epns87
Enter max string length for sg 87 ?> = 10
Enter new 6 char. name of sg 88 ?> = epns88
Enter max string length for sg 88 ?> = 10
Enter new 6 char. name of sg 89 ?> = epns89
Enter max string length for sg 89 ?> = 10
Enter new 6 char. name of sg 90 ?> = epns90
Enter max string length for sg 90 ?> = 10
Enter new 6 char. name of sg 91 ?> = epns91
Enter max string length for sg 91 ?> = 10
Enter new 6 char. name of sg 92 ?> = epns92
Enter max string length for sg 92 ?> = 10
Enter new 6 char. name of sg 93 ?> = epns93
Enter max string length for sg 93 ?> = 10
Enter new 6 char. name of sg 94 ?> = epns94
Enter max string length for sg 94 ?> = 10
Enter new 6 char. name of sg 95 ?> = epns95
Enter max string length for sg 95 ?> = 10
Enter new 6 char. name of sg 96 ?> = epns96
Enter max string length for sg 96 ?> = 10
Enter new 6 char. name of sg 97 ?> = epns97
Enter max string length for sg 97 ?> = 10
Enter new 6 char. name of sg 98 ?> = epns98
Enter max string length for sg 98 ?> = 10
Enter new 6 char. name of sg 99 ?> = epns99
Enter max string length for sg 99 ?> = 10
Enter new 6 char. name of sg 100 ?> = lsg0
Enter max string length for sg 100 ?> = 30
Enter new 6 char. name of sg 101 ?> = lsg1
Enter max string length for sg 101 ?> = 30
Enter new 6 char. name of sg 102 ?> = lsg2

Enter max string length for sg 102 ?> = 30
Enter new 6 char. name of sg 103 ?> = lsg3
Enter max string length for sg 103 ?> = 30
Enter new 6 char. name of sg 104 ?> = lsg4
Enter max string length for sg 104 ?> = 30
Enter new 6 char. name of sg 105 ?> = lsg5
Enter max string length for sg 105 ?> = 30
Enter new 6 char. name of sg 106 ?> = lsg6
Enter max string length for sg 106 ?> = 30
Enter new 6 char. name of sg 107 ?> = lsg7
Enter max string length for sg 107 ?> = 30
Enter new 6 char. name of sg 108 ?> = lsg8
Enter max string length for sg 108 ?> = 30
Enter new 6 char. name of sg 109 ?> = lsg9
Enter max string length for sg 109 ?> = 30
Enter new 6 char. name of sg 110 ?> = lsg10
Enter max string length for sg 110 ?> = 30
Enter new 6 char. name of sg 111 ?> = lsg11
Enter max string length for sg 111 ?> = 30
Enter new 6 char. name of sg 112 ?> = lsg12
Enter max string length for sg 112 ?> = 30
Enter new 6 char. name of sg 113 ?> = lsg13
Enter max string length for sg 113 ?> = 30
Enter new 6 char. name of sg 114 ?> = lsg14
Enter max string length for sg 114 ?> = 30
Enter new 6 char. name of sg 115 ?> = lsg15
Enter max string length for sg 115 ?> = 30
Enter new 6 char. name of sg 116 ?> = lsg16
Enter max string length for sg 116 ?> = 30
Enter new 6 char. name of sg 117 ?> = lsg17
Enter max string length for sg 117 ?> = 30
Enter new 6 char. name of sg 118 ?> = lsg18
Enter max string length for sg 118 ?> = 30
Enter new 6 char. name of sg 119 ?> = lsg19
Enter max string length for sg 119 ?> = 30
Enter new 6 char. name of sg 120 ?> = lsg20
Enter max string length for sg 120 ?> = 30
Enter new 6 char. name of sg 121 ?> = lsg21
Enter max string length for sg 121 ?> = 30
Enter new 6 char. name of sg 122 ?> = lsg22
Enter max string length for sg 122 ?> = 30
Enter new 6 char. name of sg 123 ?> = lsg23
Enter max string length for sg 123 ?> = 30
Enter new 6 char. name of sg 124 ?> = lsg24
Enter max string length for sg 124 ?> = 30
Enter new 6 char. name of sg 125 ?> = lsg25
Enter max string length for sg 125 ?> = 30
Enter new 6 char. name of sg 126 ?> = lsg26
Enter max string length for sg 126 ?> = 30
Enter new 6 char. name of sg 127 ?> = lsg27

Enter max string length for sg 127 ?> = 30
 Enter new 6 char. name of sg 128 ?> = lsg28
 Enter max string length for sg 128 ?> = 30
 Enter new 6 char. name of sg 129 ?> = lsg29
 Enter max string length for sg 129 ?> = 30
 Enter new 6 char. name of sg 130 ?> = lsg30
 Enter max string length for sg 130 ?> = 30
 Enter new 6 char. name of sg 131 ?> = lsg31
 Enter max string length for sg 131 ?> = 30
 Enter new 6 char. name of sg 132 ?> = lsg32
 Enter max string length for sg 132 ?> = 30
 Enter new 6 char. name of sg 133 ?> = lsg33
 Enter max string length for sg 133 ?> = 30
 Enter new 6 char. name of sg 134 ?> = lsg34
 Enter max string length for sg 134 ?> = 30
 Enter new 6 char. name of sg 135 ?> = lsg35
 Enter max string length for sg 135 ?> = 30
 Enter new 6 char. name of sg 136 ?> = lsg36
 Enter max string length for sg 136 ?> = 30
 Enter new 6 char. name of sg 137 ?> = lsg37
 Enter max string length for sg 137 ?> = 30
 Enter new 6 char. name of sg 138 ?> = lsg38
 Enter max string length for sg 138 ?> = 30
 Enter new 6 char. name of sg 139 ?> = lsg39
 Enter max string length for sg 139 ?> = 30
 Enter new 6 char. name of sg 140 ?> = lsg40
 Enter max string length for sg 140 ?> = 30
 Enter new 6 char. name of sg 141 ?> = lsg41
 Enter max string length for sg 141 ?> = 30
 Enter new 6 char. name of sg 142 ?> = lsg42
 Enter max string length for sg 142 ?> = 30
 Enter new 6 char. name of sg 143 ?> = lsg43
 Enter max string length for sg 143 ?> = 30
 Enter new 6 char. name of sg 144 ?> = lsg44
 Enter max string length for sg 144 ?> = 30
 Enter new 6 char. name of sg 145 ?> = lsg45
 Enter max string length for sg 145 ?> = 30
 Enter new 6 char. name of sg 146 ?> = lsg46
 Enter max string length for sg 146 ?> = 30
 Enter new 6 char. name of sg 147 ?> = lsg47
 Enter max string length for sg 147 ?> = 30
 Enter new 6 char. name of sg 148 ?> = lsg48
 Enter max string length for sg 148 ?> = 30
 Enter new 6 char. name of sg 149 ?> = lsg49
 Enter max string length for sg 149 ?> = 30
 Enter new 6 char. name of sg 150 ?> = lsg50
 Enter max string length for sg 150 ?> = 30
 Enter new 6 char. name of sg 151 ?> = lsg51
 Enter max string length for sg 151 ?> = 30
 Enter new 6 char. name of sg 152 ?> = lsg52

Enter max string length for sg 152 ?> = 30
Enter new 6 char. name of sg 153 ?> = lsg53
Enter max string length for sg 153 ?> = 30
Enter new 6 char. name of sg 154 ?> = lsg54
Enter max string length for sg 154 ?> = 30
Enter new 6 char. name of sg 155 ?> = lsg55
Enter max string length for sg 155 ?> = 30
Enter new 6 char. name of sg 156 ?> = lsg56
Enter max string length for sg 156 ?> = 30
Enter new 6 char. name of sg 157 ?> = lsg57
Enter max string length for sg 157 ?> = 30
Enter new 6 char. name of sg 158 ?> = lsg58
Enter max string length for sg 158 ?> = 30
Enter new 6 char. name of sg 159 ?> = lsg59
Enter max string length for sg 159 ?> = 30
Enter new 6 char. name of sg 160 ?> = lsg60
Enter max string length for sg 160 ?> = 30
Enter new 6 char. name of sg 161 ?> = lsg61
Enter max string length for sg 161 ?> = 30
Enter new 6 char. name of sg 162 ?> = lsg62
Enter max string length for sg 162 ?> = 30
Enter new 6 char. name of sg 163 ?> = lsg63
Enter max string length for sg 163 ?> = 30
Enter new 6 char. name of sg 164 ?> = lsg64
Enter max string length for sg 164 ?> = 30
Enter new 6 char. name of sg 165 ?> = lsg65
Enter max string length for sg 165 ?> = 30
Enter new 6 char. name of sg 166 ?> = lsg66
Enter max string length for sg 166 ?> = 30
Enter new 6 char. name of sg 167 ?> = lsg67
Enter max string length for sg 167 ?> = 30
Enter new 6 char. name of sg 168 ?> = lsg68
Enter max string length for sg 168 ?> = 30
Enter new 6 char. name of sg 169 ?> = lsg69
Enter max string length for sg 169 ?> = 30
Enter new 6 char. name of sg 170 ?> = lsg70
Enter max string length for sg 170 ?> = 30
Enter new 6 char. name of sg 171 ?> = lsg71
Enter max string length for sg 171 ?> = 30
Enter new 6 char. name of sg 172 ?> = lsg72
Enter max string length for sg 172 ?> = 30
Enter new 6 char. name of sg 173 ?> = lsg73
Enter max string length for sg 173 ?> = 30
Enter new 6 char. name of sg 174 ?> = lsg74
Enter max string length for sg 174 ?> = 30
Enter new 6 char. name of sg 175 ?> = lsg75
Enter max string length for sg 175 ?> = 30
Enter new 6 char. name of sg 176 ?> = lsg76
Enter max string length for sg 176 ?> = 30
Enter new 6 char. name of sg 177 ?> = lsg77

```

Enter max string length for sg 177 ?> = 30
Enter new 6 char. name of sg 178 ?> = lsg78
Enter max string length for sg 178 ?> = 30
Enter new 6 char. name of sg 179 ?> = lsg79
Enter max string length for sg 179 ?> = 30
Enter new 6 char. name of sg 180 ?> = lsg80
Enter max string length for sg 180 ?> = 30
Enter new 6 char. name of sg 181 ?> = lsg81
Enter max string length for sg 181 ?> = 30
Enter new 6 char. name of sg 182 ?> = lsg82
Enter max string length for sg 182 ?> = 30
Enter new 6 char. name of sg 183 ?> = lsg83
Enter max string length for sg 183 ?> = 30
Enter new 6 char. name of sg 184 ?> = lsg84
Enter max string length for sg 184 ?> = 30
Enter new 6 char. name of sg 185 ?> = lsg85
Enter max string length for sg 185 ?> = 30
Enter new 6 char. name of sg 186 ?> = lsg86
Enter max string length for sg 186 ?> = 30
Enter new 6 char. name of sg 187 ?> = lsg87
Enter max string length for sg 187 ?> = 30
Enter new 6 char. name of sg 188 ?> = lsg88
Enter max string length for sg 188 ?> = 30
Enter new 6 char. name of sg 189 ?> = lsg89
Enter max string length for sg 189 ?> = 30
Enter new 6 char. name of sg 190 ?> = lsg90
Enter max string length for sg 190 ?> = 30
Enter new 6 char. name of sg 191 ?> = lsg91
Enter max string length for sg 191 ?> = 30
Enter new 6 char. name of sg 192 ?> = lsg92
Enter max string length for sg 192 ?> = 30
Enter new 6 char. name of sg 193 ?> = lsg93
Enter max string length for sg 193 ?> = 30
Enter new 6 char. name of sg 194 ?> = lsg94
Enter max string length for sg 194 ?> = 30
Enter new 6 char. name of sg 195 ?> = lsg95
Enter max string length for sg 195 ?> = 30
Enter new 6 char. name of sg 196 ?> = lsg96
Enter max string length for sg 196 ?> = 30
Enter new 6 char. name of sg 197 ?> = lsg97
Enter max string length for sg 197 ?> = 30
Enter new 6 char. name of sg 198 ?> = lsg98
Enter max string length for sg 198 ?> = 30
Enter new 6 char. name of sg 199 ?> = lsg99
Enter max string length for sg 199 ?> = 30
Enter new 6 char. name of sg 200 ?> = time0
Enter max string length for sg 200 ?> = 10
Enter new 6 char. name of sg 201 ?> = date0
Enter max string length for sg 201 ?> = 10
MUPF?> = add 5 st

```


WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of st 0 ?> = rcptmr
Enter new 6 char. name of st 1 ?> = vltmr0
Enter new 6 char. name of st 2 ?> = vltmr1
Enter new 6 char. name of st 3 ?> = vltmr2
Enter new 6 char. name of st 4 ?> = vltmr3
MUPF?> = add 5 mt
Enter new 6 char. name of mt 0 ?> = dtmr0
Enter new 6 char. name of mt 1 ?> = dtmr1
Enter new 6 char. name of mt 2 ?> = dtmr2
Enter new 6 char. name of mt 3 ?> = dtmr3
Enter new 6 char. name of mt 4 ?> = dtmr4
MUPF?> =
UPF?>
end of file encountered

SHORT0

```

UPF Version 6.3-2      05-Dec-94 09:35:38
UPF?> = de short0
UPF?> = cr short0,*
Enter 4 character name of RPF to associate?> = farm
MUPF?> = add 208 fg
Enter new 6 char. name of fg 0 ?> = faln0
Enter new 6 char. name of fg 1 ?> = faln1
Enter new 6 char. name of fg 2 ?> = faln2
Enter new 6 char. name of fg 3 ?> = faln3
Enter new 6 char. name of fg 4 ?> = faln4
Enter new 6 char. name of fg 5 ?> = faln5
Enter new 6 char. name of fg 6 ?> = faln6
Enter new 6 char. name of fg 7 ?> = faln7
Enter new 6 char. name of fg 8 ?> = faln8
Enter new 6 char. name of fg 9 ?> = faln9
Enter new 6 char. name of fg 10 ?> = faln10
Enter new 6 char. name of fg 11 ?> = faln11
Enter new 6 char. name of fg 12 ?> = faln12
Enter new 6 char. name of fg 13 ?> = faln13
Enter new 6 char. name of fg 14 ?> = faln14
Enter new 6 char. name of fg 15 ?> = faln15
Enter new 6 char. name of fg 16 ?> = faln16
Enter new 6 char. name of fg 17 ?> = faln17
Enter new 6 char. name of fg 18 ?> = faln18
Enter new 6 char. name of fg 19 ?> = faln19
Enter new 6 char. name of fg 20 ?> = faln20
Enter new 6 char. name of fg 21 ?> = faln21
Enter new 6 char. name of fg 22 ?> = faln22
Enter new 6 char. name of fg 23 ?> = faln23
Enter new 6 char. name of fg 24 ?> = faln24
Enter new 6 char. name of fg 25 ?> = faln25
Enter new 6 char. name of fg 26 ?> = faln26
Enter new 6 char. name of fg 27 ?> = faln27
Enter new 6 char. name of fg 28 ?> = faln28
Enter new 6 char. name of fg 29 ?> = faln29
Enter new 6 char. name of fg 30 ?> = faln30
Enter new 6 char. name of fg 31 ?> = faln31
Enter new 6 char. name of fg 32 ?> = faln32
Enter new 6 char. name of fg 33 ?> = faln33
Enter new 6 char. name of fg 34 ?> = faln34
Enter new 6 char. name of fg 35 ?> = faln35
Enter new 6 char. name of fg 36 ?> = faln36
Enter new 6 char. name of fg 37 ?> = faln37
Enter new 6 char. name of fg 38 ?> = faln38
Enter new 6 char. name of fg 39 ?> = faln39
Enter new 6 char. name of fg 40 ?> = fpas0
Enter new 6 char. name of fg 41 ?> = fpas1
Enter new 6 char. name of fg 42 ?> = fpas2
Enter new 6 char. name of fg 43 ?> = fpas3
Enter new 6 char. name of fg 44 ?> = fpas4

```

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fg 45 ?> = fpas5
 Enter new 6 char. name of fg 46 ?> = fpas6
 Enter new 6 char. name of fg 47 ?> = fpas7
 Enter new 6 char. name of fg 48 ?> = fpas8
 Enter new 6 char. name of fg 49 ?> = fpas9
 Enter new 6 char. name of fg 50 ?> = fpas10
 Enter new 6 char. name of fg 51 ?> = fpas11
 Enter new 6 char. name of fg 52 ?> = fpas12
 Enter new 6 char. name of fg 53 ?> = fpas13
 Enter new 6 char. name of fg 54 ?> = fpas14
 Enter new 6 char. name of fg 55 ?> = fpas15
 Enter new 6 char. name of fg 56 ?> = fpas16
 Enter new 6 char. name of fg 57 ?> = fpas17
 Enter new 6 char. name of fg 58 ?> = fpas18
 Enter new 6 char. name of fg 59 ?> = fpas19
 Enter new 6 char. name of fg 60 ?> = fpas20
 Enter new 6 char. name of fg 61 ?> = fpas21
 Enter new 6 char. name of fg 62 ?> = fpas22
 Enter new 6 char. name of fg 63 ?> = fpas23
 Enter new 6 char. name of fg 64 ?> = fpas24
 Enter new 6 char. name of fg 65 ?> = fpas25
 Enter new 6 char. name of fg 66 ?> = fpas26
 Enter new 6 char. name of fg 67 ?> = fpas27
 Enter new 6 char. name of fg 68 ?> = fpas28
 Enter new 6 char. name of fg 69 ?> = fpas29
 Enter new 6 char. name of fg 70 ?> = fpas30
 Enter new 6 char. name of fg 71 ?> = fpas31
 Enter new 6 char. name of fg 72 ?> = fpas32
 Enter new 6 char. name of fg 73 ?> = fpas33
 Enter new 6 char. name of fg 74 ?> = fpas34
 Enter new 6 char. name of fg 75 ?> = fpas35
 Enter new 6 char. name of fg 76 ?> = fpas36
 Enter new 6 char. name of fg 77 ?> = fpas37
 Enter new 6 char. name of fg 78 ?> = fpas38
 Enter new 6 char. name of fg 79 ?> = fpas39
 Enter new 6 char. name of fg 80 ?> = fdis0
 Enter new 6 char. name of fg 81 ?> = fdis1
 Enter new 6 char. name of fg 82 ?> = fdis2
 Enter new 6 char. name of fg 83 ?> = fdis3
 Enter new 6 char. name of fg 84 ?> = fdis4
 Enter new 6 char. name of fg 85 ?> = fdis5
 Enter new 6 char. name of fg 86 ?> = fdis6
 Enter new 6 char. name of fg 87 ?> = fdis7
 Enter new 6 char. name of fg 88 ?> = fdis8
 Enter new 6 char. name of fg 89 ?> = fdis9
 Enter new 6 char. name of fg 90 ?> = fdis10
 Enter new 6 char. name of fg 91 ?> = fdis11
 Enter new 6 char. name of fg 92 ?> = fdis12
 Enter new 6 char. name of fg 93 ?> = fdis13
 Enter new 6 char. name of fg 94 ?> = fdis14

Enter new 6 char. name of fg 95 ?> = fdis15
Enter new 6 char. name of fg 96 ?> = fdis16
Enter new 6 char. name of fg 97 ?> = fdis17
Enter new 6 char. name of fg 98 ?> = fdis18
Enter new 6 char. name of fg 99 ?> = fdis19
Enter new 6 char. name of fg 100 ?> = fdis20
Enter new 6 char. name of fg 101 ?> = fdis21
Enter new 6 char. name of fg 102 ?> = fdis22
Enter new 6 char. name of fg 103 ?> = fdis23
Enter new 6 char. name of fg 104 ?> = fdis24
Enter new 6 char. name of fg 105 ?> = fdis25
Enter new 6 char. name of fg 106 ?> = fdis26
Enter new 6 char. name of fg 107 ?> = fdis27
Enter new 6 char. name of fg 108 ?> = fdis28
Enter new 6 char. name of fg 109 ?> = fdis29
Enter new 6 char. name of fg 110 ?> = fdis30
Enter new 6 char. name of fg 111 ?> = fdis31
Enter new 6 char. name of fg 112 ?> = fdis32
Enter new 6 char. name of fg 113 ?> = fdis33
Enter new 6 char. name of fg 114 ?> = fdis34
Enter new 6 char. name of fg 115 ?> = fdis35
Enter new 6 char. name of fg 116 ?> = fdis36
Enter new 6 char. name of fg 117 ?> = fdis37
Enter new 6 char. name of fg 118 ?> = fdis38
Enter new 6 char. name of fg 119 ?> = fdis39
Enter new 6 char. name of fg 120 ?> = frv0
Enter new 6 char. name of fg 121 ?> = frv1
Enter new 6 char. name of fg 122 ?> = frv2
Enter new 6 char. name of fg 123 ?> = frv3
Enter new 6 char. name of fg 124 ?> = frv4
Enter new 6 char. name of fg 125 ?> = frv5
Enter new 6 char. name of fg 126 ?> = frv6
Enter new 6 char. name of fg 127 ?> = frv7
Enter new 6 char. name of fg 128 ?> = frv8
Enter new 6 char. name of fg 129 ?> = frv9
Enter new 6 char. name of fg 130 ?> = frv10
Enter new 6 char. name of fg 131 ?> = frv11
Enter new 6 char. name of fg 132 ?> = frv12
Enter new 6 char. name of fg 133 ?> = frv13
Enter new 6 char. name of fg 134 ?> = frv14
Enter new 6 char. name of fg 135 ?> = frv15
Enter new 6 char. name of fg 136 ?> = frv16
Enter new 6 char. name of fg 137 ?> = frv17
Enter new 6 char. name of fg 138 ?> = frv18
Enter new 6 char. name of fg 139 ?> = frv19
Enter new 6 char. name of fg 140 ?> = frv20
Enter new 6 char. name of fg 141 ?> = frv21
Enter new 6 char. name of fg 142 ?> = frv22
Enter new 6 char. name of fg 143 ?> = frv23
Enter new 6 char. name of fg 144 ?> = frv24

Enter new 6 char. name of fg 145 ?> = frv25
Enter new 6 char. name of fg 146 ?> = frv26
Enter new 6 char. name of fg 147 ?> = frv27
Enter new 6 char. name of fg 148 ?> = frv28
Enter new 6 char. name of fg 149 ?> = frv29
Enter new 6 char. name of fg 150 ?> = frv30
Enter new 6 char. name of fg 151 ?> = frv31
Enter new 6 char. name of fg 152 ?> = frv32
Enter new 6 char. name of fg 153 ?> = frv33
Enter new 6 char. name of fg 154 ?> = frv34
Enter new 6 char. name of fg 155 ?> = frv35
Enter new 6 char. name of fg 156 ?> = frv36
Enter new 6 char. name of fg 157 ?> = frv37
Enter new 6 char. name of fg 158 ?> = frv38
Enter new 6 char. name of fg 159 ?> = frv39
Enter new 6 char. name of fg 160 ?> = fleak0
Enter new 6 char. name of fg 161 ?> = fleak1
Enter new 6 char. name of fg 162 ?> = fleak2
Enter new 6 char. name of fg 163 ?> = fleak3
Enter new 6 char. name of fg 164 ?> = fleak4
Enter new 6 char. name of fg 165 ?> = fleak5
Enter new 6 char. name of fg 166 ?> = fleak6
Enter new 6 char. name of fg 167 ?> = fleak7
Enter new 6 char. name of fg 168 ?> = fsn0
Enter new 6 char. name of fg 169 ?> = fsn1
Enter new 6 char. name of fg 170 ?> = fsn2
Enter new 6 char. name of fg 171 ?> = fsn3
Enter new 6 char. name of fg 172 ?> = fsn4
Enter new 6 char. name of fg 173 ?> = fsn5
Enter new 6 char. name of fg 174 ?> = fsn6
Enter new 6 char. name of fg 175 ?> = fsn7
Enter new 6 char. name of fg 176 ?> = fs10
Enter new 6 char. name of fg 177 ?> = fs11
Enter new 6 char. name of fg 178 ?> = fs12
Enter new 6 char. name of fg 179 ?> = fs13
Enter new 6 char. name of fg 180 ?> = fs14
Enter new 6 char. name of fg 181 ?> = fs15
Enter new 6 char. name of fg 182 ?> = fs16
Enter new 6 char. name of fg 183 ?> = fs17
Enter new 6 char. name of fg 184 ?> = ffarm
Enter new 6 char. name of fg 185 ?> = fspit
Enter new 6 char. name of fg 186 ?> = fvlpit
Enter new 6 char. name of fg 187 ?> = fhvac
Enter new 6 char. name of fg 188 ?> = f271
Enter new 6 char. name of fg 189 ?> = fflush
Enter new 6 char. name of fg 190 ?> = fleak
Enter new 6 char. name of fg 191 ?> = fdisp
Enter new 6 char. name of fg 192 ?> = fhold
Enter new 6 char. name of fg 193 ?> = fning
Enter new 6 char. name of fg 194 ?> = fabort

Enter new 6 char. name of in 36 ?> = int8
 Enter new 6 char. name of in 37 ?> = int9
 Enter new 6 char. name of in 38 ?> = int10
 Enter new 6 char. name of in 39 ?> = int11
 Enter new 6 char. name of in 40 ?> = int12
 Enter new 6 char. name of in 41 ?> = int13
 Enter new 6 char. name of in 42 ?> = int14
 Enter new 6 char. name of in 43 ?> = int15
 Enter new 6 char. name of in 44 ?> = int16
 Enter new 6 char. name of in 45 ?> = int17
 Enter new 6 char. name of in 46 ?> = int18
 Enter new 6 char. name of in 47 ?> = int19
 Enter new 6 char. name of in 48 ?> = int20
 Enter new 6 char. name of in 49 ?> = int21
 Enter new 6 char. name of in 50 ?> = int22
 Enter new 6 char. name of in 51 ?> = int23
 Enter new 6 char. name of in 52 ?> = int24
 Enter new 6 char. name of in 53 ?> = int25
 Enter new 6 char. name of in 54 ?> = int26
 Enter new 6 char. name of in 55 ?> = int27
 Enter new 6 char. name of in 56 ?> = int28
 Enter new 6 char. name of in 57 ?> = int29
 Enter new 6 char. name of in 58 ?> = int30
 Enter new 6 char. name of in 59 ?> = int31
 Enter new 6 char. name of in 60 ?> = int32
 Enter new 6 char. name of in 61 ?> = int33
 Enter new 6 char. name of in 62 ?> = int34
 Enter new 6 char. name of in 63 ?> = int35
 Enter new 6 char. name of in 64 ?> = int36
 Enter new 6 char. name of in 65 ?> = int37
 Enter new 6 char. name of in 66 ?> = int38
 Enter new 6 char. name of in 67 ?> = int39
 Enter new 6 char. name of in 68 ?> = type0
 Enter new 6 char. name of in 69 ?> = type1
 Enter new 6 char. name of in 70 ?> = type2
 Enter new 6 char. name of in 71 ?> = type3
 Enter new 6 char. name of in 72 ?> = type4
 Enter new 6 char. name of in 73 ?> = type5
 Enter new 6 char. name of in 74 ?> = type6
 Enter new 6 char. name of in 75 ?> = type7
 Enter new 6 char. name of in 76 ?> = type8
 Enter new 6 char. name of in 77 ?> = type9
 Enter new 6 char. name of in 78 ?> = type10
 Enter new 6 char. name of in 79 ?> = type11
 Enter new 6 char. name of in 80 ?> = type12
 Enter new 6 char. name of in 81 ?> = type13
 Enter new 6 char. name of in 82 ?> = type14
 Enter new 6 char. name of in 83 ?> = type15
 Enter new 6 char. name of in 84 ?> = type16
 Enter new 6 char. name of in 85 ?> = type17

Enter new 6 char. name of in 86 ?> = type18
Enter new 6 char. name of in 87 ?> = type19
Enter new 6 char. name of in 88 ?> = type20
Enter new 6 char. name of in 89 ?> = type21
Enter new 6 char. name of in 90 ?> = type22
Enter new 6 char. name of in 91 ?> = type23
Enter new 6 char. name of in 92 ?> = type24
Enter new 6 char. name of in 93 ?> = type25
Enter new 6 char. name of in 94 ?> = type26
Enter new 6 char. name of in 95 ?> = type27
Enter new 6 char. name of in 96 ?> = type28
Enter new 6 char. name of in 97 ?> = type29
Enter new 6 char. name of in 98 ?> = type30
Enter new 6 char. name of in 99 ?> = type31
Enter new 6 char. name of in 100 ?> = type32
Enter new 6 char. name of in 101 ?> = type33
Enter new 6 char. name of in 102 ?> = type34
Enter new 6 char. name of in 103 ?> = type35
Enter new 6 char. name of in 104 ?> = type36
Enter new 6 char. name of in 105 ?> = type37
Enter new 6 char. name of in 106 ?> = type38
Enter new 6 char. name of in 107 ?> = type39
Enter new 6 char. name of in 108 ?> = skid0
Enter new 6 char. name of in 109 ?> = skid1
Enter new 6 char. name of in 110 ?> = skid2
Enter new 6 char. name of in 111 ?> = skid3
Enter new 6 char. name of in 112 ?> = skid4
Enter new 6 char. name of in 113 ?> = skid5
Enter new 6 char. name of in 114 ?> = skid6
Enter new 6 char. name of in 115 ?> = skid7
Enter new 6 char. name of in 116 ?> = skid8
Enter new 6 char. name of in 117 ?> = skid9
Enter new 6 char. name of in 118 ?> = skid10
Enter new 6 char. name of in 119 ?> = skid11
Enter new 6 char. name of in 120 ?> = skid12
Enter new 6 char. name of in 121 ?> = skid13
Enter new 6 char. name of in 122 ?> = skid14
Enter new 6 char. name of in 123 ?> = skid15
Enter new 6 char. name of in 124 ?> = skid16
Enter new 6 char. name of in 125 ?> = skid17
Enter new 6 char. name of in 126 ?> = skid18
Enter new 6 char. name of in 127 ?> = skid19
Enter new 6 char. name of in 128 ?> = skid20
Enter new 6 char. name of in 129 ?> = skid21
Enter new 6 char. name of in 130 ?> = skid22
Enter new 6 char. name of in 131 ?> = skid23
Enter new 6 char. name of in 132 ?> = skid24
Enter new 6 char. name of in 133 ?> = skid25
Enter new 6 char. name of in 134 ?> = skid26
Enter new 6 char. name of in 135 ?> = skid27


```

Enter new 6 char. name of in 136 ?> = skid28
Enter new 6 char. name of in 137 ?> = skid29
Enter new 6 char. name of in 138 ?> = skid30
Enter new 6 char. name of in 139 ?> = skid31
Enter new 6 char. name of in 140 ?> = skid32
Enter new 6 char. name of in 141 ?> = skid33
Enter new 6 char. name of in 142 ?> = skid34
Enter new 6 char. name of in 143 ?> = skid35
Enter new 6 char. name of in 144 ?> = skid36
Enter new 6 char. name of in 145 ?> = skid37
Enter new 6 char. name of in 146 ?> = skid38
Enter new 6 char. name of in 147 ?> = skid39
Enter new 6 char. name of in 148 ?> = dev0
Enter new 6 char. name of in 149 ?> = dev1
Enter new 6 char. name of in 150 ?> = dev2
Enter new 6 char. name of in 151 ?> = dev3
Enter new 6 char. name of in 152 ?> = dev4
Enter new 6 char. name of in 153 ?> = dev5
Enter new 6 char. name of in 154 ?> = dev6
Enter new 6 char. name of in 155 ?> = dev7
Enter new 6 char. name of in 156 ?> = dev8
Enter new 6 char. name of in 157 ?> = dev9
Enter new 6 char. name of in 158 ?> = leak0
Enter new 6 char. name of in 159 ?> = leak1
Enter new 6 char. name of in 160 ?> = leak2
Enter new 6 char. name of in 161 ?> = leak3
Enter new 6 char. name of in 162 ?> = leak4
Enter new 6 char. name of in 163 ?> = leak5
Enter new 6 char. name of in 164 ?> = color0
Enter new 6 char. name of in 165 ?> = color1
Enter new 6 char. name of in 166 ?> = color2
Enter new 6 char. name of in 167 ?> = color3
Enter new 6 char. name of in 168 ?> = color4
Enter new 6 char. name of in 169 ?> = grafx0
Enter new 6 char. name of in 170 ?> = grafx1
Enter new 6 char. name of in 171 ?> = grafx2
Enter new 6 char. name of in 172 ?> = grafx3
Enter new 6 char. name of in 173 ?> = grafx4
MUPF?> = add 30 fl
Enter new 6 char. name of fl 0 ?> = float0
Enter new 6 char. name of fl 1 ?> = float1
Enter new 6 char. name of fl 2 ?> = float2
Enter new 6 char. name of fl 3 ?> = float3
Enter new 6 char. name of fl 4 ?> = float4
Enter new 6 char. name of fl 5 ?> = float5
Enter new 6 char. name of fl 6 ?> = float6
Enter new 6 char. name of fl 7 ?> = float7
Enter new 6 char. name of fl 8 ?> = float8
Enter new 6 char. name of fl 9 ?> = float9
Enter new 6 char. name of fl 10 ?> = aio

```

```

Enter new 6 char. name of fl 11 ?> = ai1
Enter new 6 char. name of fl 12 ?> = ai2
Enter new 6 char. name of fl 13 ?> = ai3
Enter new 6 char. name of fl 14 ?> = ai4
Enter new 6 char. name of fl 15 ?> = ai5
Enter new 6 char. name of fl 16 ?> = ai6
Enter new 6 char. name of fl 17 ?> = ai7
Enter new 6 char. name of fl 18 ?> = ai8
Enter new 6 char. name of fl 19 ?> = ai9
Enter new 6 char. name of fl 20 ?> = ai10
Enter new 6 char. name of fl 21 ?> = ai11
Enter new 6 char. name of fl 22 ?> = ai12
Enter new 6 char. name of fl 23 ?> = ai13
Enter new 6 char. name of fl 24 ?> = ai14
Enter new 6 char. name of fl 25 ?> = ai15
Enter new 6 char. name of fl 26 ?> = ai16
Enter new 6 char. name of fl 27 ?> = ai17
Enter new 6 char. name of fl 28 ?> = ai18
Enter new 6 char. name of fl 29 ?> = ai19
MUPF?> = add 10 pt
Enter new 6 char. name of pt 0 ?> = pepn
Enter new 6 char. name of pt 1 ?> = pdin
Enter new 6 char. name of pt 2 ?> = pdev
Enter new 6 char. name of pt 3 ?> = pai
Enter new 6 char. name of pt 4 ?> = palarm
Enter new 6 char. name of pt 5 ?> = pskid
Enter new 6 char. name of pt 6 ?> = pstat
Enter new 6 char. name of pt 7 ?> = ptype
Enter new 6 char. name of pt 8 ?> = ppass
Enter new 6 char. name of pt 9 ?> = prev
MUPF?> = add 35 sg
Enter new 6 char. name of sg 0 ?> = swhen0
Enter max string length for sg 0 ?> = 8
Enter new 6 char. name of sg 1
Enter new 6 char. name of sg 1 ?> = swhen1
Enter max string length for sg 1 ?> = 8
Enter new 6 char. name of sg 2 ?> = swhen2
Enter max string length for sg 2 ?> = 8
Enter new 6 char. name of sg 3 ?> = swhen3
Enter max string length for sg 3 ?> = 8
Enter new 6 char. name of sg 4 ?> = swhen4
Enter max string length for sg 4 ?> = 8
Enter new 6 char. name of sg 5 ?> = ssng0
Enter max string length for sg 5 ?> = 8
Enter new 6 char. name of sg 6 ?> = ssng1
Enter max string length for sg 6 ?> = 8
Enter new 6 char. name of sg 7 ?> = ssng2
Enter max string length for sg 7 ?> = 8
Enter new 6 char. name of sg 8 ?> = ssng3
Enter max string length for sg 8 ?> = 8

```

Enter new 6 char. name of sg 9 ?> = ssng4
 Enter max string length for sg 9 ?> = 8
 Enter new 6 char. name of sg 10 ?> = sdate0
 Enter max string length for sg 10 ?> = 10
 Enter new 6 char. name of sg 11 ?> = sdate1
 Enter max string length for sg 11 ?> = 10
 Enter new 6 char. name of sg 12 ?> = sdate2
 Enter max string length for sg 12 ?> = 10
 Enter new 6 char. name of sg 13 ?> = sdate3
 Enter max string length for sg 13 ?> = 10
 Enter new 6 char. name of sg 14 ?> = sdate4
 Enter max string length for sg 14 ?> = 10
 Enter new 6 char. name of sg 15 ?> = epns0
 Enter max string length for sg 15 ?> = 10
 Enter new 6 char. name of sg 16 ?> = epns1
 Enter max string length for sg 16 ?> = 10
 Enter new 6 char. name of sg 17 ?> = epns2
 Enter max string length for sg 17 ?> = 10
 Enter new 6 char. name of sg 18 ?> = epns3
 Enter max string length for sg 18 ?> = 10
 Enter new 6 char. name of sg 19 ?> = epns4
 Enter max string length for sg 19 ?> = 10
 Enter new 6 char. name of sg 20 ?> = salm0
 Enter max string length for sg 20 ?> = 20
 Enter new 6 char. name of sg 21 ?> = salm1
 Enter max string length for sg 21 ?> = 20
 Enter new 6 char. name of sg 22 ?> = salm2
 Enter max string length for sg 22 ?> = 20
 Enter new 6 char. name of sg 23 ?> = salm3
 Enter max string length for sg 23 ?> = 20
 Enter new 6 char. name of sg 24 ?> = salm4
 Enter max string length for sg 24 ?> = 20
 Enter new 6 char. name of sg 25 ?> = lsg0
 Enter max string length for sg 25 ?> = 30
 Enter new 6 char. name of sg 26 ?> = lsg1
 Enter max string length for sg 26 ?> = 30
 Enter new 6 char. name of sg 27 ?> = lsg2
 Enter max string length for sg 27 ?> = 30
 Enter new 6 char. name of sg 28 ?> = lsg3
 Enter max string length for sg 28 ?> = 30
 Enter new 6 char. name of sg 29 ?> = lsg4
 Enter max string length for sg 29 ?> = 30
 Enter new 6 char. name of sg 30 ?> = strng0
 Enter max string length for sg 30 ?> = 30
 Enter new 6 char. name of sg 31 ?> = strng1
 Enter max string length for sg 31 ?> = 30
 Enter new 6 char. name of sg 32 ?> = strng2
 Enter max string length for sg 32 ?> = 30
 Enter new 6 char. name of sg 33 ?> = strng3
 Enter max string length for sg 33 ?> = 30

WMC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of sg 34 ?> = strng4
Enter max string length for sg 34 ?> = 30
MUPF?> =
UPF?>
end of file encountered

PROCESS

```

UPF Version 6.3-2      05-Dec-94 09:36:16
UPF?> = de prcess
UPF?> = cr prcess,*
Enter 4 character name of RPF to associate?> = 242a
MUPF?> = add 368 fg
Enter new 6 char. name of fg 0 ?> = fal m0
Enter new 6 char. name of fg 1 ?> = fal m1
Enter new 6 char. name of fg 2 ?> = fal m2
Enter new 6 char. name of fg 3 ?> = fal m3
Enter new 6 char. name of fg 4 ?> = fal m4
Enter new 6 char. name of fg 5 ?> = fal m5
Enter new 6 char. name of fg 6 ?> = fal m6
Enter new 6 char. name of fg 7 ?> = fal m7
Enter new 6 char. name of fg 8 ?> = fal m8
Enter new 6 char. name of fg 9 ?> = fal m9
Enter new 6 char. name of fg 10 ?> = fal m10
Enter new 6 char. name of fg 11 ?> = fal m11
Enter new 6 char. name of fg 12 ?> = fal m12
Enter new 6 char. name of fg 13 ?> = fal m13
Enter new 6 char. name of fg 14 ?> = fal m14
Enter new 6 char. name of fg 15 ?> = fal m15
Enter new 6 char. name of fg 16 ?> = fal m16
Enter new 6 char. name of fg 17 ?> = fal m17
Enter new 6 char. name of fg 18 ?> = fal m18
Enter new 6 char. name of fg 19 ?> = fal m19
Enter new 6 char. name of fg 20 ?> = fal m20
Enter new 6 char. name of fg 21 ?> = fal m21
Enter new 6 char. name of fg 22 ?> = fal m22
Enter new 6 char. name of fg 23 ?> = fal m23
Enter new 6 char. name of fg 24 ?> = fal m24
Enter new 6 char. name of fg 25 ?> = fal m25
Enter new 6 char. name of fg 26 ?> = fal m26
Enter new 6 char. name of fg 27 ?> = fal m27
Enter new 6 char. name of fg 28 ?> = fal m28
Enter new 6 char. name of fg 29 ?> = fal m29
Enter new 6 char. name of fg 30 ?> = fal m30
Enter new 6 char. name of fg 31 ?> = fal m31
Enter new 6 char. name of fg 32 ?> = fal m32
Enter new 6 char. name of fg 33 ?> = fal m33
Enter new 6 char. name of fg 34 ?> = fal m34
Enter new 6 char. name of fg 35 ?> = fal m35
Enter new 6 char. name of fg 36 ?> = fal m36
Enter new 6 char. name of fg 37 ?> = fal m37
Enter new 6 char. name of fg 38 ?> = fal m38
Enter new 6 char. name of fg 39 ?> = fal m39
Enter new 6 char. name of fg 40 ?> = fal m40
Enter new 6 char. name of fg 41 ?> = fal m41
Enter new 6 char. name of fg 42 ?> = fal m42
Enter new 6 char. name of fg 43 ?> = fal m43
Enter new 6 char. name of fg 44 ?> = fal m44

```

Enter new 6 char. name of fg 45 ?> = fal45
 Enter new 6 char. name of fg 46 ?> = fal46
 Enter new 6 char. name of fg 47 ?> = fal47
 Enter new 6 char. name of fg 48 ?> = fal48
 Enter new 6 char. name of fg 49 ?> = fal49
 Enter new 6 char. name of fg 50 ?> = fal50
 Enter new 6 char. name of fg 51 ?> = fal51
 Enter new 6 char. name of fg 52 ?> = fal52
 Enter new 6 char. name of fg 53 ?> = fal53
 Enter new 6 char. name of fg 54 ?> = fal54
 Enter new 6 char. name of fg 55 ?> = fal55
 Enter new 6 char. name of fg 56 ?> = fal56
 Enter new 6 char. name of fg 57 ?> = fal57
 Enter new 6 char. name of fg 58 ?> = fal58
 Enter new 6 char. name of fg 59 ?> = fal59
 Enter new 6 char. name of fg 60 ?> = fal60
 Enter new 6 char. name of fg 61 ?> = fal61
 Enter new 6 char. name of fg 62 ?> = fal62
 Enter new 6 char. name of fg 63 ?> = fpas0
 Enter new 6 char. name of fg 64 ?> = fpas1
 Enter new 6 char. name of fg 65 ?> = fpas2
 Enter new 6 char. name of fg 66 ?> = fpas3
 Enter new 6 char. name of fg 67 ?> = fpas4
 Enter new 6 char. name of fg 68 ?> = fpas5
 Enter new 6 char. name of fg 69 ?> = fpas6
 Enter new 6 char. name of fg 70 ?> = fpas7
 Enter new 6 char. name of fg 71 ?> = fpas8
 Enter new 6 char. name of fg 72 ?> = fpas9
 Enter new 6 char. name of fg 73 ?> = fpas10
 Enter new 6 char. name of fg 74 ?> = fpas11
 Enter new 6 char. name of fg 75 ?> = fpas12
 Enter new 6 char. name of fg 76 ?> = fpas13
 Enter new 6 char. name of fg 77 ?> = fpas14
 Enter new 6 char. name of fg 78 ?> = fpas15
 Enter new 6 char. name of fg 79 ?> = fpas16
 Enter new 6 char. name of fg 80 ?> = fpas17
 Enter new 6 char. name of fg 81 ?> = fpas18
 Enter new 6 char. name of fg 82 ?> = fpas19
 Enter new 6 char. name of fg 83 ?> = fpas20
 Enter new 6 char. name of fg 84 ?> = fpas21
 Enter new 6 char. name of fg 85 ?> = fpas22
 Enter new 6 char. name of fg 86 ?> = fpas23
 Enter new 6 char. name of fg 87 ?> = fpas24
 Enter new 6 char. name of fg 88 ?> = fpas25
 Enter new 6 char. name of fg 89 ?> = fpas26
 Enter new 6 char. name of fg 90 ?> = fpas27
 Enter new 6 char. name of fg 91 ?> = fpas28
 Enter new 6 char. name of fg 92 ?> = fpas29
 Enter new 6 char. name of fg 93 ?> = fpas30
 Enter new 6 char. name of fg 94 ?> = fpas31

Enter new 6 char. name of fg 95 ?> = fpas32
Enter new 6 char. name of fg 96 ?> = fpas33
Enter new 6 char. name of fg 97 ?> = fpas34
Enter new 6 char. name of fg 98 ?> = fpas35
Enter new 6 char. name of fg 99 ?> = fpas36
Enter new 6 char. name of fg 100 ?> = fpas37
Enter new 6 char. name of fg 101 ?> = fpas38
Enter new 6 char. name of fg 102 ?> = fpas39
Enter new 6 char. name of fg 103 ?> = fpas40
Enter new 6 char. name of fg 104 ?> = fpas41
Enter new 6 char. name of fg 105 ?> = fpas42
Enter new 6 char. name of fg 106 ?> = fpas43
Enter new 6 char. name of fg 107 ?> = fpas44
Enter new 6 char. name of fg 108 ?> = fpas45
Enter new 6 char. name of fg 109 ?> = fpas46
Enter new 6 char. name of fg 110 ?> = fpas47
Enter new 6 char. name of fg 111 ?> = fpas48
Enter new 6 char. name of fg 112 ?> = fpas49
Enter new 6 char. name of fg 113 ?> = fpas50
Enter new 6 char. name of fg 114 ?> = fpas51
Enter new 6 char. name of fg 115 ?> = fpas52
Enter new 6 char. name of fg 116 ?> = fpas53
Enter new 6 char. name of fg 117 ?> = fpas54
Enter new 6 char. name of fg 118 ?> = fpas55
Enter new 6 char. name of fg 119 ?> = fpas56
Enter new 6 char. name of fg 120 ?> = fpas57
Enter new 6 char. name of fg 121 ?> = fpas58
Enter new 6 char. name of fg 122 ?> = fpas59
Enter new 6 char. name of fg 123 ?> = fpas60
Enter new 6 char. name of fg 124 ?> = fpas61
Enter new 6 char. name of fg 125 ?> = fpas62
Enter new 6 char. name of fg 126 ?> = fpas63
Enter new 6 char. name of fg 127 ?> = frv0
Enter new 6 char. name of fg 128 ?> = frv1
Enter new 6 char. name of fg 129 ?> = frv2
Enter new 6 char. name of fg 130 ?> = frv3
Enter new 6 char. name of fg 131 ?> = frv4
Enter new 6 char. name of fg 132 ?> = frv5
Enter new 6 char. name of fg 133 ?> = frv6
Enter new 6 char. name of fg 134 ?> = frv7
Enter new 6 char. name of fg 135 ?> = frv8
Enter new 6 char. name of fg 136 ?> = frv9
Enter new 6 char. name of fg 137 ?> = frv10
Enter new 6 char. name of fg 138 ?> = frv11
Enter new 6 char. name of fg 139 ?> = frv12
Enter new 6 char. name of fg 140 ?> = frv13
Enter new 6 char. name of fg 141 ?> = frv14
Enter new 6 char. name of fg 142 ?> = frv15
Enter new 6 char. name of fg 143 ?> = frv16
Enter new 6 char. name of fg 144 ?> = frv17

Enter new 6 char. name of fg 145 ?> = frv18
Enter new 6 char. name of fg 146 ?> = frv19
Enter new 6 char. name of fg 147 ?> = frv20
Enter new 6 char. name of fg 148 ?> = frv21
Enter new 6 char. name of fg 149 ?> = frv22
Enter new 6 char. name of fg 150 ?> = frv23
Enter new 6 char. name of fg 151 ?> = frv24
Enter new 6 char. name of fg 152 ?> = frv25
Enter new 6 char. name of fg 153 ?> = frv26
Enter new 6 char. name of fg 154 ?> = frv27
Enter new 6 char. name of fg 155 ?> = frv28
Enter new 6 char. name of fg 156 ?> = frv29
Enter new 6 char. name of fg 157 ?> = frv30
Enter new 6 char. name of fg 158 ?> = frv31
Enter new 6 char. name of fg 159 ?> = frv32
Enter new 6 char. name of fg 160 ?> = frv33
Enter new 6 char. name of fg 161 ?> = frv34
Enter new 6 char. name of fg 162 ?> = frv35
Enter new 6 char. name of fg 163 ?> = frv36
Enter new 6 char. name of fg 164 ?> = frv37
Enter new 6 char. name of fg 165 ?> = frv38
Enter new 6 char. name of fg 166 ?> = frv39
Enter new 6 char. name of fg 167 ?> = frv40
Enter new 6 char. name of fg 168 ?> = frv41
Enter new 6 char. name of fg 169 ?> = frv42
Enter new 6 char. name of fg 170 ?> = frv43
Enter new 6 char. name of fg 171 ?> = frv44
Enter new 6 char. name of fg 172 ?> = frv45
Enter new 6 char. name of fg 173 ?> = frv46
Enter new 6 char. name of fg 174 ?> = frv47
Enter new 6 char. name of fg 175 ?> = frv48
Enter new 6 char. name of fg 176 ?> = frv49
Enter new 6 char. name of fg 177 ?> = frv50
Enter new 6 char. name of fg 178 ?> = frv51
Enter new 6 char. name of fg 179 ?> = frv52
Enter new 6 char. name of fg 180 ?> = frv53
Enter new 6 char. name of fg 181 ?> = frv54
Enter new 6 char. name of fg 182 ?> = frv55
Enter new 6 char. name of fg 183 ?> = frv56
Enter new 6 char. name of fg 184 ?> = frv57
Enter new 6 char. name of fg 185 ?> = frv58
Enter new 6 char. name of fg 186 ?> = frv59
Enter new 6 char. name of fg 187 ?> = frv60
Enter new 6 char. name of fg 188 ?> = frv61
Enter new 6 char. name of fg 189 ?> = frv62
Enter new 6 char. name of fg 190 ?> = frv63
Enter new 6 char. name of fg 191 ?> = aleak0
Enter new 6 char. name of fg 192 ?> = aleak1
Enter new 6 char. name of fg 193 ?> = aleak2
Enter new 6 char. name of fg 194 ?> = aleak3

Enter new 6 char. name of fg 195 ?> = bleak0
Enter new 6 char. name of fg 196 ?> = bleak1
Enter new 6 char. name of fg 197 ?> = bleak2
Enter new 6 char. name of fg 198 ?> = bleak3
Enter new 6 char. name of fg 199 ?> = cleak0
Enter new 6 char. name of fg 200 ?> = cleak1
Enter new 6 char. name of fg 201 ?> = cleak2
Enter new 6 char. name of fg 202 ?> = cleak3
Enter new 6 char. name of fg 203 ?> = dleak0
Enter new 6 char. name of fg 204 ?> = dleak1
Enter new 6 char. name of fg 205 ?> = dleak2
Enter new 6 char. name of fg 206 ?> = dleak3
Enter new 6 char. name of fg 207 ?> = at0
Enter new 6 char. name of fg 208 ?> = at1
Enter new 6 char. name of fg 209 ?> = at2
Enter new 6 char. name of fg 210 ?> = at3
Enter new 6 char. name of fg 211 ?> = at4
Enter new 6 char. name of fg 212 ?> = at5
Enter new 6 char. name of fg 213 ?> = at6
Enter new 6 char. name of fg 214 ?> = at7
Enter new 6 char. name of fg 215 ?> = at8
Enter new 6 char. name of fg 216 ?> = at9
Enter new 6 char. name of fg 217 ?> = at10
Enter new 6 char. name of fg 218 ?> = at11
Enter new 6 char. name of fg 219 ?> = at12
Enter new 6 char. name of fg 220 ?> = at13
Enter new 6 char. name of fg 221 ?> = at14
Enter new 6 char. name of fg 222 ?> = at15
Enter new 6 char. name of fg 223 ?> = at16
Enter new 6 char. name of fg 224 ?> = at17
Enter new 6 char. name of fg 225 ?> = at18
Enter new 6 char. name of fg 226 ?> = at19
Enter new 6 char. name of fg 227 ?> = at20
Enter new 6 char. name of fg 228 ?> = at21
Enter new 6 char. name of fg 229 ?> = at22
Enter new 6 char. name of fg 230 ?> = at23
Enter new 6 char. name of fg 231 ?> = at24
Enter new 6 char. name of fg 232 ?> = at25
Enter new 6 char. name of fg 233 ?> = at26
Enter new 6 char. name of fg 234 ?> = at27
Enter new 6 char. name of fg 235 ?> = at28
Enter new 6 char. name of fg 236 ?> = at29
Enter new 6 char. name of fg 237 ?> = at30
Enter new 6 char. name of fg 238 ?> = at31
Enter new 6 char. name of fg 239 ?> = bt0
Enter new 6 char. name of fg 240 ?> = bt1
Enter new 6 char. name of fg 241 ?> = bt2
Enter new 6 char. name of fg 242 ?> = bt3
Enter new 6 char. name of fg 243 ?> = bt4
Enter new 6 char. name of fg 244 ?> = bt5

Enter new 6 char. name of fg 245 ?> = bt6
 Enter new 6 char. name of fg 246 ?> = bt7
 Enter new 6 char. name of fg 247 ?> = bt8
 Enter new 6 char. name of fg 248 ?> = bt9
 Enter new 6 char. name of fg 249 ?> = bt10
 Enter new 6 char. name of fg 250 ?> = bt11
 Enter new 6 char. name of fg 251 ?> = bt12
 Enter new 6 char. name of fg 252 ?> = bt13
 Enter new 6 char. name of fg 253 ?> = bt14
 Enter new 6 char. name of fg 254 ?> = bt15
 Enter new 6 char. name of fg 255 ?> = bt16
 Enter new 6 char. name of fg 256 ?> = bt17
 Enter new 6 char. name of fg 257 ?> = bt18
 Enter new 6 char. name of fg 258 ?> = bt19
 Enter new 6 char. name of fg 259 ?> = bt20
 Enter new 6 char. name of fg 260 ?> = bt21
 Enter new 6 char. name of fg 261 ?> = bt22
 Enter new 6 char. name of fg 262 ?> = bt23
 Enter new 6 char. name of fg 263 ?> = bt24
 Enter new 6 char. name of fg 264 ?> = bt25
 Enter new 6 char. name of fg 265 ?> = bt26
 Enter new 6 char. name of fg 266 ?> = bt27
 Enter new 6 char. name of fg 267 ?> = bt28
 Enter new 6 char. name of fg 268 ?> = bt29
 Enter new 6 char. name of fg 269 ?> = bt30
 Enter new 6 char. name of fg 270 ?> = bt31
 Enter new 6 char. name of fg 271 ?> = ct0
 Enter new 6 char. name of fg 272 ?> = ct1
 Enter new 6 char. name of fg 273 ?> = ct2
 Enter new 6 char. name of fg 274 ?> = ct3
 Enter new 6 char. name of fg 275 ?> = ct4
 Enter new 6 char. name of fg 276 ?> = ct5
 Enter new 6 char. name of fg 277 ?> = ct6
 Enter new 6 char. name of fg 278 ?> = ct7
 Enter new 6 char. name of fg 279 ?> = ct8
 Enter new 6 char. name of fg 280 ?> = ct9
 Enter new 6 char. name of fg 281 ?> = ct10
 Enter new 6 char. name of fg 282 ?> = ct11
 Enter new 6 char. name of fg 283 ?> = ct12
 Enter new 6 char. name of fg 284 ?> = ct13
 Enter new 6 char. name of fg 285 ?> = ct14
 Enter new 6 char. name of fg 286 ?> = ct15
 Enter new 6 char. name of fg 287 ?> = ct16
 Enter new 6 char. name of fg 288 ?> = ct17
 Enter new 6 char. name of fg 289 ?> = ct18
 Enter new 6 char. name of fg 290 ?> = ct19
 Enter new 6 char. name of fg 291 ?> = ct20
 Enter new 6 char. name of fg 292 ?> = ct21
 Enter new 6 char. name of fg 293 ?> = ct22
 Enter new 6 char. name of fg 294 ?> = ct23

Enter new 6 char. name of fg 295 ?> = ct24
 Enter new 6 char. name of fg 296 ?> = ct25
 Enter new 6 char. name of fg 297 ?> = ct26
 Enter new 6 char. name of fg 298 ?> = ct27
 Enter new 6 char. name of fg 299 ?> = ct28
 Enter new 6 char. name of fg 300 ?> = ct29
 Enter new 6 char. name of fg 301 ?> = ct30
 Enter new 6 char. name of fg 302 ?> = ct31
 Enter new 6 char. name of fg 303 ?> = sa0
 Enter new 6 char. name of fg 304 ?> = sa1
 Enter new 6 char. name of fg 305 ?> = sa2
 Enter new 6 char. name of fg 306 ?> = sa3
 Enter new 6 char. name of fg 307 ?> = sa4
 Enter new 6 char. name of fg 308 ?> = sa5
 Enter new 6 char. name of fg 309 ?> = sa6
 Enter new 6 char. name of fg 310 ?> = sa7
 Enter new 6 char. name of fg 311 ?> = sb0
 Enter new 6 char. name of fg 312 ?> = sb1
 Enter new 6 char. name of fg 313 ?> = sb2
 Enter new 6 char. name of fg 314 ?> = sb3
 Enter new 6 char. name of fg 315 ?> = sb4
 Enter new 6 char. name of fg 316 ?> = sb5
 Enter new 6 char. name of fg 317 ?> = sb6
 Enter new 6 char. name of fg 318 ?> = sb7
 Enter new 6 char. name of fg 319 ?> = sc0
 Enter new 6 char. name of fg 320 ?> = sc1
 Enter new 6 char. name of fg 321 ?> = sc2
 Enter new 6 char. name of fg 322 ?> = sc3
 Enter new 6 char. name of fg 323 ?> = sc4
 Enter new 6 char. name of fg 324 ?> = sc5
 Enter new 6 char. name of fg 325 ?> = sc6
 Enter new 6 char. name of fg 326 ?> = sc7
 Enter new 6 char. name of fg 327 ?> = sd0
 Enter new 6 char. name of fg 328 ?> = sd1
 Enter new 6 char. name of fg 329 ?> = sd2
 Enter new 6 char. name of fg 330 ?> = sd3
 Enter new 6 char. name of fg 331 ?> = sd4
 Enter new 6 char. name of fg 332 ?> = sd5
 Enter new 6 char. name of fg 333 ?> = sd6
 Enter new 6 char. name of fg 334 ?> = sd7
 Enter new 6 char. name of fg 335 ?> = gt0
 Enter new 6 char. name of fg 336 ?> = gt1
 Enter new 6 char. name of fg 337 ?> = gt2
 Enter new 6 char. name of fg 338 ?> = gt3
 Enter new 6 char. name of fg 339 ?> = gt4
 Enter new 6 char. name of fg 340 ?> = gt5
 Enter new 6 char. name of fg 341 ?> = gt6
 Enter new 6 char. name of fg 342 ?> = gt7
 Enter new 6 char. name of fg 343 ?> = gt8
 Enter new 6 char. name of fg 344 ?> = gt9

```

Enter new 6 char. name of fg 345 ?> = gt10
Enter new 6 char. name of fg 346 ?> = gt11
Enter new 6 char. name of fg 347 ?> = gt12
Enter new 6 char. name of fg 348 ?> = gt13
Enter new 6 char. name of fg 349 ?> = gt14
Enter new 6 char. name of fg 350 ?> = gt15
Enter new 6 char. name of fg 351 ?> = fhold
Enter new 6 char. name of fg 352 ?> = fning
Enter new 6 char. name of fg 353 ?> = fabort
Enter new 6 char. name of fg 354 ?> = fstart
Enter new 6 char. name of fg 355 ?> = fpass
Enter new 6 char. name of fg 356 ?> = ftrip0
Enter new 6 char. name of fg 357 ?> = ftrip1
Enter new 6 char. name of fg 358 ?> = ftrip2
Enter new 6 char. name of fg 359 ?> = ftrip3
Enter new 6 char. name of fg 360 ?> = ftrip4
Enter new 6 char. name of fg 361 ?> = fspar0
Enter new 6 char. name of fg 362 ?> = fspar1
Enter new 6 char. name of fg 363 ?> = fsn0
Enter new 6 char. name of fg 364 ?> = falmon
Enter new 6 char. name of fg 365 ?> = fack
Enter new 6 char. name of fg 366 ?> = fclear
Enter new 6 char. name of fg 367 ?> = ack
MUPF?> = add 170 in
Enter new 6 char. name of in 0 ?> = unitt
Enter new 6 char. name of in 1 ?> = analog
Enter new 6 char. name of in 2 ?> = white
Enter new 6 char. name of in 3 ?> = bitval
Enter new 6 char. name of in 4 ?> = lmpvar
Enter new 6 char. name of in 5 ?> = skdkey
Enter new 6 char. name of in 6 ?> = iloop
Enter new 6 char. name of in 7 ?> = jloop
Enter new 6 char. name of in 8 ?> = index
Enter new 6 char. name of in 9 ?> = last
Enter new 6 char. name of in 10 ?> = curent
Enter new 6 char. name of in 11 ?> = nxt
Enter new 6 char. name of in 12 ?> = int
Enter new 6 char. name of in 13 ?> = remain
Enter new 6 char. name of in 14 ?> = trip
Enter new 6 char. name of in 15 ?> = nadi
Enter new 6 char. name of in 16 ?> = blkval
Enter new 6 char. name of in 17 ?> = sort
Enter new 6 char. name of in 18 ?> = bval
Enter new 6 char. name of in 19 ?> = hldstp
Enter new 6 char. name of in 20 ?> = errn
Enter new 6 char. name of in 21 ?> = errl
Enter new 6 char. name of in 22 ?> = adi
Enter new 6 char. name of in 23 ?> = star
Enter new 6 char. name of in 24 ?> = dev0
Enter new 6 char. name of in 25 ?> = dev1

```

WHC-SD-534-CSWL-001 REV 3

```

Enter new 6 char. name of in 26 ?> = dev2
Enter new 6 char. name of in 27 ?> = dev3
Enter new 6 char. name of in 28 ?> = dev4
Enter new 6 char. name of in 29 ?> = dev5
Enter new 6 char. name of in 30 ?> = dev6
Enter new 6 char. name of in 31 ?> = dev7
Enter new 6 char. name of in 32 ?> = dev8
Enter new 6 char. name of in 33 ?> = dev9
Enter new 6 char. name of in 34 ?> = skid0
Enter new 6 char. name of in 35 ?> = skid1
Enter new 6 char. name of in 36 ?> = skid2
Enter new 6 char. name of in 37 ?> = skid3
Enter new 6 char. name of in 38 ?> = skid4
Enter new 6 char. name of in 39 ?> = skid5
Enter new 6 char. name of in 40 ?> = skid6
Enter new 6 char. name of in 41 ?> = skid7
Enter new 6 char. name of in 42 ?> = skid8
Enter new 6 char. name of in 43 ?> = skid9
Enter new 6 char. name of in 44 ?> = skid10
Enter new 6 char. name of in 45 ?> = skid11
Enter new 6 char. name of in 46 ?> = skid12
Enter new 6 char. name of in 47 ?> = skid13
Enter new 6 char. name of in 48 ?> = skid14
Enter new 6 char. name of in 49 ?> = skid15
Enter new 6 char. name of in 50 ?> = skid16
Enter new 6 char. name of in 51 ?> = skid17
Enter new 6 char. name of in 52 ?> = skid18
Enter new 6 char. name of in 53 ?> = skid19
Enter new 6 char. name of in 54 ?> = skid20
Enter new 6 char. name of in 55 ?> = skid21
Enter new 6 char. name of in 56 ?> = skid22
Enter new 6 char. name of in 57 ?> = skid23
Enter new 6 char. name of in 58 ?> = skid24
Enter new 6 char. name of in 59 ?> = skid25
Enter new 6 char. name of in 60 ?> = skid26
Enter new 6 char. name of in 61 ?> = skid27
Enter new 6 char. name of in 62 ?> = skid28
Enter new 6 char. name of in 63 ?> = skid29
Enter new 6 char. name of in 64 ?> = skid30
Enter new 6 char. name of in 65 ?> = skid31
Enter new 6 char. name of in 66 ?> = skid32
Enter new 6 char. name of in 67 ?> = skid33
Enter new 6 char. name of in 68 ?> = skid34
Enter new 6 char. name of in 69 ?> = skid35
Enter new 6 char. name of in 70 ?> = skid36
Enter new 6 char. name of in 71 ?> = skid37
Enter new 6 char. name of in 72 ?> = skid38
Enter new 6 char. name of in 73 ?> = skid39
Enter new 6 char. name of in 74 ?> = type0
Enter new 6 char. name of in 75 ?> = type1

```

Enter new 6 char. name of in 76 ?> = type2
 Enter new 6 char. name of in 77 ?> = type3
 Enter new 6 char. name of in 78 ?> = type4
 Enter new 6 char. name of in 79 ?> = type5
 Enter new 6 char. name of in 80 ?> = type6
 Enter new 6 char. name of in 81 ?> = type7
 Enter new 6 char. name of in 82 ?> = type8
 Enter new 6 char. name of in 83 ?> = type9
 Enter new 6 char. name of in 84 ?> = type10
 Enter new 6 char. name of in 85 ?> = type11
 Enter new 6 char. name of in 86 ?> = type12
 Enter new 6 char. name of in 87 ?> = type13
 Enter new 6 char. name of in 88 ?> = type14
 Enter new 6 char. name of in 89 ?> = type15
 Enter new 6 char. name of in 90 ?> = type16
 Enter new 6 char. name of in 91 ?> = type17
 Enter new 6 char. name of in 92 ?> = type18
 Enter new 6 char. name of in 93 ?> = type19
 Enter new 6 char. name of in 94 ?> = type20
 Enter new 6 char. name of in 95 ?> = type21
 Enter new 6 char. name of in 96 ?> = type22
 Enter new 6 char. name of in 97 ?> = type23
 Enter new 6 char. name of in 98 ?> = type24
 Enter new 6 char. name of in 99 ?> = type25
 Enter new 6 char. name of in 100 ?> = type26
 Enter new 6 char. name of in 101 ?> = type27
 Enter new 6 char. name of in 102 ?> = type28
 Enter new 6 char. name of in 103 ?> = type29
 Enter new 6 char. name of in 104 ?> = type30
 Enter new 6 char. name of in 105 ?> = type31
 Enter new 6 char. name of in 106 ?> = type32
 Enter new 6 char. name of in 107 ?> = type33
 Enter new 6 char. name of in 108 ?> = type34
 Enter new 6 char. name of in 109 ?> = type35
 Enter new 6 char. name of in 110 ?> = type36
 Enter new 6 char. name of in 111 ?> = type37
 Enter new 6 char. name of in 112 ?> = type38
 Enter new 6 char. name of in 113 ?> = type39
 Enter new 6 char. name of in 114 ?> = int0
 Enter new 6 char. name of in 115 ?> = int1
 Enter new 6 char. name of in 116 ?> = int2
 Enter new 6 char. name of in 117 ?> = int3
 Enter new 6 char. name of in 118 ?> = int4
 Enter new 6 char. name of in 119 ?> = int5
 Enter new 6 char. name of in 120 ?> = int6
 Enter new 6 char. name of in 121 ?> = int7
 Enter new 6 char. name of in 122 ?> = int8
 Enter new 6 char. name of in 123 ?> = int9
 Enter new 6 char. name of in 124 ?> = int10
 Enter new 6 char. name of in 125 ?> = int11

```

Enter new 6 char. name of in 126 ?> = int12
Enter new 6 char. name of in 127 ?> = int13
Enter new 6 char. name of in 128 ?> = int14
Enter new 6 char. name of in 129 ?> = int15
Enter new 6 char. name of in 130 ?> = int16
Enter new 6 char. name of in 131 ?> = int17
Enter new 6 char. name of in 132 ?> = int18
Enter new 6 char. name of in 133 ?> = int19
Enter new 6 char. name of in 134 ?> = int20
Enter new 6 char. name of in 135 ?> = int21
Enter new 6 char. name of in 136 ?> = int22
Enter new 6 char. name of in 137 ?> = int23
Enter new 6 char. name of in 138 ?> = int24
Enter new 6 char. name of in 139 ?> = int25
Enter new 6 char. name of in 140 ?> = int26
Enter new 6 char. name of in 141 ?> = int27
Enter new 6 char. name of in 142 ?> = int28
Enter new 6 char. name of in 143 ?> = int29
Enter new 6 char. name of in 144 ?> = int30
Enter new 6 char. name of in 145 ?> = int31
Enter new 6 char. name of in 146 ?> = int32
Enter new 6 char. name of in 147 ?> = int33
Enter new 6 char. name of in 148 ?> = int34
Enter new 6 char. name of in 149 ?> = int35
Enter new 6 char. name of in 150 ?> = int36
Enter new 6 char. name of in 151 ?> = int37
Enter new 6 char. name of in 152 ?> = int38
Enter new 6 char. name of in 153 ?> = int39
Enter new 6 char. name of in 154 ?> = leak0
Enter new 6 char. name of in 155 ?> = leak1
Enter new 6 char. name of in 156 ?> = leak2
Enter new 6 char. name of in 157 ?> = leak3
Enter new 6 char. name of in 158 ?> = leak4
Enter new 6 char. name of in 159 ?> = leak5
Enter new 6 char. name of in 160 ?> = color0
Enter new 6 char. name of in 161 ?> = color1
Enter new 6 char. name of in 162 ?> = color2
Enter new 6 char. name of in 163 ?> = color3
Enter new 6 char. name of in 164 ?> = color4
Enter new 6 char. name of in 165 ?> = grafx0
Enter new 6 char. name of in 166 ?> = grafx1
Enter new 6 char. name of in 167 ?> = grafx2
Enter new 6 char. name of in 168 ?> = grafx3
Enter new 6 char. name of in 169 ?> = grafx4
MUPF?> = add 36 fl
Enter new 6 char. name of fl 0 ?> = ai0
Enter new 6 char. name of fl 1 ?> = ai1
Enter new 6 char. name of fl 2 ?> = ai2
Enter new 6 char. name of fl 3 ?> = ai3
Enter new 6 char. name of fl 4 ?> = ai4

```



```

Enter new 6 char. name of fl 5 ?> = ai5
Enter new 6 char. name of fl 6 ?> = ai6
Enter new 6 char. name of fl 7 ?> = ai7
Enter new 6 char. name of fl 8 ?> = ai8
Enter new 6 char. name of fl 9 ?> = ai9
Enter new 6 char. name of fl 10 ?> = ai10
Enter new 6 char. name of fl 11 ?> = ai11
Enter new 6 char. name of fl 12 ?> = ai12
Enter new 6 char. name of fl 13 ?> = ai13
Enter new 6 char. name of fl 14 ?> = ai14
Enter new 6 char. name of fl 15 ?> = ai15
Enter new 6 char. name of fl 16 ?> = ai16
Enter new 6 char. name of fl 17 ?> = ai17
Enter new 6 char. name of fl 18 ?> = ai18
Enter new 6 char. name of fl 19 ?> = ai19
Enter new 6 char. name of fl 20 ?> = ai20
Enter new 6 char. name of fl 21 ?> = ai21
Enter new 6 char. name of fl 22 ?> = ai22
Enter new 6 char. name of fl 23 ?> = ai23
Enter new 6 char. name of fl 24 ?> = ai24
Enter new 6 char. name of fl 25 ?> = ai25
Enter new 6 char. name of fl 26 ?> = ai26
Enter new 6 char. name of fl 27 ?> = ai27
Enter new 6 char. name of fl 28 ?> = ai28
Enter new 6 char. name of fl 29 ?> = ai29
Enter new 6 char. name of fl 30 ?> = ai30
Enter new 6 char. name of fl 31 ?> = ai31
Enter new 6 char. name of fl 32 ?> = ai32
Enter new 6 char. name of fl 33 ?> = ai33
Enter new 6 char. name of fl 34 ?> = ai34
Enter new 6 char. name of fl 35 ?> = ai35
MUPF?> = add 11 pt
Enter new 6 char. name of pt 0 ?> = pepn
Enter new 6 char. name of pt 1 ?> = pdin
Enter new 6 char. name of pt 2 ?> = pdev
Enter new 6 char. name of pt 3 ?> = pai
Enter new 6 char. name of pt 4 ?> = palarm
Enter new 6 char. name of pt 5 ?> = pskid
Enter new 6 char. name of pt 6 ?> = pstat
Enter new 6 char. name of pt 7 ?> = ptype
Enter new 6 char. name of pt 8 ?> = ppass
Enter new 6 char. name of pt 9 ?> = prev
Enter new 6 char. name of pt 10 ?> = pspar0
MUPF?> = add 15 sg
Enter new 6 char. name of sg 0 ?> = epns0
Enter max string length for sg 0 ?> = 10
Enter new 6 char. name of sg 1 ?> = epns1
Enter max string length for sg 1 ?> = 10
Enter new 6 char. name of sg 2 ?> = epns2
Enter max string length for sg 2 ?> = 10

```

```

Enter new 6 char. name of sg 3 ?> = epns3
Enter max string length for sg 3 ?> = 10
Enter new 6 char. name of sg 4 ?> = epns4
Enter max string length for sg 4 ?> = 10
Enter new 6 char. name of sg 5 ?> = lsg0
Enter max string length for sg 5 ?> = 30
Enter new 6 char. name of sg 6 ?> = lsg1
Enter max string length for sg 6 ?> = 30
Enter new 6 char. name of sg 7 ?> = lsg2
Enter max string length for sg 7 ?> = 30
Enter new 6 char. name of sg 8 ?> = lsg3
Enter max string length for sg 8 ?> = 30
Enter new 6 char. name of sg 9 ?> = lsg4
Enter max string length for sg 9 ?> = 30
Enter new 6 char. name of sg 10 ?> = strng0
Enter max string length for sg 10 ?> = 30
Enter new 6 char. name of sg 11 ?> = strng1
Enter max string length for sg 11 ?> = 30
Enter new 6 char. name of sg 12 ?> = strng2
Enter max string length for sg 12 ?> = 30
Enter new 6 char. name of sg 13 ?> = strng3
Enter max string length for sg 13 ?> = 30
Enter new 6 char. name of sg 14 ?> = strng4
Enter max string length for sg 14 ?> = 30
MUPF?>
    end of file encountered
UPF?>
    end of file encountered

```

OVERVIEW

UPF Version 6.3-2   05-Dec-94 09:37:11
UPF?> = de ovrview
UPF?> = cr ovrview,*
Enter 4 character name of RPF to associate?> = 242a
MUPF?> = add 720 fg
Enter new 6 char. name of fg 0 ?> = fai0
Enter new 6 char. name of fg 1 ?> = fai1
Enter new 6 char. name of fg 2 ?> = fai2
Enter new 6 char. name of fg 3 ?> = fai3
Enter new 6 char. name of fg 4 ?> = fai4
Enter new 6 char. name of fg 5 ?> = fai5
Enter new 6 char. name of fg 6 ?> = fai6
Enter new 6 char. name of fg 7 ?> = fai7
Enter new 6 char. name of fg 8 ?> = fai8
Enter new 6 char. name of fg 9 ?> = fai9
Enter new 6 char. name of fg 10 ?> = fai10
Enter new 6 char. name of fg 11 ?> = fai11
Enter new 6 char. name of fg 12 ?> = fai12
Enter new 6 char. name of fg 13 ?> = fai13
Enter new 6 char. name of fg 14 ?> = fai14
Enter new 6 char. name of fg 15 ?> = fai15
Enter new 6 char. name of fg 16 ?> = fai16
Enter new 6 char. name of fg 17 ?> = fai17
Enter new 6 char. name of fg 18 ?> = fai18
Enter new 6 char. name of fg 19 ?> = fai19
Enter new 6 char. name of fg 20 ?> = fai20
Enter new 6 char. name of fg 21 ?> = fai21
Enter new 6 char. name of fg 22 ?> = fai22
Enter new 6 char. name of fg 23 ?> = fai23
Enter new 6 char. name of fg 24 ?> = fai24
Enter new 6 char. name of fg 25 ?> = fai25
Enter new 6 char. name of fg 26 ?> = fai26
Enter new 6 char. name of fg 27 ?> = fai27
Enter new 6 char. name of fg 28 ?> = fai28
Enter new 6 char. name of fg 29 ?> = fai29
Enter new 6 char. name of fg 30 ?> = fai30
Enter new 6 char. name of fg 31 ?> = fai31
Enter new 6 char. name of fg 32 ?> = fai32
Enter new 6 char. name of fg 33 ?> = fai33
Enter new 6 char. name of fg 34 ?> = fai34
Enter new 6 char. name of fg 35 ?> = fai35
Enter new 6 char. name of fg 36 ?> = fai36
Enter new 6 char. name of fg 37 ?> = fai37
Enter new 6 char. name of fg 38 ?> = fai38
Enter new 6 char. name of fg 39 ?> = fai39
Enter new 6 char. name of fg 40 ?> = fai40
Enter new 6 char. name of fg 41 ?> = fai41
Enter new 6 char. name of fg 42 ?> = fai42
Enter new 6 char. name of fg 43 ?> = fai43
Enter new 6 char. name of fg 44 ?> = fai44

Enter new 6 char. name of fg 45 ?> = fai45
Enter new 6 char. name of fg 46 ?> = fai46
Enter new 6 char. name of fg 47 ?> = fai47
Enter new 6 char. name of fg 48 ?> = fai48
Enter new 6 char. name of fg 49 ?> = fai49
Enter new 6 char. name of fg 50 ?> = fai50
Enter new 6 char. name of fg 51 ?> = fai51
Enter new 6 char. name of fg 52 ?> = fai52
Enter new 6 char. name of fg 53 ?> = fai53
Enter new 6 char. name of fg 54 ?> = fai54
Enter new 6 char. name of fg 55 ?> = fai55
Enter new 6 char. name of fg 56 ?> = fai56
Enter new 6 char. name of fg 57 ?> = fai57
Enter new 6 char. name of fg 58 ?> = fai58
Enter new 6 char. name of fg 59 ?> = fai59
Enter new 6 char. name of fg 60 ?> = fai60
Enter new 6 char. name of fg 61 ?> = fai61
Enter new 6 char. name of fg 62 ?> = fai62
Enter new 6 char. name of fg 63 ?> = fai63
Enter new 6 char. name of fg 64 ?> = fai64
Enter new 6 char. name of fg 65 ?> = fai65
Enter new 6 char. name of fg 66 ?> = fai66
Enter new 6 char. name of fg 67 ?> = fai67
Enter new 6 char. name of fg 68 ?> = fai68
Enter new 6 char. name of fg 69 ?> = fai69
Enter new 6 char. name of fg 70 ?> = fai70
Enter new 6 char. name of fg 71 ?> = fai71
Enter new 6 char. name of fg 72 ?> = fai72
Enter new 6 char. name of fg 73 ?> = fai73
Enter new 6 char. name of fg 74 ?> = fai74
Enter new 6 char. name of fg 75 ?> = fai75
Enter new 6 char. name of fg 76 ?> = fai76
Enter new 6 char. name of fg 77 ?> = fai77
Enter new 6 char. name of fg 78 ?> = fai78
Enter new 6 char. name of fg 79 ?> = fai79
Enter new 6 char. name of fg 80 ?> = fai80
Enter new 6 char. name of fg 81 ?> = fai81
Enter new 6 char. name of fg 82 ?> = fai82
Enter new 6 char. name of fg 83 ?> = fai83
Enter new 6 char. name of fg 84 ?> = fai84
Enter new 6 char. name of fg 85 ?> = fai85
Enter new 6 char. name of fg 86 ?> = fai86
Enter new 6 char. name of fg 87 ?> = fai87
Enter new 6 char. name of fg 88 ?> = fai88
Enter new 6 char. name of fg 89 ?> = fai89
Enter new 6 char. name of fg 90 ?> = fai90
Enter new 6 char. name of fg 91 ?> = fai91
Enter new 6 char. name of fg 92 ?> = fai92
Enter new 6 char. name of fg 93 ?> = fai93
Enter new 6 char. name of fg 94 ?> = fai94

Enter new 6 char. name of fg 95 ?> = fail95
Enter new 6 char. name of fg 96 ?> = fail96
Enter new 6 char. name of fg 97 ?> = fail97
Enter new 6 char. name of fg 98 ?> = fail98
Enter new 6 char. name of fg 99 ?> = fail99
Enter new 6 char. name of fg 100 ?> = fail100
Enter new 6 char. name of fg 101 ?> = fail101
Enter new 6 char. name of fg 102 ?> = fail102
Enter new 6 char. name of fg 103 ?> = fail103
Enter new 6 char. name of fg 104 ?> = fail104
Enter new 6 char. name of fg 105 ?> = fail105
Enter new 6 char. name of fg 106 ?> = fail106
Enter new 6 char. name of fg 107 ?> = fail107
Enter new 6 char. name of fg 108 ?> = fail108
Enter new 6 char. name of fg 109 ?> = fail109
Enter new 6 char. name of fg 110 ?> = fail110
Enter new 6 char. name of fg 111 ?> = fail111
Enter new 6 char. name of fg 112 ?> = fail112
Enter new 6 char. name of fg 113 ?> = fail113
Enter new 6 char. name of fg 114 ?> = fail114
Enter new 6 char. name of fg 115 ?> = fail115
Enter new 6 char. name of fg 116 ?> = fail116
Enter new 6 char. name of fg 117 ?> = fail117
Enter new 6 char. name of fg 118 ?> = fail118
Enter new 6 char. name of fg 119 ?> = fail119
Enter new 6 char. name of fg 120 ?> = fail120
Enter new 6 char. name of fg 121 ?> = fail121
Enter new 6 char. name of fg 122 ?> = fail122
Enter new 6 char. name of fg 123 ?> = fail123
Enter new 6 char. name of fg 124 ?> = fail124
Enter new 6 char. name of fg 125 ?> = fail125
Enter new 6 char. name of fg 126 ?> = fail126
Enter new 6 char. name of fg 127 ?> = fail127
Enter new 6 char. name of fg 128 ?> = fail128
Enter new 6 char. name of fg 129 ?> = fail129
Enter new 6 char. name of fg 130 ?> = fail130
Enter new 6 char. name of fg 131 ?> = fail131
Enter new 6 char. name of fg 132 ?> = fail132
Enter new 6 char. name of fg 133 ?> = fail133
Enter new 6 char. name of fg 134 ?> = fail134
Enter new 6 char. name of fg 135 ?> = fail135
Enter new 6 char. name of fg 136 ?> = fail136
Enter new 6 char. name of fg 137 ?> = fail137
Enter new 6 char. name of fg 138 ?> = fail138
Enter new 6 char. name of fg 139 ?> = fail139
Enter new 6 char. name of fg 140 ?> = fail140
Enter new 6 char. name of fg 141 ?> = fail141
Enter new 6 char. name of fg 142 ?> = fail142
Enter new 6 char. name of fg 143 ?> = fail143
Enter new 6 char. name of fg 144 ?> = fail144

Enter new 6 char. name of fg 145 ?> = fail145
Enter new 6 char. name of fg 146 ?> = fail146
Enter new 6 char. name of fg 147 ?> = fail147
Enter new 6 char. name of fg 148 ?> = fail148
Enter new 6 char. name of fg 149 ?> = fail149
Enter new 6 char. name of fg 150 ?> = fail150
Enter new 6 char. name of fg 151 ?> = fail151
Enter new 6 char. name of fg 152 ?> = fail152
Enter new 6 char. name of fg 153 ?> = fail153
Enter new 6 char. name of fg 154 ?> = fail154
Enter new 6 char. name of fg 155 ?> = fail155
Enter new 6 char. name of fg 156 ?> = fail156
Enter new 6 char. name of fg 157 ?> = fail157
Enter new 6 char. name of fg 158 ?> = fail158
Enter new 6 char. name of fg 159 ?> = fail159
Enter new 6 char. name of fg 160 ?> = fail160
Enter new 6 char. name of fg 161 ?> = fail161
Enter new 6 char. name of fg 162 ?> = fail162
Enter new 6 char. name of fg 163 ?> = fail163
Enter new 6 char. name of fg 164 ?> = fail164
Enter new 6 char. name of fg 165 ?> = fail165
Enter new 6 char. name of fg 166 ?> = fail166
Enter new 6 char. name of fg 167 ?> = fail167
Enter new 6 char. name of fg 168 ?> = fail168
Enter new 6 char. name of fg 169 ?> = fail169
Enter new 6 char. name of fg 170 ?> = fail170
Enter new 6 char. name of fg 171 ?> = fail171
Enter new 6 char. name of fg 172 ?> = fail172
Enter new 6 char. name of fg 173 ?> = fail173
Enter new 6 char. name of fg 174 ?> = fail174
Enter new 6 char. name of fg 175 ?> = fail175
Enter new 6 char. name of fg 176 ?> = fail176
Enter new 6 char. name of fg 177 ?> = fail177
Enter new 6 char. name of fg 178 ?> = fail178
Enter new 6 char. name of fg 179 ?> = fail179
Enter new 6 char. name of fg 180 ?> = fail180
Enter new 6 char. name of fg 181 ?> = fail181
Enter new 6 char. name of fg 182 ?> = fail182
Enter new 6 char. name of fg 183 ?> = fail183
Enter new 6 char. name of fg 184 ?> = fail184
Enter new 6 char. name of fg 185 ?> = fail185
Enter new 6 char. name of fg 186 ?> = fail186
Enter new 6 char. name of fg 187 ?> = fail187
Enter new 6 char. name of fg 188 ?> = fail188
Enter new 6 char. name of fg 189 ?> = fail189
Enter new 6 char. name of fg 190 ?> = fail190
Enter new 6 char. name of fg 191 ?> = fail191
Enter new 6 char. name of fg 192 ?> = fail192
Enter new 6 char. name of fg 193 ?> = fail193
Enter new 6 char. name of fg 194 ?> = fail194

Enter new 6 char. name of fg 195 ?> = fal119
Enter new 6 char. name of fg 196 ?> = fal120
Enter new 6 char. name of fg 197 ?> = fal121
Enter new 6 char. name of fg 198 ?> = fal122
Enter new 6 char. name of fg 199 ?> = fal123
Enter new 6 char. name of fg 200 ?> = fal124
Enter new 6 char. name of fg 201 ?> = fal125
Enter new 6 char. name of fg 202 ?> = fal126
Enter new 6 char. name of fg 203 ?> = fal127
Enter new 6 char. name of fg 204 ?> = fal128
Enter new 6 char. name of fg 205 ?> = fal129
Enter new 6 char. name of fg 206 ?> = fal130
Enter new 6 char. name of fg 207 ?> = fal131
Enter new 6 char. name of fg 208 ?> = fal132
Enter new 6 char. name of fg 209 ?> = fal133
Enter new 6 char. name of fg 210 ?> = fal134
Enter new 6 char. name of fg 211 ?> = fal135
Enter new 6 char. name of fg 212 ?> = fal136
Enter new 6 char. name of fg 213 ?> = fal137
Enter new 6 char. name of fg 214 ?> = fal138
Enter new 6 char. name of fg 215 ?> = fal139
Enter new 6 char. name of fg 216 ?> = fal140
Enter new 6 char. name of fg 217 ?> = fal141
Enter new 6 char. name of fg 218 ?> = fal142
Enter new 6 char. name of fg 219 ?> = fal143
Enter new 6 char. name of fg 220 ?> = fal144
Enter new 6 char. name of fg 221 ?> = fal145
Enter new 6 char. name of fg 222 ?> = fal146
Enter new 6 char. name of fg 223 ?> = fal147
Enter new 6 char. name of fg 224 ?> = fal148
Enter new 6 char. name of fg 225 ?> = fal149
Enter new 6 char. name of fg 226 ?> = fal150
Enter new 6 char. name of fg 227 ?> = fal151
Enter new 6 char. name of fg 228 ?> = fal152
Enter new 6 char. name of fg 229 ?> = fal153
Enter new 6 char. name of fg 230 ?> = fal154
Enter new 6 char. name of fg 231 ?> = fal155
Enter new 6 char. name of fg 232 ?> = fal156
Enter new 6 char. name of fg 233 ?> = fal157
Enter new 6 char. name of fg 234 ?> = fal158
Enter new 6 char. name of fg 235 ?> = fal159
Enter new 6 char. name of fg 236 ?> = fal160
Enter new 6 char. name of fg 237 ?> = fal161
Enter new 6 char. name of fg 238 ?> = fal162
Enter new 6 char. name of fg 239 ?> = fal163
Enter new 6 char. name of fg 240 ?> = fal164
Enter new 6 char. name of fg 241 ?> = fal165
Enter new 6 char. name of fg 242 ?> = fal166
Enter new 6 char. name of fg 243 ?> = fal167
Enter new 6 char. name of fg 244 ?> = fal168

Enter new 6 char. name of fg 245 ?> = fal69
Enter new 6 char. name of fg 246 ?> = fal70
Enter new 6 char. name of fg 247 ?> = fal71
Enter new 6 char. name of fg 248 ?> = fal72
Enter new 6 char. name of fg 249 ?> = fal73
Enter new 6 char. name of fg 250 ?> = fal74
Enter new 6 char. name of fg 251 ?> = fal75
Enter new 6 char. name of fg 252 ?> = fal76
Enter new 6 char. name of fg 253 ?> = fal77
Enter new 6 char. name of fg 254 ?> = fal78
Enter new 6 char. name of fg 255 ?> = fal79
Enter new 6 char. name of fg 256 ?> = fal80
Enter new 6 char. name of fg 257 ?> = fal81
Enter new 6 char. name of fg 258 ?> = fal82
Enter new 6 char. name of fg 259 ?> = fal83
Enter new 6 char. name of fg 260 ?> = fal84
Enter new 6 char. name of fg 261 ?> = fal85
Enter new 6 char. name of fg 262 ?> = fal86
Enter new 6 char. name of fg 263 ?> = fal87
Enter new 6 char. name of fg 264 ?> = fal88
Enter new 6 char. name of fg 265 ?> = fal89
Enter new 6 char. name of fg 266 ?> = fal90
Enter new 6 char. name of fg 267 ?> = fal91
Enter new 6 char. name of fg 268 ?> = fal92
Enter new 6 char. name of fg 269 ?> = fal93
Enter new 6 char. name of fg 270 ?> = fal94
Enter new 6 char. name of fg 271 ?> = fal95
Enter new 6 char. name of fg 272 ?> = fal96
Enter new 6 char. name of fg 273 ?> = fal97
Enter new 6 char. name of fg 274 ?> = fal98
Enter new 6 char. name of fg 275 ?> = fal99
Enter new 6 char. name of fg 276 ?> = fal100
Enter new 6 char. name of fg 277 ?> = fal101
Enter new 6 char. name of fg 278 ?> = fal102
Enter new 6 char. name of fg 279 ?> = fal103
Enter new 6 char. name of fg 280 ?> = fal104
Enter new 6 char. name of fg 281 ?> = fal105
Enter new 6 char. name of fg 282 ?> = fal106
Enter new 6 char. name of fg 283 ?> = fal107
Enter new 6 char. name of fg 284 ?> = fal108
Enter new 6 char. name of fg 285 ?> = fal109
Enter new 6 char. name of fg 286 ?> = fal110
Enter new 6 char. name of fg 287 ?> = fal111
Enter new 6 char. name of fg 288 ?> = fal112
Enter new 6 char. name of fg 289 ?> = fal113
Enter new 6 char. name of fg 290 ?> = fal114
Enter new 6 char. name of fg 291 ?> = fal115
Enter new 6 char. name of fg 292 ?> = fal116
Enter new 6 char. name of fg 293 ?> = fal117
Enter new 6 char. name of fg 294 ?> = fal118

Enter new 6 char. name of fg 295 ?> = fall119
 Enter new 6 char. name of fg 296 ?> = fall120
 Enter new 6 char. name of fg 297 ?> = fall121
 Enter new 6 char. name of fg 298 ?> = fall122
 Enter new 6 char. name of fg 299 ?> = fall123
 Enter new 6 char. name of fg 300 ?> = fall124
 Enter new 6 char. name of fg 301 ?> = fall125
 Enter new 6 char. name of fg 302 ?> = fall126
 Enter new 6 char. name of fg 303 ?> = fall127
 Enter new 6 char. name of fg 304 ?> = fall128
 Enter new 6 char. name of fg 305 ?> = fall129
 Enter new 6 char. name of fg 306 ?> = fall130
 Enter new 6 char. name of fg 307 ?> = fall131
 Enter new 6 char. name of fg 308 ?> = fall132
 Enter new 6 char. name of fg 309 ?> = fall133
 Enter new 6 char. name of fg 310 ?> = fall134
 Enter new 6 char. name of fg 311 ?> = fall135
 Enter new 6 char. name of fg 312 ?> = fall136
 Enter new 6 char. name of fg 313 ?> = fall137
 Enter new 6 char. name of fg 314 ?> = fall138
 Enter new 6 char. name of fg 315 ?> = fall139
 Enter new 6 char. name of fg 316 ?> = fall140
 Enter new 6 char. name of fg 317 ?> = fall141
 Enter new 6 char. name of fg 318 ?> = fall142
 Enter new 6 char. name of fg 319 ?> = fall143
 Enter new 6 char. name of fg 320 ?> = fall144
 Enter new 6 char. name of fg 321 ?> = fall145
 Enter new 6 char. name of fg 322 ?> = fall146
 Enter new 6 char. name of fg 323 ?> = fall147
 Enter new 6 char. name of fg 324 ?> = fall148
 Enter new 6 char. name of fg 325 ?> = fall149
 Enter new 6 char. name of fg 326 ?> = fall150
 Enter new 6 char. name of fg 327 ?> = fall151
 Enter new 6 char. name of fg 328 ?> = fall152
 Enter new 6 char. name of fg 329 ?> = fall153
 Enter new 6 char. name of fg 330 ?> = fall154
 Enter new 6 char. name of fg 331 ?> = fall155
 Enter new 6 char. name of fg 332 ?> = fall156
 Enter new 6 char. name of fg 333 ?> = fall157
 Enter new 6 char. name of fg 334 ?> = fall158
 Enter new 6 char. name of fg 335 ?> = fall159
 Enter new 6 char. name of fg 336 ?> = fall160
 Enter new 6 char. name of fg 337 ?> = fall161
 Enter new 6 char. name of fg 338 ?> = fall162
 Enter new 6 char. name of fg 339 ?> = fall163
 Enter new 6 char. name of fg 340 ?> = fall164
 Enter new 6 char. name of fg 341 ?> = fall165
 Enter new 6 char. name of fg 342 ?> = fall166
 Enter new 6 char. name of fg 343 ?> = fall167
 Enter new 6 char. name of fg 344 ?> = fall168

Enter new 6 char. name of fg 345 ?> = fal169
Enter new 6 char. name of fg 346 ?> = fal170
Enter new 6 char. name of fg 347 ?> = fal171
Enter new 6 char. name of fg 348 ?> = fal172
Enter new 6 char. name of fg 349 ?> = fal173
Enter new 6 char. name of fg 350 ?> = fal174
Enter new 6 char. name of fg 351 ?> = fal175
Enter new 6 char. name of fg 352 ?> = frv0
Enter new 6 char. name of fg 353 ?> = frv1
Enter new 6 char. name of fg 354 ?> = frv2
Enter new 6 char. name of fg 355 ?> = frv3
Enter new 6 char. name of fg 356 ?> = frv4
Enter new 6 char. name of fg 357 ?> = frv5
Enter new 6 char. name of fg 358 ?> = frv6
Enter new 6 char. name of fg 359 ?> = frv7
Enter new 6 char. name of fg 360 ?> = frv8
Enter new 6 char. name of fg 361 ?> = frv9
Enter new 6 char. name of fg 362 ?> = frv10
Enter new 6 char. name of fg 363 ?> = frv11
Enter new 6 char. name of fg 364 ?> = frv12
Enter new 6 char. name of fg 365 ?> = frv13
Enter new 6 char. name of fg 366 ?> = frv14
Enter new 6 char. name of fg 367 ?> = frv15
Enter new 6 char. name of fg 368 ?> = frv16
Enter new 6 char. name of fg 369 ?> = frv17
Enter new 6 char. name of fg 370 ?> = frv18
Enter new 6 char. name of fg 371 ?> = frv19
Enter new 6 char. name of fg 372 ?> = frv20
Enter new 6 char. name of fg 373 ?> = frv21
Enter new 6 char. name of fg 374 ?> = frv22
Enter new 6 char. name of fg 375 ?> = frv23
Enter new 6 char. name of fg 376 ?> = frv24
Enter new 6 char. name of fg 377 ?> = frv25
Enter new 6 char. name of fg 378 ?> = frv26
Enter new 6 char. name of fg 379 ?> = frv27
Enter new 6 char. name of fg 380 ?> = frv28
Enter new 6 char. name of fg 381 ?> = frv29
Enter new 6 char. name of fg 382 ?> = frv30
Enter new 6 char. name of fg 383 ?> = frv31
Enter new 6 char. name of fg 384 ?> = frv32
Enter new 6 char. name of fg 385 ?> = frv33
Enter new 6 char. name of fg 386 ?> = frv34
Enter new 6 char. name of fg 387 ?> = frv35
Enter new 6 char. name of fg 388 ?> = frv36
Enter new 6 char. name of fg 389 ?> = frv37
Enter new 6 char. name of fg 390 ?> = frv38
Enter new 6 char. name of fg 391 ?> = frv39
Enter new 6 char. name of fg 392 ?> = frv40
Enter new 6 char. name of fg 393 ?> = frv41
Enter new 6 char. name of fg 394 ?> = frv42

Enter new 6 char. name of fg 395 ?> = frv43
Enter new 6 char. name of fg 396 ?> = frv44
Enter new 6 char. name of fg 397 ?> = frv45
Enter new 6 char. name of fg 398 ?> = frv46
Enter new 6 char. name of fg 399 ?> = frv47
Enter new 6 char. name of fg 400 ?> = frv48
Enter new 6 char. name of fg 401 ?> = frv49
Enter new 6 char. name of fg 402 ?> = frv50
Enter new 6 char. name of fg 403 ?> = frv51
Enter new 6 char. name of fg 404 ?> = frv52
Enter new 6 char. name of fg 405 ?> = frv53
Enter new 6 char. name of fg 406 ?> = frv54
Enter new 6 char. name of fg 407 ?> = frv55
Enter new 6 char. name of fg 408 ?> = frv56
Enter new 6 char. name of fg 409 ?> = frv57
Enter new 6 char. name of fg 410 ?> = frv58
Enter new 6 char. name of fg 411 ?> = frv59
Enter new 6 char. name of fg 412 ?> = frv60
Enter new 6 char. name of fg 413 ?> = frv61
Enter new 6 char. name of fg 414 ?> = frv62
Enter new 6 char. name of fg 415 ?> = frv63
Enter new 6 char. name of fg 416 ?> = frv64
Enter new 6 char. name of fg 417 ?> = frv65
Enter new 6 char. name of fg 418 ?> = frv66
Enter new 6 char. name of fg 419 ?> = frv67
Enter new 6 char. name of fg 420 ?> = frv68
Enter new 6 char. name of fg 421 ?> = frv69
Enter new 6 char. name of fg 422 ?> = frv70
Enter new 6 char. name of fg 423 ?> = frv71
Enter new 6 char. name of fg 424 ?> = frv72
Enter new 6 char. name of fg 425 ?> = frv73
Enter new 6 char. name of fg 426 ?> = frv74
Enter new 6 char. name of fg 427 ?> = frv75
Enter new 6 char. name of fg 428 ?> = frv76
Enter new 6 char. name of fg 429 ?> = frv77
Enter new 6 char. name of fg 430 ?> = frv78
Enter new 6 char. name of fg 431 ?> = frv79
Enter new 6 char. name of fg 432 ?> = frv80
Enter new 6 char. name of fg 433 ?> = frv81
Enter new 6 char. name of fg 434 ?> = frv82
Enter new 6 char. name of fg 435 ?> = frv83
Enter new 6 char. name of fg 436 ?> = frv84
Enter new 6 char. name of fg 437 ?> = frv85
Enter new 6 char. name of fg 438 ?> = frv86
Enter new 6 char. name of fg 439 ?> = frv87
Enter new 6 char. name of fg 440 ?> = frv88
Enter new 6 char. name of fg 441 ?> = frv89
Enter new 6 char. name of fg 442 ?> = frv90
Enter new 6 char. name of fg 443 ?> = frv91
Enter new 6 char. name of fg 444 ?> = frv92

Enter new 6 char. name of fg 445 ?> = frv93
Enter new 6 char. name of fg 446 ?> = frv94
Enter new 6 char. name of fg 447 ?> = frv95
Enter new 6 char. name of fg 448 ?> = frv96
Enter new 6 char. name of fg 449 ?> = frv97
Enter new 6 char. name of fg 450 ?> = frv98
Enter new 6 char. name of fg 451 ?> = frv99
Enter new 6 char. name of fg 452 ?> = frv100
Enter new 6 char. name of fg 453 ?> = frv101
Enter new 6 char. name of fg 454 ?> = frv102
Enter new 6 char. name of fg 455 ?> = frv103
Enter new 6 char. name of fg 456 ?> = frv104
Enter new 6 char. name of fg 457 ?> = frv105
Enter new 6 char. name of fg 458 ?> = frv106
Enter new 6 char. name of fg 459 ?> = frv107
Enter new 6 char. name of fg 460 ?> = frv108
Enter new 6 char. name of fg 461 ?> = frv109
Enter new 6 char. name of fg 462 ?> = frv110
Enter new 6 char. name of fg 463 ?> = frv111
Enter new 6 char. name of fg 464 ?> = frv112
Enter new 6 char. name of fg 465 ?> = frv113
Enter new 6 char. name of fg 466 ?> = frv114
Enter new 6 char. name of fg 467 ?> = frv115
Enter new 6 char. name of fg 468 ?> = frv116
Enter new 6 char. name of fg 469 ?> = frv117
Enter new 6 char. name of fg 470 ?> = frv118
Enter new 6 char. name of fg 471 ?> = frv119
Enter new 6 char. name of fg 472 ?> = frv120
Enter new 6 char. name of fg 473 ?> = frv121
Enter new 6 char. name of fg 474 ?> = frv122
Enter new 6 char. name of fg 475 ?> = frv123
Enter new 6 char. name of fg 476 ?> = frv124
Enter new 6 char. name of fg 477 ?> = frv125
Enter new 6 char. name of fg 478 ?> = frv126
Enter new 6 char. name of fg 479 ?> = frv127
Enter new 6 char. name of fg 480 ?> = frv128
Enter new 6 char. name of fg 481 ?> = frv129
Enter new 6 char. name of fg 482 ?> = frv130
Enter new 6 char. name of fg 483 ?> = frv131
Enter new 6 char. name of fg 484 ?> = frv132
Enter new 6 char. name of fg 485 ?> = frv133
Enter new 6 char. name of fg 486 ?> = frv134
Enter new 6 char. name of fg 487 ?> = frv135
Enter new 6 char. name of fg 488 ?> = frv136
Enter new 6 char. name of fg 489 ?> = frv137
Enter new 6 char. name of fg 490 ?> = frv138
Enter new 6 char. name of fg 491 ?> = frv139
Enter new 6 char. name of fg 492 ?> = frv140
Enter new 6 char. name of fg 493 ?> = frv141
Enter new 6 char. name of fg 494 ?> = frv142

Enter new 6 char. name of fg 495 ?> = frvl43
 Enter new 6 char. name of fg 496 ?> = frvl44
 Enter new 6 char. name of fg 497 ?> = frvl45
 Enter new 6 char. name of fg 498 ?> = frvl46
 Enter new 6 char. name of fg 499 ?> = frvl47
 Enter new 6 char. name of fg 500 ?> = frvl48
 Enter new 6 char. name of fg 501 ?> = frvl49
 Enter new 6 char. name of fg 502 ?> = frvl50
 Enter new 6 char. name of fg 503 ?> = frvl51
 Enter new 6 char. name of fg 504 ?> = frvl52
 Enter new 6 char. name of fg 505 ?> = frvl53
 Enter new 6 char. name of fg 506 ?> = frvl54
 Enter new 6 char. name of fg 507 ?> = frvl55
 Enter new 6 char. name of fg 508 ?> = frvl56
 Enter new 6 char. name of fg 509 ?> = frvl57
 Enter new 6 char. name of fg 510 ?> = frvl58
 Enter new 6 char. name of fg 511 ?> = frvl59
 Enter new 6 char. name of fg 512 ?> = frvl60
 Enter new 6 char. name of fg 513 ?> = frvl61
 Enter new 6 char. name of fg 514 ?> = frvl62
 Enter new 6 char. name of fg 515 ?> = frvl63
 Enter new 6 char. name of fg 516 ?> = frvl64
 Enter new 6 char. name of fg 517 ?> = frvl65
 Enter new 6 char. name of fg 518 ?> = frvl66
 Enter new 6 char. name of fg 519 ?> = frvl67
 Enter new 6 char. name of fg 520 ?> = frvl68
 Enter new 6 char. name of fg 521 ?> = frvl69
 Enter new 6 char. name of fg 522 ?> = frvl70
 Enter new 6 char. name of fg 523 ?> = frvl71
 Enter new 6 char. name of fg 524 ?> = frvl72
 Enter new 6 char. name of fg 525 ?> = frvl73
 Enter new 6 char. name of fg 526 ?> = frvl74
 Enter new 6 char. name of fg 527 ?> = frvl75
 Enter new 6 char. name of fg 528 ?> = fds0
 Enter new 6 char. name of fg 529 ?> = fds1
 Enter new 6 char. name of fg 530 ?> = fds2
 Enter new 6 char. name of fg 531 ?> = fds3
 Enter new 6 char. name of fg 532 ?> = fds4
 Enter new 6 char. name of fg 533 ?> = fds5
 Enter new 6 char. name of fg 534 ?> = fds6
 Enter new 6 char. name of fg 535 ?> = fds7
 Enter new 6 char. name of fg 536 ?> = fds8
 Enter new 6 char. name of fg 537 ?> = fds9
 Enter new 6 char. name of fg 538 ?> = fds10
 Enter new 6 char. name of fg 539 ?> = fds11
 Enter new 6 char. name of fg 540 ?> = fds12
 Enter new 6 char. name of fg 541 ?> = fds13
 Enter new 6 char. name of fg 542 ?> = fds14
 Enter new 6 char. name of fg 543 ?> = fds15
 Enter new 6 char. name of fg 544 ?> = fds16

WMC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fg 545 ?> = fds17
Enter new 6 char. name of fg 546 ?> = fds18
Enter new 6 char. name of fg 547 ?> = fds19
Enter new 6 char. name of fg 548 ?> = fds20
Enter new 6 char. name of fg 549 ?> = fds21
Enter new 6 char. name of fg 550 ?> = fds22
Enter new 6 char. name of fg 551 ?> = fds23
Enter new 6 char. name of fg 552 ?> = fds24
Enter new 6 char. name of fg 553 ?> = fds25
Enter new 6 char. name of fg 554 ?> = fds26
Enter new 6 char. name of fg 555 ?> = fds27
Enter new 6 char. name of fg 556 ?> = fds28
Enter new 6 char. name of fg 557 ?> = fds29
Enter new 6 char. name of fg 558 ?> = fds30
Enter new 6 char. name of fg 559 ?> = fds31
Enter new 6 char. name of fg 560 ?> = fds32
Enter new 6 char. name of fg 561 ?> = fds33
Enter new 6 char. name of fg 562 ?> = fds34
Enter new 6 char. name of fg 563 ?> = fds35
Enter new 6 char. name of fg 564 ?> = fds36
Enter new 6 char. name of fg 565 ?> = fds37
Enter new 6 char. name of fg 566 ?> = fds38
Enter new 6 char. name of fg 567 ?> = fds39
Enter new 6 char. name of fg 568 ?> = fds40
Enter new 6 char. name of fg 569 ?> = fds41
Enter new 6 char. name of fg 570 ?> = fds42
Enter new 6 char. name of fg 571 ?> = fds43
Enter new 6 char. name of fg 572 ?> = fds44
Enter new 6 char. name of fg 573 ?> = fds45
Enter new 6 char. name of fg 574 ?> = fds46
Enter new 6 char. name of fg 575 ?> = fds47
Enter new 6 char. name of fg 576 ?> = fds48
Enter new 6 char. name of fg 577 ?> = fds49
Enter new 6 char. name of fg 578 ?> = fds50
Enter new 6 char. name of fg 579 ?> = fds51
Enter new 6 char. name of fg 580 ?> = fds52
Enter new 6 char. name of fg 581 ?> = fds53
Enter new 6 char. name of fg 582 ?> = fds54
Enter new 6 char. name of fg 583 ?> = fds55
Enter new 6 char. name of fg 584 ?> = fds56
Enter new 6 char. name of fg 585 ?> = fds57
Enter new 6 char. name of fg 586 ?> = fds58
Enter new 6 char. name of fg 587 ?> = fds59
Enter new 6 char. name of fg 588 ?> = fds60
Enter new 6 char. name of fg 589 ?> = fds61
Enter new 6 char. name of fg 590 ?> = fds62
Enter new 6 char. name of fg 591 ?> = fds63
Enter new 6 char. name of fg 592 ?> = fds64
Enter new 6 char. name of fg 593 ?> = fds65
Enter new 6 char. name of fg 594 ?> = fds66

Enter new 6 char. name of fg 595 ?> = fds67
 Enter new 6 char. name of fg 596 ?> = fds68
 Enter new 6 char. name of fg 597 ?> = fds69
 Enter new 6 char. name of fg 598 ?> = fds70
 Enter new 6 char. name of fg 599 ?> = fds71
 Enter new 6 char. name of fg 600 ?> = fds72
 Enter new 6 char. name of fg 601 ?> = fds73
 Enter new 6 char. name of fg 602 ?> = fds74
 Enter new 6 char. name of fg 603 ?> = fds75
 Enter new 6 char. name of fg 604 ?> = fds76
 Enter new 6 char. name of fg 605 ?> = fds77
 Enter new 6 char. name of fg 606 ?> = fds78
 Enter new 6 char. name of fg 607 ?> = fds79
 Enter new 6 char. name of fg 608 ?> = fds80
 Enter new 6 char. name of fg 609 ?> = fds81
 Enter new 6 char. name of fg 610 ?> = fds82
 Enter new 6 char. name of fg 611 ?> = fds83
 Enter new 6 char. name of fg 612 ?> = fds84
 Enter new 6 char. name of fg 613 ?> = fds85
 Enter new 6 char. name of fg 614 ?> = fds86
 Enter new 6 char. name of fg 615 ?> = fds87
 Enter new 6 char. name of fg 616 ?> = fds88
 Enter new 6 char. name of fg 617 ?> = fds89
 Enter new 6 char. name of fg 618 ?> = fds90
 Enter new 6 char. name of fg 619 ?> = fds91
 Enter new 6 char. name of fg 620 ?> = fds92
 Enter new 6 char. name of fg 621 ?> = fds93
 Enter new 6 char. name of fg 622 ?> = fds94
 Enter new 6 char. name of fg 623 ?> = fds95
 Enter new 6 char. name of fg 624 ?> = fds96
 Enter new 6 char. name of fg 625 ?> = fds97
 Enter new 6 char. name of fg 626 ?> = fds98
 Enter new 6 char. name of fg 627 ?> = fds99
 Enter new 6 char. name of fg 628 ?> = fds100
 Enter new 6 char. name of fg 629 ?> = fds101
 Enter new 6 char. name of fg 630 ?> = fds102
 Enter new 6 char. name of fg 631 ?> = fds103
 Enter new 6 char. name of fg 632 ?> = fds104
 Enter new 6 char. name of fg 633 ?> = fds105
 Enter new 6 char. name of fg 634 ?> = fds106
 Enter new 6 char. name of fg 635 ?> = fds107
 Enter new 6 char. name of fg 636 ?> = fds108
 Enter new 6 char. name of fg 637 ?> = fds109
 Enter new 6 char. name of fg 638 ?> = fds110
 Enter new 6 char. name of fg 639 ?> = fds111
 Enter new 6 char. name of fg 640 ?> = fds112
 Enter new 6 char. name of fg 641 ?> = fds113
 Enter new 6 char. name of fg 642 ?> = fds114
 Enter new 6 char. name of fg 643 ?> = fds115
 Enter new 6 char. name of fg 644 ?> = fds116

Enter new 6 char. name of fg 645 ?> = fds117
 Enter new 6 char. name of fg 646 ?> = fds118
 Enter new 6 char. name of fg 647 ?> = fds119
 Enter new 6 char. name of fg 648 ?> = fds120
 Enter new 6 char. name of fg 649 ?> = fds121
 Enter new 6 char. name of fg 650 ?> = fds122
 Enter new 6 char. name of fg 651 ?> = fds123
 Enter new 6 char. name of fg 652 ?> = fds124
 Enter new 6 char. name of fg 653 ?> = fds125
 Enter new 6 char. name of fg 654 ?> = fds126
 Enter new 6 char. name of fg 655 ?> = fds127
 Enter new 6 char. name of fg 656 ?> = fds128
 Enter new 6 char. name of fg 657 ?> = fds129
 Enter new 6 char. name of fg 658 ?> = fds130
 Enter new 6 char. name of fg 659 ?> = fds131
 Enter new 6 char. name of fg 660 ?> = fds132
 Enter new 6 char. name of fg 661 ?> = fds133
 Enter new 6 char. name of fg 662 ?> = fds134
 Enter new 6 char. name of fg 663 ?> = fds135
 Enter new 6 char. name of fg 664 ?> = fds136
 Enter new 6 char. name of fg 665 ?> = fds137
 Enter new 6 char. name of fg 666 ?> = fds138
 Enter new 6 char. name of fg 667 ?> = fds139
 Enter new 6 char. name of fg 668 ?> = fds140
 Enter new 6 char. name of fg 669 ?> = fds141
 Enter new 6 char. name of fg 670 ?> = fds142
 Enter new 6 char. name of fg 671 ?> = fds143
 Enter new 6 char. name of fg 672 ?> = fds144
 Enter new 6 char. name of fg 673 ?> = fds145
 Enter new 6 char. name of fg 674 ?> = fds146
 Enter new 6 char. name of fg 675 ?> = fds147
 Enter new 6 char. name of fg 676 ?> = fds148
 Enter new 6 char. name of fg 677 ?> = fds149
 Enter new 6 char. name of fg 678 ?> = fds150
 Enter new 6 char. name of fg 679 ?> = fds151
 Enter new 6 char. name of fg 680 ?> = fds152
 Enter new 6 char. name of fg 681 ?> = fds153
 Enter new 6 char. name of fg 682 ?> = fds154
 Enter new 6 char. name of fg 683 ?> = fds155
 Enter new 6 char. name of fg 684 ?> = fds156
 Enter new 6 char. name of fg 685 ?> = fds157
 Enter new 6 char. name of fg 686 ?> = fds158
 Enter new 6 char. name of fg 687 ?> = fds159
 Enter new 6 char. name of fg 688 ?> = fds160
 Enter new 6 char. name of fg 689 ?> = fds161
 Enter new 6 char. name of fg 690 ?> = fds162
 Enter new 6 char. name of fg 691 ?> = fds163
 Enter new 6 char. name of fg 692 ?> = fds164
 Enter new 6 char. name of fg 693 ?> = fds165
 Enter new 6 char. name of fg 694 ?> = fds166

```

Enter new 6 char. name of fg 695 ?> = fds167
Enter new 6 char. name of fg 696 ?> = fds168
Enter new 6 char. name of fg 697 ?> = fds169
Enter new 6 char. name of fg 698 ?> = fds170
Enter new 6 char. name of fg 699 ?> = fds171
Enter new 6 char. name of fg 700 ?> = fds172
Enter new 6 char. name of fg 701 ?> = fds173
Enter new 6 char. name of fg 702 ?> = fds174
Enter new 6 char. name of fg 703 ?> = fds175
Enter new 6 char. name of fg 704 ?> = fdisp
Enter new 6 char. name of fg 705 ?> = fhold
Enter new 6 char. name of fg 706 ?> = fning
Enter new 6 char. name of fg 707 ?> = fabort
Enter new 6 char. name of fg 708 ?> = fstart
Enter new 6 char. name of fg 709 ?> = ftrip0
Enter new 6 char. name of fg 710 ?> = ftrip1
Enter new 6 char. name of fg 711 ?> = ftrip2
Enter new 6 char. name of fg 712 ?> = fspar0
Enter new 6 char. name of fg 713 ?> = fspar1
Enter new 6 char. name of fg 714 ?> = fspar2
Enter new 6 char. name of fg 715 ?> = fspar3
Enter new 6 char. name of fg 716 ?> = fspar4
Enter new 6 char. name of fg 717 ?> = fspar5
Enter new 6 char. name of fg 718 ?> = fspar6
Enter new 6 char. name of fg 719 ?> = fspar7
MUPF?> = add 367 in
Enter new 6 char. name of in 0 ?> = bitval
Enter new 6 char. name of in 1 ?> = iloop
Enter new 6 char. name of in 2 ?> = jloop
Enter new 6 char. name of in 3 ?> = kloop
Enter new 6 char. name of in 4 ?> = lloop
Enter new 6 char. name of in 5 ?> = index
Enter new 6 char. name of in 6 ?> = trip
Enter new 6 char. name of in 7 ?> = nadi
Enter new 6 char. name of in 8 ?> = blkval
Enter new 6 char. name of in 9 ?> = sort
Enter new 6 char. name of in 10 ?> = bval
Enter new 6 char. name of in 11 ?> = hldstp
Enter new 6 char. name of in 12 ?> = remain
Enter new 6 char. name of in 13 ?> = errn
Enter new 6 char. name of in 14 ?> = errl
Enter new 6 char. name of in 15 ?> = int0
Enter new 6 char. name of in 16 ?> = int1
Enter new 6 char. name of in 17 ?> = int2
Enter new 6 char. name of in 18 ?> = int3
Enter new 6 char. name of in 19 ?> = int4
Enter new 6 char. name of in 20 ?> = int5
Enter new 6 char. name of in 21 ?> = int6
Enter new 6 char. name of in 22 ?> = int7
Enter new 6 char. name of in 23 ?> = int8

```

Enter new 6 char. name of in 24 ?> = int9
 Enter new 6 char. name of in 25 ?> = int10
 Enter new 6 char. name of in 26 ?> = int11
 Enter new 6 char. name of in 27 ?> = int12
 Enter new 6 char. name of in 28 ?> = int13
 Enter new 6 char. name of in 29 ?> = int14
 Enter new 6 char. name of in 30 ?> = int15
 Enter new 6 char. name of in 31 ?> = int16
 Enter new 6 char. name of in 32 ?> = int17
 Enter new 6 char. name of in 33 ?> = int18
 Enter new 6 char. name of in 34 ?> = int19
 Enter new 6 char. name of in 35 ?> = int20
 Enter new 6 char. name of in 36 ?> = int21
 Enter new 6 char. name of in 37 ?> = int22
 Enter new 6 char. name of in 38 ?> = int23
 Enter new 6 char. name of in 39 ?> = int24
 Enter new 6 char. name of in 40 ?> = int25
 Enter new 6 char. name of in 41 ?> = int26
 Enter new 6 char. name of in 42 ?> = int27
 Enter new 6 char. name of in 43 ?> = int28
 Enter new 6 char. name of in 44 ?> = int29
 Enter new 6 char. name of in 45 ?> = int30
 Enter new 6 char. name of in 46 ?> = int31
 Enter new 6 char. name of in 47 ?> = int32
 Enter new 6 char. name of in 48 ?> = int33
 Enter new 6 char. name of in 49 ?> = int34
 Enter new 6 char. name of in 50 ?> = int35
 Enter new 6 char. name of in 51 ?> = int36
 Enter new 6 char. name of in 52 ?> = int37
 Enter new 6 char. name of in 53 ?> = int38
 Enter new 6 char. name of in 54 ?> = int39
 Enter new 6 char. name of in 55 ?> = int40
 Enter new 6 char. name of in 56 ?> = int41
 Enter new 6 char. name of in 57 ?> = int42
 Enter new 6 char. name of in 58 ?> = int43
 Enter new 6 char. name of in 59 ?> = int44
 Enter new 6 char. name of in 60 ?> = int45
 Enter new 6 char. name of in 61 ?> = int46
 Enter new 6 char. name of in 62 ?> = int47
 Enter new 6 char. name of in 63 ?> = int48
 Enter new 6 char. name of in 64 ?> = int49
 Enter new 6 char. name of in 65 ?> = int50
 Enter new 6 char. name of in 66 ?> = int51
 Enter new 6 char. name of in 67 ?> = int52
 Enter new 6 char. name of in 68 ?> = int53
 Enter new 6 char. name of in 69 ?> = int54
 Enter new 6 char. name of in 70 ?> = int55
 Enter new 6 char. name of in 71 ?> = int56
 Enter new 6 char. name of in 72 ?> = int57
 Enter new 6 char. name of in 73 ?> = int58

Enter new 6 char. name of in 74 ?> = int59
 Enter new 6 char. name of in 75 ?> = int60
 Enter new 6 char. name of in 76 ?> = int61
 Enter new 6 char. name of in 77 ?> = int62
 Enter new 6 char. name of in 78 ?> = int63
 Enter new 6 char. name of in 79 ?> = int64
 Enter new 6 char. name of in 80 ?> = int65
 Enter new 6 char. name of in 81 ?> = int66
 Enter new 6 char. name of in 82 ?> = int67
 Enter new 6 char. name of in 83 ?> = int68
 Enter new 6 char. name of in 84 ?> = int69
 Enter new 6 char. name of in 85 ?> = int70
 Enter new 6 char. name of in 86 ?> = int71
 Enter new 6 char. name of in 87 ?> = int72
 Enter new 6 char. name of in 88 ?> = int73
 Enter new 6 char. name of in 89 ?> = int74
 Enter new 6 char. name of in 90 ?> = int75
 Enter new 6 char. name of in 91 ?> = int76
 Enter new 6 char. name of in 92 ?> = int77
 Enter new 6 char. name of in 93 ?> = int78
 Enter new 6 char. name of in 94 ?> = int79
 Enter new 6 char. name of in 95 ?> = int80
 Enter new 6 char. name of in 96 ?> = int81
 Enter new 6 char. name of in 97 ?> = int82
 Enter new 6 char. name of in 98 ?> = int83
 Enter new 6 char. name of in 99 ?> = int84
 Enter new 6 char. name of in 100 ?> = int85
 Enter new 6 char. name of in 101 ?> = int86
 Enter new 6 char. name of in 102 ?> = int87
 Enter new 6 char. name of in 103 ?> = int88
 Enter new 6 char. name of in 104 ?> = int89
 Enter new 6 char. name of in 105 ?> = int90
 Enter new 6 char. name of in 106 ?> = int91
 Enter new 6 char. name of in 107 ?> = int92
 Enter new 6 char. name of in 108 ?> = int93
 Enter new 6 char. name of in 109 ?> = int94
 Enter new 6 char. name of in 110 ?> = int95
 Enter new 6 char. name of in 111 ?> = int96
 Enter new 6 char. name of in 112 ?> = int97
 Enter new 6 char. name of in 113 ?> = int98
 Enter new 6 char. name of in 114 ?> = int99
 Enter new 6 char. name of in 115 ?> = sta100
 Enter new 6 char. name of in 116 ?> = sta101
 Enter new 6 char. name of in 117 ?> = sta102
 Enter new 6 char. name of in 118 ?> = sta103
 Enter new 6 char. name of in 119 ?> = sta104
 Enter new 6 char. name of in 120 ?> = sta105
 Enter new 6 char. name of in 121 ?> = sta106
 Enter new 6 char. name of in 122 ?> = sta107
 Enter new 6 char. name of in 123 ?> = sta108

Enter new 6 char. name of in 124 ?> = sta109
 Enter new 6 char. name of in 125 ?> = sta110
 Enter new 6 char. name of in 126 ?> = sta111
 Enter new 6 char. name of in 127 ?> = sta112
 Enter new 6 char. name of in 128 ?> = sta113
 Enter new 6 char. name of in 129 ?> = sta114
 Enter new 6 char. name of in 130 ?> = sta115
 Enter new 6 char. name of in 131 ?> = sta116
 Enter new 6 char. name of in 132 ?> = sta117
 Enter new 6 char. name of in 133 ?> = sta118
 Enter new 6 char. name of in 134 ?> = sta119
 Enter new 6 char. name of in 135 ?> = sta120
 Enter new 6 char. name of in 136 ?> = sta121
 Enter new 6 char. name of in 137 ?> = sta122
 Enter new 6 char. name of in 138 ?> = sta123
 Enter new 6 char. name of in 139 ?> = sta124
 Enter new 6 char. name of in 140 ?> = sta125
 Enter new 6 char. name of in 141 ?> = sta126
 Enter new 6 char. name of in 142 ?> = sta127
 Enter new 6 char. name of in 143 ?> = sta128
 Enter new 6 char. name of in 144 ?> = sta129
 Enter new 6 char. name of in 145 ?> = sta130
 Enter new 6 char. name of in 146 ?> = sta131
 Enter new 6 char. name of in 147 ?> = sta132
 Enter new 6 char. name of in 148 ?> = sta133
 Enter new 6 char. name of in 149 ?> = sta134
 Enter new 6 char. name of in 150 ?> = sta135
 Enter new 6 char. name of in 151 ?> = sta136
 Enter new 6 char. name of in 152 ?> = sta137
 Enter new 6 char. name of in 153 ?> = sta138
 Enter new 6 char. name of in 154 ?> = sta139
 Enter new 6 char. name of in 155 ?> = sta140
 Enter new 6 char. name of in 156 ?> = sta141
 Enter new 6 char. name of in 157 ?> = sta142
 Enter new 6 char. name of in 158 ?> = sta143
 Enter new 6 char. name of in 159 ?> = sta144
 Enter new 6 char. name of in 160 ?> = sta145
 Enter new 6 char. name of in 161 ?> = sta146
 Enter new 6 char. name of in 162 ?> = sta147
 Enter new 6 char. name of in 163 ?> = sta148
 Enter new 6 char. name of in 164 ?> = sta149
 Enter new 6 char. name of in 165 ?> = sta150
 Enter new 6 char. name of in 166 ?> = sta151
 Enter new 6 char. name of in 167 ?> = sta152
 Enter new 6 char. name of in 168 ?> = sta153
 Enter new 6 char. name of in 169 ?> = sta154
 Enter new 6 char. name of in 170 ?> = sta155
 Enter new 6 char. name of in 171 ?> = sta156
 Enter new 6 char. name of in 172 ?> = sta157
 Enter new 6 char. name of in 173 ?> = sta158

Enter new 6 char. name of in 174 ?> = stal59
Enter new 6 char. name of in 175 ?> = stal60
Enter new 6 char. name of in 176 ?> = stal61
Enter new 6 char. name of in 177 ?> = stal62
Enter new 6 char. name of in 178 ?> = stal63
Enter new 6 char. name of in 179 ?> = stal64
Enter new 6 char. name of in 180 ?> = stal65
Enter new 6 char. name of in 181 ?> = stal66
Enter new 6 char. name of in 182 ?> = stal67
Enter new 6 char. name of in 183 ?> = stal68
Enter new 6 char. name of in 184 ?> = stal69
Enter new 6 char. name of in 185 ?> = stal70
Enter new 6 char. name of in 186 ?> = stal71
Enter new 6 char. name of in 187 ?> = stal72
Enter new 6 char. name of in 188 ?> = stal73
Enter new 6 char. name of in 189 ?> = stal74
Enter new 6 char. name of in 190 ?> = stal75
Enter new 6 char. name of in 191 ?> = type0
Enter new 6 char. name of in 192 ?> = type1
Enter new 6 char. name of in 193 ?> = type2
Enter new 6 char. name of in 194 ?> = type3
Enter new 6 char. name of in 195 ?> = type4
Enter new 6 char. name of in 196 ?> = type5
Enter new 6 char. name of in 197 ?> = type6
Enter new 6 char. name of in 198 ?> = type7
Enter new 6 char. name of in 199 ?> = type8
Enter new 6 char. name of in 200 ?> = type9
Enter new 6 char. name of in 201 ?> = type10
Enter new 6 char. name of in 202 ?> = type11
Enter new 6 char. name of in 203 ?> = type12
Enter new 6 char. name of in 204 ?> = type13
Enter new 6 char. name of in 205 ?> = type14
Enter new 6 char. name of in 206 ?> = type15
Enter new 6 char. name of in 207 ?> = type16
Enter new 6 char. name of in 208 ?> = type17
Enter new 6 char. name of in 209 ?> = type18
Enter new 6 char. name of in 210 ?> = type19
Enter new 6 char. name of in 211 ?> = type20
Enter new 6 char. name of in 212 ?> = type21
Enter new 6 char. name of in 213 ?> = type22
Enter new 6 char. name of in 214 ?> = type23
Enter new 6 char. name of in 215 ?> = type24
Enter new 6 char. name of in 216 ?> = type25
Enter new 6 char. name of in 217 ?> = type26
Enter new 6 char. name of in 218 ?> = type27
Enter new 6 char. name of in 219 ?> = type28
Enter new 6 char. name of in 220 ?> = type29
Enter new 6 char. name of in 221 ?> = type30
Enter new 6 char. name of in 222 ?> = type31
Enter new 6 char. name of in 223 ?> = type32

Enter new 6 char. name of in 224 ?> = type33
 Enter new 6 char. name of in 225 ?> = type34
 Enter new 6 char. name of in 226 ?> = type35
 Enter new 6 char. name of in 227 ?> = type36
 Enter new 6 char. name of in 228 ?> = type37
 Enter new 6 char. name of in 229 ?> = type38
 Enter new 6 char. name of in 230 ?> = type39
 Enter new 6 char. name of in 231 ?> = type40
 Enter new 6 char. name of in 232 ?> = type41
 Enter new 6 char. name of in 233 ?> = type42
 Enter new 6 char. name of in 234 ?> = type43
 Enter new 6 char. name of in 235 ?> = type44
 Enter new 6 char. name of in 236 ?> = type45
 Enter new 6 char. name of in 237 ?> = type46
 Enter new 6 char. name of in 238 ?> = type47
 Enter new 6 char. name of in 239 ?> = type48
 Enter new 6 char. name of in 240 ?> = type49
 Enter new 6 char. name of in 241 ?> = type50
 Enter new 6 char. name of in 242 ?> = type51
 Enter new 6 char. name of in 243 ?> = type52
 Enter new 6 char. name of in 244 ?> = type53
 Enter new 6 char. name of in 245 ?> = type54
 Enter new 6 char. name of in 246 ?> = type55
 Enter new 6 char. name of in 247 ?> = type56
 Enter new 6 char. name of in 248 ?> = type57
 Enter new 6 char. name of in 249 ?> = type58
 Enter new 6 char. name of in 250 ?> = type59
 Enter new 6 char. name of in 251 ?> = type60
 Enter new 6 char. name of in 252 ?> = type61
 Enter new 6 char. name of in 253 ?> = type62
 Enter new 6 char. name of in 254 ?> = type63
 Enter new 6 char. name of in 255 ?> = type64
 Enter new 6 char. name of in 256 ?> = type65
 Enter new 6 char. name of in 257 ?> = type66
 Enter new 6 char. name of in 258 ?> = type67
 Enter new 6 char. name of in 259 ?> = type68
 Enter new 6 char. name of in 260 ?> = type69
 Enter new 6 char. name of in 261 ?> = type70
 Enter new 6 char. name of in 262 ?> = type71
 Enter new 6 char. name of in 263 ?> = type72
 Enter new 6 char. name of in 264 ?> = type73
 Enter new 6 char. name of in 265 ?> = type74
 Enter new 6 char. name of in 266 ?> = type75
 Enter new 6 char. name of in 267 ?> = type76
 Enter new 6 char. name of in 268 ?> = type77
 Enter new 6 char. name of in 269 ?> = type78
 Enter new 6 char. name of in 270 ?> = type79
 Enter new 6 char. name of in 271 ?> = type80
 Enter new 6 char. name of in 272 ?> = type81
 Enter new 6 char. name of in 273 ?> = type82

Enter new 6 char. name of in 274 ?> = type83
 Enter new 6 char. name of in 275 ?> = type84
 Enter new 6 char. name of in 276 ?> = type85
 Enter new 6 char. name of in 277 ?> = type86
 Enter new 6 char. name of in 278 ?> = type87
 Enter new 6 char. name of in 279 ?> = type88
 Enter new 6 char. name of in 280 ?> = type89
 Enter new 6 char. name of in 281 ?> = type90
 Enter new 6 char. name of in 282 ?> = type91
 Enter new 6 char. name of in 283 ?> = type92
 Enter new 6 char. name of in 284 ?> = type93
 Enter new 6 char. name of in 285 ?> = type94
 Enter new 6 char. name of in 286 ?> = type95
 Enter new 6 char. name of in 287 ?> = type96
 Enter new 6 char. name of in 288 ?> = type97
 Enter new 6 char. name of in 289 ?> = type98
 Enter new 6 char. name of in 290 ?> = type99
 Enter new 6 char. name of in 291 ?> = typ100
 Enter new 6 char. name of in 292 ?> = typ101
 Enter new 6 char. name of in 293 ?> = typ102
 Enter new 6 char. name of in 294 ?> = typ103
 Enter new 6 char. name of in 295 ?> = typ104
 Enter new 6 char. name of in 296 ?> = typ105
 Enter new 6 char. name of in 297 ?> = typ106
 Enter new 6 char. name of in 298 ?> = typ107
 Enter new 6 char. name of in 299 ?> = typ108
 Enter new 6 char. name of in 300 ?> = typ109
 Enter new 6 char. name of in 301 ?> = typ110
 Enter new 6 char. name of in 302 ?> = typ111
 Enter new 6 char. name of in 303 ?> = typ112
 Enter new 6 char. name of in 304 ?> = typ113
 Enter new 6 char. name of in 305 ?> = typ114
 Enter new 6 char. name of in 306 ?> = typ115
 Enter new 6 char. name of in 307 ?> = typ116
 Enter new 6 char. name of in 308 ?> = typ117
 Enter new 6 char. name of in 309 ?> = typ118
 Enter new 6 char. name of in 310 ?> = typ119
 Enter new 6 char. name of in 311 ?> = typ120
 Enter new 6 char. name of in 312 ?> = typ121
 Enter new 6 char. name of in 313 ?> = typ122
 Enter new 6 char. name of in 314 ?> = typ123
 Enter new 6 char. name of in 315 ?> = typ124
 Enter new 6 char. name of in 316 ?> = typ125
 Enter new 6 char. name of in 317 ?> = typ126
 Enter new 6 char. name of in 318 ?> = typ127
 Enter new 6 char. name of in 319 ?> = typ128
 Enter new 6 char. name of in 320 ?> = typ129
 Enter new 6 char. name of in 321 ?> = typ130
 Enter new 6 char. name of in 322 ?> = typ131
 Enter new 6 char. name of in 323 ?> = typ132


```

Enter new 6 char. name of in 324 ?> = typ133
Enter new 6 char. name of in 325 ?> = typ134
Enter new 6 char. name of in 326 ?> = typ135
Enter new 6 char. name of in 327 ?> = typ136
Enter new 6 char. name of in 328 ?> = typ137
Enter new 6 char. name of in 329 ?> = typ138
Enter new 6 char. name of in 330 ?> = typ139
Enter new 6 char. name of in 331 ?> = typ140
Enter new 6 char. name of in 332 ?> = typ141
Enter new 6 char. name of in 333 ?> = typ142
Enter new 6 char. name of in 334 ?> = typ143
Enter new 6 char. name of in 335 ?> = typ144
Enter new 6 char. name of in 336 ?> = typ145
Enter new 6 char. name of in 337 ?> = typ146
Enter new 6 char. name of in 338 ?> = typ147
Enter new 6 char. name of in 339 ?> = typ148
Enter new 6 char. name of in 340 ?> = typ149
Enter new 6 char. name of in 341 ?> = typ150
Enter new 6 char. name of in 342 ?> = typ151
Enter new 6 char. name of in 343 ?> = typ152
Enter new 6 char. name of in 344 ?> = typ153
Enter new 6 char. name of in 345 ?> = typ154
Enter new 6 char. name of in 346 ?> = typ155
Enter new 6 char. name of in 347 ?> = typ156
Enter new 6 char. name of in 348 ?> = typ157
Enter new 6 char. name of in 349 ?> = typ158
Enter new 6 char. name of in 350 ?> = typ159
Enter new 6 char. name of in 351 ?> = typ160
Enter new 6 char. name of in 352 ?> = typ161
Enter new 6 char. name of in 353 ?> = typ162
Enter new 6 char. name of in 354 ?> = typ163
Enter new 6 char. name of in 355 ?> = typ164
Enter new 6 char. name of in 356 ?> = typ165
Enter new 6 char. name of in 357 ?> = typ166
Enter new 6 char. name of in 358 ?> = typ167
Enter new 6 char. name of in 359 ?> = typ168
Enter new 6 char. name of in 360 ?> = typ169
Enter new 6 char. name of in 361 ?> = typ170
Enter new 6 char. name of in 362 ?> = typ171
Enter new 6 char. name of in 363 ?> = typ172
Enter new 6 char. name of in 364 ?> = typ173
Enter new 6 char. name of in 365 ?> = typ174
Enter new 6 char. name of in 366 ?> = typ175
MUPF?> = add 50 fl
Enter new 6 char. name of fl 0 ?> = trip0
Enter new 6 char. name of fl 1 ?> = trip1
Enter new 6 char. name of fl 2 ?> = trip2
Enter new 6 char. name of fl 3 ?> = trip3
Enter new 6 char. name of fl 4 ?> = trip4
Enter new 6 char. name of fl 5 ?> = trip5

```

```

Enter new 6 char. name of fl 6 ?> = trip6
Enter new 6 char. name of fl 7 ?> = trip7
Enter new 6 char. name of fl 8 ?> = trip8
Enter new 6 char. name of fl 9 ?> = trip9
Enter new 6 char. name of fl 10 ?> = ai0
Enter new 6 char. name of fl 11 ?> = ai1
Enter new 6 char. name of fl 12 ?> = ai2
Enter new 6 char. name of fl 13 ?> = ai3
Enter new 6 char. name of fl 14 ?> = ai4
Enter new 6 char. name of fl 15 ?> = ai5
Enter new 6 char. name of fl 16 ?> = ai6
Enter new 6 char. name of fl 17 ?> = ai7
Enter new 6 char. name of fl 18 ?> = ai8
Enter new 6 char. name of fl 19 ?> = ai9
Enter new 6 char. name of fl 20 ?> = ai10
Enter new 6 char. name of fl 21 ?> = ai11
Enter new 6 char. name of fl 22 ?> = ai12
Enter new 6 char. name of fl 23 ?> = ai13
Enter new 6 char. name of fl 24 ?> = ai14
Enter new 6 char. name of fl 25 ?> = ai15
Enter new 6 char. name of fl 26 ?> = ai16
Enter new 6 char. name of fl 27 ?> = ai17
Enter new 6 char. name of fl 28 ?> = ai18
Enter new 6 char. name of fl 29 ?> = ai19
Enter new 6 char. name of fl 30 ?> = ai20
Enter new 6 char. name of fl 31 ?> = ai21
Enter new 6 char. name of fl 32 ?> = ai22
Enter new 6 char. name of fl 33 ?> = ai23
Enter new 6 char. name of fl 34 ?> = ai24
Enter new 6 char. name of fl 35 ?> = ai25
Enter new 6 char. name of fl 36 ?> = ai26
Enter new 6 char. name of fl 37 ?> = ai27
Enter new 6 char. name of fl 38 ?> = ai28
Enter new 6 char. name of fl 39 ?> = ai29
Enter new 6 char. name of fl 40 ?> = ai30
Enter new 6 char. name of fl 41 ?> = ai31
Enter new 6 char. name of fl 42 ?> = ai32
Enter new 6 char. name of fl 43 ?> = ai33
Enter new 6 char. name of fl 44 ?> = ai34
Enter new 6 char. name of fl 45 ?> = ai35
Enter new 6 char. name of fl 46 ?> = ai36
Enter new 6 char. name of fl 47 ?> = ai37
Enter new 6 char. name of fl 48 ?> = ai38
Enter new 6 char. name of fl 49 ?> = ai39
MUPF?> = add 12 pt
Enter new 6 char. name of pt 0 ?> = pepn
Enter new 6 char. name of pt 1 ?> = pdev
Enter new 6 char. name of pt 2 ?> = pdin
Enter new 6 char. name of pt 3 ?> = pai
Enter new 6 char. name of pt 4 ?> = ptrip

```

```

Enter new 6 char. name of pt 5 ?> = pack
Enter new 6 char. name of pt 6 ?> = pdsabl
Enter new 6 char. name of pt 7 ?> = preps
Enter new 6 char. name of pt 8 ?> = pskid
Enter new 6 char. name of pt 9 ?> = pstat
Enter new 6 char. name of pt 10 ?> = ptype
Enter new 6 char. name of pt 11 ?> = prev
MUPF?> = add 25 sg
Enter new 6 char. name of sg 0 ?> = date0
Enter max string length for sg 0 ?> = 10
Enter new 6 char. name of sg 1 ?> = datel
Enter max string length for sg 1 ?> = 10
Enter new 6 char. name of sg 2 ?> = date2
Enter max string length for sg 2 ?> = 10
Enter new 6 char. name of sg 3 ?> = date3
Enter max string length for sg 3 ?> = 10
Enter new 6 char. name of sg 4 ?> = date4
Enter max string length for sg 4 ?> = 10
Enter new 6 char. name of sg 5 ?> = time0
Enter max string length for sg 5 ?> = 8
Enter new 6 char. name of sg 6 ?> = time1
Enter max string length for sg 6 ?> = 8
Enter new 6 char. name of sg 7 ?> = time2
Enter max string length for sg 7 ?> = 8
Enter new 6 char. name of sg 8 ?> = time3
Enter max string length for sg 8 ?> = 8
Enter new 6 char. name of sg 9 ?> = time4
Enter max string length for sg 9 ?> = 8
Enter new 6 char. name of sg 10 ?> = epn0
Enter max string length for sg 10 ?> = 10
Enter new 6 char. name of sg 11 ?> = epn1
Enter max string length for sg 11 ?> = 10
Enter new 6 char. name of sg 12 ?> = epn2
Enter max string length for sg 12 ?> = 10
Enter new 6 char. name of sg 13 ?> = epn3
Enter max string length for sg 13 ?> = 10
Enter new 6 char. name of sg 14 ?> = epn4
Enter max string length for sg 14 ?> = 10
Enter new 6 char. name of sg 15 ?> = mssg0
Enter max string length for sg 15 ?> = 30
Enter new 6 char. name of sg 16 ?> = mssg1
Enter max string length for sg 16 ?> = 30
Enter new 6 char. name of sg 17 ?> = mssg2
Enter max string length for sg 17 ?> = 30
Enter new 6 char. name of sg 18 ?> = mssg3
Enter max string length for sg 18 ?> = 30
Enter new 6 char. name of sg 19 ?> = mssg4
Enter max string length for sg 19 ?> = 30
Enter new 6 char. name of sg 20 ?> = lsg0
Enter max string length for sg 20 ?> = 30

```

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of sg 21 ?> = lsg1
Enter max string length for sg 21 ?> = 30
Enter new 6 char. name of sg 22 ?> = lsg2
Enter max string length for sg 22 ?> = 30
Enter new 6 char. name of sg 23 ?> = lsg3
Enter max string length for sg 23 ?> = 30
Enter new 6 char. name of sg 24 ?> = lsg4
Enter max string length for sg 24 ?> = 30
MUPF?>
    end of file encountered
UPF?>
    end of file encountered

TOTAL1

```

UPF Version 6.3-2      05-Dec-94 09:39:28
UPF?> = de totall
UPF?> = cr totall,*
Enter 4 character name of RPF to associate?> = all1
MUPF?> = add 784 fg
Enter new 6 char. name of fg 0 ?> = fds0
Enter new 6 char. name of fg 1 ?> = fds1
Enter new 6 char. name of fg 2 ?> = fds2
Enter new 6 char. name of fg 3 ?> = fds3
Enter new 6 char. name of fg 4 ?> = fds4
Enter new 6 char. name of fg 5 ?> = fds5
Enter new 6 char. name of fg 6 ?> = fds6
Enter new 6 char. name of fg 7 ?> = fds7
Enter new 6 char. name of fg 8 ?> = fds8
Enter new 6 char. name of fg 9 ?> = fds9
Enter new 6 char. name of fg 10 ?> = fds10
Enter new 6 char. name of fg 11 ?> = fds11
Enter new 6 char. name of fg 12 ?> = fds12
Enter new 6 char. name of fg 13 ?> = fds13
Enter new 6 char. name of fg 14 ?> = fds14
Enter new 6 char. name of fg 15 ?> = fds15
Enter new 6 char. name of fg 16 ?> = fds16
Enter new 6 char. name of fg 17 ?> = fds17
Enter new 6 char. name of fg 18 ?> = fds18
Enter new 6 char. name of fg 19 ?> = fds19
Enter new 6 char. name of fg 20 ?> = fds20
Enter new 6 char. name of fg 21 ?> = fds21
Enter new 6 char. name of fg 22 ?> = fds22
Enter new 6 char. name of fg 23 ?> = fds23
Enter new 6 char. name of fg 24 ?> = fds24
Enter new 6 char. name of fg 25 ?> = fds25
Enter new 6 char. name of fg 26 ?> = fds26
Enter new 6 char. name of fg 27 ?> = fds27
Enter new 6 char. name of fg 28 ?> = fds28
Enter new 6 char. name of fg 29 ?> = fds29
Enter new 6 char. name of fg 30 ?> = fds30
Enter new 6 char. name of fg 31 ?> = fds31
Enter new 6 char. name of fg 32 ?> = fds32
Enter new 6 char. name of fg 33 ?> = fds33
Enter new 6 char. name of fg 34 ?> = fds34
Enter new 6 char. name of fg 35 ?> = fds35
Enter new 6 char. name of fg 36 ?> = fds36
Enter new 6 char. name of fg 37 ?> = fds37
Enter new 6 char. name of fg 38 ?> = fds38
Enter new 6 char. name of fg 39 ?> = fds39
Enter new 6 char. name of fg 40 ?> = fds40
Enter new 6 char. name of fg 41 ?> = fds41
Enter new 6 char. name of fg 42 ?> = fds42
Enter new 6 char. name of fg 43 ?> = fds43
Enter new 6 char. name of fg 44 ?> = fds44

```

Enter new 6 char. name of fg 45 ?> = fds45
Enter new 6 char. name of fg 46 ?> = fds46
Enter new 6 char. name of fg 47 ?> = fds47
Enter new 6 char. name of fg 48 ?> = fds48
Enter new 6 char. name of fg 49 ?> = fds49
Enter new 6 char. name of fg 50 ?> = fds50
Enter new 6 char. name of fg 51 ?> = fds51
Enter new 6 char. name of fg 52 ?> = fds52
Enter new 6 char. name of fg 53 ?> = fds53
Enter new 6 char. name of fg 54 ?> = fds54
Enter new 6 char. name of fg 55 ?> = fds55
Enter new 6 char. name of fg 56 ?> = fds56
Enter new 6 char. name of fg 57 ?> = fds57
Enter new 6 char. name of fg 58 ?> = fds58
Enter new 6 char. name of fg 59 ?> = fds59
Enter new 6 char. name of fg 60 ?> = fds60
Enter new 6 char. name of fg 61 ?> = fds61
Enter new 6 char. name of fg 62 ?> = fds62
Enter new 6 char. name of fg 63 ?> = fds63
Enter new 6 char. name of fg 64 ?> = fds64
Enter new 6 char. name of fg 65 ?> = fds65
Enter new 6 char. name of fg 66 ?> = fds66
Enter new 6 char. name of fg 67 ?> = fds67
Enter new 6 char. name of fg 68 ?> = fds68
Enter new 6 char. name of fg 69 ?> = fds69
Enter new 6 char. name of fg 70 ?> = fds70
Enter new 6 char. name of fg 71 ?> = fds71
Enter new 6 char. name of fg 72 ?> = fds72
Enter new 6 char. name of fg 73 ?> = fds73
Enter new 6 char. name of fg 74 ?> = fds74
Enter new 6 char. name of fg 75 ?> = fds75
Enter new 6 char. name of fg 76 ?> = fds76
Enter new 6 char. name of fg 77 ?> = fds77
Enter new 6 char. name of fg 78 ?> = fds78
Enter new 6 char. name of fg 79 ?> = fds79
Enter new 6 char. name of fg 80 ?> = fds80
Enter new 6 char. name of fg 81 ?> = fds81
Enter new 6 char. name of fg 82 ?> = fds82
Enter new 6 char. name of fg 83 ?> = fds83
Enter new 6 char. name of fg 84 ?> = fds84
Enter new 6 char. name of fg 85 ?> = fds85
Enter new 6 char. name of fg 86 ?> = fds86
Enter new 6 char. name of fg 87 ?> = fds87
Enter new 6 char. name of fg 88 ?> = fds88
Enter new 6 char. name of fg 89 ?> = fds89
Enter new 6 char. name of fg 90 ?> = fds90
Enter new 6 char. name of fg 91 ?> = fds91
Enter new 6 char. name of fg 92 ?> = fds92
Enter new 6 char. name of fg 93 ?> = fds93
Enter new 6 char. name of fg 94 ?> = fds94

Enter new 6 char. name of fg 95 ?> = fds95
Enter new 6 char. name of fg 96 ?> = fds96
Enter new 6 char. name of fg 97 ?> = fds97
Enter new 6 char. name of fg 98 ?> = fds98
Enter new 6 char. name of fg 99 ?> = fds99
Enter new 6 char. name of fg 100 ?> = fds100
Enter new 6 char. name of fg 101 ?> = fds101
Enter new 6 char. name of fg 102 ?> = fds102
Enter new 6 char. name of fg 103 ?> = fds103
Enter new 6 char. name of fg 104 ?> = fds104
Enter new 6 char. name of fg 105 ?> = fds105
Enter new 6 char. name of fg 106 ?> = fds106
Enter new 6 char. name of fg 107 ?> = fds107
Enter new 6 char. name of fg 108 ?> = fds108
Enter new 6 char. name of fg 109 ?> = fds109
Enter new 6 char. name of fg 110 ?> = fds110
Enter new 6 char. name of fg 111 ?> = fds111
Enter new 6 char. name of fg 112 ?> = fds112
Enter new 6 char. name of fg 113 ?> = fds113
Enter new 6 char. name of fg 114 ?> = fds114
Enter new 6 char. name of fg 115 ?> = fds115
Enter new 6 char. name of fg 116 ?> = fds116
Enter new 6 char. name of fg 117 ?> = fds117
Enter new 6 char. name of fg 118 ?> = fds118
Enter new 6 char. name of fg 119 ?> = fds119
Enter new 6 char. name of fg 120 ?> = fds120
Enter new 6 char. name of fg 121 ?> = fds121
Enter new 6 char. name of fg 122 ?> = fds122
Enter new 6 char. name of fg 123 ?> = fds123
Enter new 6 char. name of fg 124 ?> = fds124
Enter new 6 char. name of fg 125 ?> = fds125
Enter new 6 char. name of fg 126 ?> = fds126
Enter new 6 char. name of fg 127 ?> = fds127
Enter new 6 char. name of fg 128 ?> = fds128
Enter new 6 char. name of fg 129 ?> = fds129
Enter new 6 char. name of fg 130 ?> = fds130
Enter new 6 char. name of fg 131 ?> = fds131
Enter new 6 char. name of fg 132 ?> = fds132
Enter new 6 char. name of fg 133 ?> = fds133
Enter new 6 char. name of fg 134 ?> = fds134
Enter new 6 char. name of fg 135 ?> = fds135
Enter new 6 char. name of fg 136 ?> = fds136
Enter new 6 char. name of fg 137 ?> = fds137
Enter new 6 char. name of fg 138 ?> = fds138
Enter new 6 char. name of fg 139 ?> = fds139
Enter new 6 char. name of fg 140 ?> = fds140
Enter new 6 char. name of fg 141 ?> = fds141
Enter new 6 char. name of fg 142 ?> = fds142
Enter new 6 char. name of fg 143 ?> = fds143
Enter new 6 char. name of fg 144 ?> = fds144

Enter new 6 char. name of fg 145 ?> = fds145
 Enter new 6 char. name of fg 146 ?> = fds146
 Enter new 6 char. name of fg 147 ?> = fds147
 Enter new 6 char. name of fg 148 ?> = fds148
 Enter new 6 char. name of fg 149 ?> = fds149
 Enter new 6 char. name of fg 150 ?> = fds150
 Enter new 6 char. name of fg 151 ?> = fds151
 Enter new 6 char. name of fg 152 ?> = fds152
 Enter new 6 char. name of fg 153 ?> = fds153
 Enter new 6 char. name of fg 154 ?> = fds154
 Enter new 6 char. name of fg 155 ?> = fds155
 Enter new 6 char. name of fg 156 ?> = fds156
 Enter new 6 char. name of fg 157 ?> = fds157
 Enter new 6 char. name of fg 158 ?> = fds158
 Enter new 6 char. name of fg 159 ?> = fds159
 Enter new 6 char. name of fg 160 ?> = fds160
 Enter new 6 char. name of fg 161 ?> = fds161
 Enter new 6 char. name of fg 162 ?> = fds162
 Enter new 6 char. name of fg 163 ?> = fds163
 Enter new 6 char. name of fg 164 ?> = fds164
 Enter new 6 char. name of fg 165 ?> = fds165
 Enter new 6 char. name of fg 166 ?> = fds166
 Enter new 6 char. name of fg 167 ?> = fds167
 Enter new 6 char. name of fg 168 ?> = fds168
 Enter new 6 char. name of fg 169 ?> = fds169
 Enter new 6 char. name of fg 170 ?> = fds170
 Enter new 6 char. name of fg 171 ?> = fds171
 Enter new 6 char. name of fg 172 ?> = fds172
 Enter new 6 char. name of fg 173 ?> = fds173
 Enter new 6 char. name of fg 174 ?> = fds174
 Enter new 6 char. name of fg 175 ?> = fds175
 Enter new 6 char. name of fg 176 ?> = fds176
 Enter new 6 char. name of fg 177 ?> = fds177
 Enter new 6 char. name of fg 178 ?> = fds178
 Enter new 6 char. name of fg 179 ?> = fds179
 Enter new 6 char. name of fg 180 ?> = fds180
 Enter new 6 char. name of fg 181 ?> = fds181
 Enter new 6 char. name of fg 182 ?> = fds182
 Enter new 6 char. name of fg 183 ?> = fds183
 Enter new 6 char. name of fg 184 ?> = fds184
 Enter new 6 char. name of fg 185 ?> = fds185
 Enter new 6 char. name of fg 186 ?> = fds186
 Enter new 6 char. name of fg 187 ?> = fds187
 Enter new 6 char. name of fg 188 ?> = fds188
 Enter new 6 char. name of fg 189 ?> = fds189
 Enter new 6 char. name of fg 190 ?> = fds190
 Enter new 6 char. name of fg 191 ?> = fds191
 Enter new 6 char. name of fg 192 ?> = fx0
 Enter new 6 char. name of fg 193 ?> = fx1
 Enter new 6 char. name of fg 194 ?> = fx2

Enter new 6 char. name of fg 195 ?> = fx3
Enter new 6 char. name of fg 196 ?> = fx4
Enter new 6 char. name of fg 197 ?> = fx5
Enter new 6 char. name of fg 198 ?> = fx6
Enter new 6 char. name of fg 199 ?> = fx7
Enter new 6 char. name of fg 200 ?> = fx8
Enter new 6 char. name of fg 201 ?> = fx9
Enter new 6 char. name of fg 202 ?> = fx10
Enter new 6 char. name of fg 203 ?> = fx11
Enter new 6 char. name of fg 204 ?> = fx12
Enter new 6 char. name of fg 205 ?> = fx13
Enter new 6 char. name of fg 206 ?> = fx14
Enter new 6 char. name of fg 207 ?> = fx15
Enter new 6 char. name of fg 208 ?> = fx16
Enter new 6 char. name of fg 209 ?> = fx17
Enter new 6 char. name of fg 210 ?> = fx18
Enter new 6 char. name of fg 211 ?> = fx19
Enter new 6 char. name of fg 212 ?> = fx20
Enter new 6 char. name of fg 213 ?> = fx21
Enter new 6 char. name of fg 214 ?> = fx22
Enter new 6 char. name of fg 215 ?> = fx23
Enter new 6 char. name of fg 216 ?> = fx24
Enter new 6 char. name of fg 217 ?> = fx25
Enter new 6 char. name of fg 218 ?> = fx26
Enter new 6 char. name of fg 219 ?> = fx27
Enter new 6 char. name of fg 220 ?> = fx28
Enter new 6 char. name of fg 221 ?> = fx29
Enter new 6 char. name of fg 222 ?> = fx30
Enter new 6 char. name of fg 223 ?> = fx31
Enter new 6 char. name of fg 224 ?> = fx32
Enter new 6 char. name of fg 225 ?> = fx33
Enter new 6 char. name of fg 226 ?> = fx34
Enter new 6 char. name of fg 227 ?> = fx35
Enter new 6 char. name of fg 228 ?> = fx36
Enter new 6 char. name of fg 229 ?> = fx37
Enter new 6 char. name of fg 230 ?> = fx38
Enter new 6 char. name of fg 231 ?> = fx39
Enter new 6 char. name of fg 232 ?> = fx40
Enter new 6 char. name of fg 233 ?> = fx41
Enter new 6 char. name of fg 234 ?> = fx42
Enter new 6 char. name of fg 235 ?> = fx43
Enter new 6 char. name of fg 236 ?> = fx44
Enter new 6 char. name of fg 237 ?> = fx45
Enter new 6 char. name of fg 238 ?> = fx46
Enter new 6 char. name of fg 239 ?> = fx47
Enter new 6 char. name of fg 240 ?> = fx48
Enter new 6 char. name of fg 241 ?> = fx49
Enter new 6 char. name of fg 242 ?> = fx50
Enter new 6 char. name of fg 243 ?> = fx51
Enter new 6 char. name of fg 244 ?> = fx52

Enter new 6 char. name of fg 245 ?> = fx53
 Enter new 6 char. name of fg 246 ?> = fx54
 Enter new 6 char. name of fg 247 ?> = fx55
 Enter new 6 char. name of fg 248 ?> = fx56
 Enter new 6 char. name of fg 249 ?> = fx57
 Enter new 6 char. name of fg 250 ?> = fx58
 Enter new 6 char. name of fg 251 ?> = fx59
 Enter new 6 char. name of fg 252 ?> = fx60
 Enter new 6 char. name of fg 253 ?> = fx61
 Enter new 6 char. name of fg 254 ?> = fx62
 Enter new 6 char. name of fg 255 ?> = fx63
 Enter new 6 char. name of fg 256 ?> = fx64
 Enter new 6 char. name of fg 257 ?> = fx65
 Enter new 6 char. name of fg 258 ?> = fx66
 Enter new 6 char. name of fg 259 ?> = fx67
 Enter new 6 char. name of fg 260 ?> = fx68
 Enter new 6 char. name of fg 261 ?> = fx69
 Enter new 6 char. name of fg 262 ?> = fx70
 Enter new 6 char. name of fg 263 ?> = fx71
 Enter new 6 char. name of fg 264 ?> = fx72
 Enter new 6 char. name of fg 265 ?> = fx73
 Enter new 6 char. name of fg 266 ?> = fx74
 Enter new 6 char. name of fg 267 ?> = fx75
 Enter new 6 char. name of fg 268 ?> = fx76
 Enter new 6 char. name of fg 269 ?> = fx77
 Enter new 6 char. name of fg 270 ?> = fx78
 Enter new 6 char. name of fg 271 ?> = fx79
 Enter new 6 char. name of fg 272 ?> = fx80
 Enter new 6 char. name of fg 273 ?> = fx81
 Enter new 6 char. name of fg 274 ?> = fx82
 Enter new 6 char. name of fg 275 ?> = fx83
 Enter new 6 char. name of fg 276 ?> = fx84
 Enter new 6 char. name of fg 277 ?> = fx85
 Enter new 6 char. name of fg 278 ?> = fx86
 Enter new 6 char. name of fg 279 ?> = fx87
 Enter new 6 char. name of fg 280 ?> = fx88
 Enter new 6 char. name of fg 281 ?> = fx89
 Enter new 6 char. name of fg 282 ?> = fx90
 Enter new 6 char. name of fg 283 ?> = fx91
 Enter new 6 char. name of fg 284 ?> = fx92
 Enter new 6 char. name of fg 285 ?> = fx93
 Enter new 6 char. name of fg 286 ?> = fx94
 Enter new 6 char. name of fg 287 ?> = fx95
 Enter new 6 char. name of fg 288 ?> = fx96
 Enter new 6 char. name of fg 289 ?> = fx97
 Enter new 6 char. name of fg 290 ?> = fx98
 Enter new 6 char. name of fg 291 ?> = fx99
 Enter new 6 char. name of fg 292 ?> = fx100
 Enter new 6 char. name of fg 293 ?> = fx101
 Enter new 6 char. name of fg 294 ?> = fx102

Enter new 6 char. name of fg 295 ?> = fx103
 Enter new 6 char. name of fg 296 ?> = fx104
 Enter new 6 char. name of fg 297 ?> = fx105
 Enter new 6 char. name of fg 298 ?> = fx106
 Enter new 6 char. name of fg 299 ?> = fx107
 Enter new 6 char. name of fg 300 ?> = fx108
 Enter new 6 char. name of fg 301 ?> = fx109
 Enter new 6 char. name of fg 302 ?> = fx110
 Enter new 6 char. name of fg 303 ?> = fx111
 Enter new 6 char. name of fg 304 ?> = fx112
 Enter new 6 char. name of fg 305 ?> = fx113
 Enter new 6 char. name of fg 306 ?> = fx114
 Enter new 6 char. name of fg 307 ?> = fx115
 Enter new 6 char. name of fg 308 ?> = fx116
 Enter new 6 char. name of fg 309 ?> = fx117
 Enter new 6 char. name of fg 310 ?> = fx118
 Enter new 6 char. name of fg 311 ?> = fx119
 Enter new 6 char. name of fg 312 ?> = fx120
 Enter new 6 char. name of fg 313 ?> = fx121
 Enter new 6 char. name of fg 314 ?> = fx122
 Enter new 6 char. name of fg 315 ?> = fx123
 Enter new 6 char. name of fg 316 ?> = fx124
 Enter new 6 char. name of fg 317 ?> = fx125
 Enter new 6 char. name of fg 318 ?> = fx126
 Enter new 6 char. name of fg 319 ?> = fx127
 Enter new 6 char. name of fg 320 ?> = fx128
 Enter new 6 char. name of fg 321 ?> = fx129
 Enter new 6 char. name of fg 322 ?> = fx130
 Enter new 6 char. name of fg 323 ?> = fx131
 Enter new 6 char. name of fg 324 ?> = fx132
 Enter new 6 char. name of fg 325 ?> = fx133
 Enter new 6 char. name of fg 326 ?> = fx134
 Enter new 6 char. name of fg 327 ?> = fx135
 Enter new 6 char. name of fg 328 ?> = fx136
 Enter new 6 char. name of fg 329 ?> = fx137
 Enter new 6 char. name of fg 330 ?> = fx138
 Enter new 6 char. name of fg 331 ?> = fx139
 Enter new 6 char. name of fg 332 ?> = fx140
 Enter new 6 char. name of fg 333 ?> = fx141
 Enter new 6 char. name of fg 334 ?> = fx142
 Enter new 6 char. name of fg 335 ?> = fx143
 Enter new 6 char. name of fg 336 ?> = fx144
 Enter new 6 char. name of fg 337 ?> = fx145
 Enter new 6 char. name of fg 338 ?> = fx146
 Enter new 6 char. name of fg 339 ?> = fx147
 Enter new 6 char. name of fg 340 ?> = fx148
 Enter new 6 char. name of fg 341 ?> = fx149
 Enter new 6 char. name of fg 342 ?> = fx150
 Enter new 6 char. name of fg 343 ?> = fx151
 Enter new 6 char. name of fg 344 ?> = fx152

WMC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fg 345 ?> = fx153
 Enter new 6 char. name of fg 346 ?> = fx154
 Enter new 6 char. name of fg 347 ?> = fx155
 Enter new 6 char. name of fg 348 ?> = fx156
 Enter new 6 char. name of fg 349 ?> = fx157
 Enter new 6 char. name of fg 350 ?> = fx158
 Enter new 6 char. name of fg 351 ?> = fx159
 Enter new 6 char. name of fg 352 ?> = fx160
 Enter new 6 char. name of fg 353 ?> = fx161
 Enter new 6 char. name of fg 354 ?> = fx162
 Enter new 6 char. name of fg 355 ?> = fx163
 Enter new 6 char. name of fg 356 ?> = fx164
 Enter new 6 char. name of fg 357 ?> = fx165
 Enter new 6 char. name of fg 358 ?> = fx166
 Enter new 6 char. name of fg 359 ?> = fx167
 Enter new 6 char. name of fg 360 ?> = fx168
 Enter new 6 char. name of fg 361 ?> = fx169
 Enter new 6 char. name of fg 362 ?> = fx170
 Enter new 6 char. name of fg 363 ?> = fx171
 Enter new 6 char. name of fg 364 ?> = fx172
 Enter new 6 char. name of fg 365 ?> = fx173
 Enter new 6 char. name of fg 366 ?> = fx174
 Enter new 6 char. name of fg 367 ?> = fx175
 Enter new 6 char. name of fg 368 ?> = fx176
 Enter new 6 char. name of fg 369 ?> = fx177
 Enter new 6 char. name of fg 370 ?> = fx178
 Enter new 6 char. name of fg 371 ?> = fx179
 Enter new 6 char. name of fg 372 ?> = fx180
 Enter new 6 char. name of fg 373 ?> = fx181
 Enter new 6 char. name of fg 374 ?> = fx182
 Enter new 6 char. name of fg 375 ?> = fx183
 Enter new 6 char. name of fg 376 ?> = fx184
 Enter new 6 char. name of fg 377 ?> = fx185
 Enter new 6 char. name of fg 378 ?> = fx186
 Enter new 6 char. name of fg 379 ?> = fx187
 Enter new 6 char. name of fg 380 ?> = fx188
 Enter new 6 char. name of fg 381 ?> = fx189
 Enter new 6 char. name of fg 382 ?> = fx190
 Enter new 6 char. name of fg 383 ?> = fx191
 Enter new 6 char. name of fg 384 ?> = fa10
 Enter new 6 char. name of fg 385 ?> = fa11
 Enter new 6 char. name of fg 386 ?> = fa12
 Enter new 6 char. name of fg 387 ?> = fa13
 Enter new 6 char. name of fg 388 ?> = fa14
 Enter new 6 char. name of fg 389 ?> = fa15
 Enter new 6 char. name of fg 390 ?> = fa16
 Enter new 6 char. name of fg 391 ?> = fa17
 Enter new 6 char. name of fg 392 ?> = fa18
 Enter new 6 char. name of fg 393 ?> = fa19
 Enter new 6 char. name of fg 394 ?> = fa110

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fg 395 ?> = fal11
Enter new 6 char. name of fg 396 ?> = fal12
Enter new 6 char. name of fg 397 ?> = fal13
Enter new 6 char. name of fg 398 ?> = fal14
Enter new 6 char. name of fg 399 ?> = fal15
Enter new 6 char. name of fg 400 ?> = fal16
Enter new 6 char. name of fg 401 ?> = fal17
Enter new 6 char. name of fg 402 ?> = fal18
Enter new 6 char. name of fg 403 ?> = fal19
Enter new 6 char. name of fg 404 ?> = fal20
Enter new 6 char. name of fg 405 ?> = fal21
Enter new 6 char. name of fg 406 ?> = fal22
Enter new 6 char. name of fg 407 ?> = fal23
Enter new 6 char. name of fg 408 ?> = fal24
Enter new 6 char. name of fg 409 ?> = fal25
Enter new 6 char. name of fg 410 ?> = fal26
Enter new 6 char. name of fg 411 ?> = fal27
Enter new 6 char. name of fg 412 ?> = fal28
Enter new 6 char. name of fg 413 ?> = fal29
Enter new 6 char. name of fg 414 ?> = fal30
Enter new 6 char. name of fg 415 ?> = fal31
Enter new 6 char. name of fg 416 ?> = fal32
Enter new 6 char. name of fg 417 ?> = fal33
Enter new 6 char. name of fg 418 ?> = fal34
Enter new 6 char. name of fg 419 ?> = fal35
Enter new 6 char. name of fg 420 ?> = fal36
Enter new 6 char. name of fg 421 ?> = fal37
Enter new 6 char. name of fg 422 ?> = fal38
Enter new 6 char. name of fg 423 ?> = fal39
Enter new 6 char. name of fg 424 ?> = fal40
Enter new 6 char. name of fg 425 ?> = fal41
Enter new 6 char. name of fg 426 ?> = fal42
Enter new 6 char. name of fg 427 ?> = fal43
Enter new 6 char. name of fg 428 ?> = fal44
Enter new 6 char. name of fg 429 ?> = fal45
Enter new 6 char. name of fg 430 ?> = fal46
Enter new 6 char. name of fg 431 ?> = fal47
Enter new 6 char. name of fg 432 ?> = fal48
Enter new 6 char. name of fg 433 ?> = fal49
Enter new 6 char. name of fg 434 ?> = fal50
Enter new 6 char. name of fg 435 ?> = fal51
Enter new 6 char. name of fg 436 ?> = fal52
Enter new 6 char. name of fg 437 ?> = fal53
Enter new 6 char. name of fg 438 ?> = fal54
Enter new 6 char. name of fg 439 ?> = fal55
Enter new 6 char. name of fg 440 ?> = fal56
Enter new 6 char. name of fg 441 ?> = fal57
Enter new 6 char. name of fg 442 ?> = fal58
Enter new 6 char. name of fg 443 ?> = fal59
Enter new 6 char. name of fg 444 ?> = fal60

Enter new 6 char. name of fg 445 ?> = fal161
 Enter new 6 char. name of fg 446 ?> = fal162
 Enter new 6 char. name of fg 447 ?> = fal163
 Enter new 6 char. name of fg 448 ?> = fal164
 Enter new 6 char. name of fg 449 ?> = fal165
 Enter new 6 char. name of fg 450 ?> = fal166
 Enter new 6 char. name of fg 451 ?> = fal167
 Enter new 6 char. name of fg 452 ?> = fal168
 Enter new 6 char. name of fg 453 ?> = fal169
 Enter new 6 char. name of fg 454 ?> = fal170
 Enter new 6 char. name of fg 455 ?> = fal171
 Enter new 6 char. name of fg 456 ?> = fal172
 Enter new 6 char. name of fg 457 ?> = fal173
 Enter new 6 char. name of fg 458 ?> = fal174
 Enter new 6 char. name of fg 459 ?> = fal175
 Enter new 6 char. name of fg 460 ?> = fal176
 Enter new 6 char. name of fg 461 ?> = fal177
 Enter new 6 char. name of fg 462 ?> = fal178
 Enter new 6 char. name of fg 463 ?> = fal179
 Enter new 6 char. name of fg 464 ?> = fal180
 Enter new 6 char. name of fg 465 ?> = fal181
 Enter new 6 char. name of fg 466 ?> = fal182
 Enter new 6 char. name of fg 467 ?> = fal183
 Enter new 6 char. name of fg 468 ?> = fal184
 Enter new 6 char. name of fg 469 ?> = fal185
 Enter new 6 char. name of fg 470 ?> = fal186
 Enter new 6 char. name of fg 471 ?> = fal187
 Enter new 6 char. name of fg 472 ?> = fal188
 Enter new 6 char. name of fg 473 ?> = fal189
 Enter new 6 char. name of fg 474 ?> = fal190
 Enter new 6 char. name of fg 475 ?> = fal191
 Enter new 6 char. name of fg 476 ?> = fal192
 Enter new 6 char. name of fg 477 ?> = fal193
 Enter new 6 char. name of fg 478 ?> = fal194
 Enter new 6 char. name of fg 479 ?> = fal195
 Enter new 6 char. name of fg 480 ?> = fal196
 Enter new 6 char. name of fg 481 ?> = fal197
 Enter new 6 char. name of fg 482 ?> = fal198
 Enter new 6 char. name of fg 483 ?> = fal199
 Enter new 6 char. name of fg 484 ?> = fal100
 Enter new 6 char. name of fg 485 ?> = fal101
 Enter new 6 char. name of fg 486 ?> = fal102
 Enter new 6 char. name of fg 487 ?> = fal103
 Enter new 6 char. name of fg 488 ?> = fal104
 Enter new 6 char. name of fg 489 ?> = fal105
 Enter new 6 char. name of fg 490 ?> = fal106
 Enter new 6 char. name of fg 491 ?> = fal107
 Enter new 6 char. name of fg 492 ?> = fal108
 Enter new 6 char. name of fg 493 ?> = fal109
 Enter new 6 char. name of fg 494 ?> = fal110

WMC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fg 495 ?> = fall111
Enter new 6 char. name of fg 496 ?> = fall112
Enter new 6 char. name of fg 497 ?> = fall113
Enter new 6 char. name of fg 498 ?> = fall114
Enter new 6 char. name of fg 499 ?> = fall115
Enter new 6 char. name of fg 500 ?> = fall116
Enter new 6 char. name of fg 501 ?> = fall117
Enter new 6 char. name of fg 502 ?> = fall118
Enter new 6 char. name of fg 503 ?> = fall119
Enter new 6 char. name of fg 504 ?> = fall120
Enter new 6 char. name of fg 505 ?> = fall121
Enter new 6 char. name of fg 506 ?> = fall122
Enter new 6 char. name of fg 507 ?> = fall123
Enter new 6 char. name of fg 508 ?> = fall124
Enter new 6 char. name of fg 509 ?> = fall125
Enter new 6 char. name of fg 510 ?> = fall126
Enter new 6 char. name of fg 511 ?> = fall127
Enter new 6 char. name of fg 512 ?> = fall128
Enter new 6 char. name of fg 513 ?> = fall129
Enter new 6 char. name of fg 514 ?> = fall130
Enter new 6 char. name of fg 515 ?> = fall131
Enter new 6 char. name of fg 516 ?> = fall132
Enter new 6 char. name of fg 517 ?> = fall133
Enter new 6 char. name of fg 518 ?> = fall134
Enter new 6 char. name of fg 519 ?> = fall135
Enter new 6 char. name of fg 520 ?> = fall136
Enter new 6 char. name of fg 521 ?> = fall137
Enter new 6 char. name of fg 522 ?> = fall138
Enter new 6 char. name of fg 523 ?> = fall139
Enter new 6 char. name of fg 524 ?> = fall140
Enter new 6 char. name of fg 525 ?> = fall141
Enter new 6 char. name of fg 526 ?> = fall142
Enter new 6 char. name of fg 527 ?> = fall143
Enter new 6 char. name of fg 528 ?> = fall144
Enter new 6 char. name of fg 529 ?> = fall145
Enter new 6 char. name of fg 530 ?> = fall146
Enter new 6 char. name of fg 531 ?> = fall147
Enter new 6 char. name of fg 532 ?> = fall148
Enter new 6 char. name of fg 533 ?> = fall149
Enter new 6 char. name of fg 534 ?> = fall150
Enter new 6 char. name of fg 535 ?> = fall151
Enter new 6 char. name of fg 536 ?> = fall152
Enter new 6 char. name of fg 537 ?> = fall153
Enter new 6 char. name of fg 538 ?> = fall154
Enter new 6 char. name of fg 539 ?> = fall155
Enter new 6 char. name of fg 540 ?> = fall156
Enter new 6 char. name of fg 541 ?> = fall157
Enter new 6 char. name of fg 542 ?> = fall158
Enter new 6 char. name of fg 543 ?> = fall159
Enter new 6 char. name of fg 544 ?> = fall160

Enter new 6 char. name of fg 545 ?> = fal161
 Enter new 6 char. name of fg 546 ?> = fal162
 Enter new 6 char. name of fg 547 ?> = fal163
 Enter new 6 char. name of fg 548 ?> = fal164
 Enter new 6 char. name of fg 549 ?> = fal165
 Enter new 6 char. name of fg 550 ?> = fal166
 Enter new 6 char. name of fg 551 ?> = fal167
 Enter new 6 char. name of fg 552 ?> = fal168
 Enter new 6 char. name of fg 553 ?> = fal169
 Enter new 6 char. name of fg 554 ?> = fal170
 Enter new 6 char. name of fg 555 ?> = fal171
 Enter new 6 char. name of fg 556 ?> = fal172
 Enter new 6 char. name of fg 557 ?> = fal173
 Enter new 6 char. name of fg 558 ?> = fal174
 Enter new 6 char. name of fg 559 ?> = fal175
 Enter new 6 char. name of fg 560 ?> = fal176
 Enter new 6 char. name of fg 561 ?> = fal177
 Enter new 6 char. name of fg 562 ?> = fal178
 Enter new 6 char. name of fg 563 ?> = fal179
 Enter new 6 char. name of fg 564 ?> = fal180
 Enter new 6 char. name of fg 565 ?> = fal181
 Enter new 6 char. name of fg 566 ?> = fal182
 Enter new 6 char. name of fg 567 ?> = fal183
 Enter new 6 char. name of fg 568 ?> = fal184
 Enter new 6 char. name of fg 569 ?> = fal185
 Enter new 6 char. name of fg 570 ?> = fal186
 Enter new 6 char. name of fg 571 ?> = fal187
 Enter new 6 char. name of fg 572 ?> = fal188
 Enter new 6 char. name of fg 573 ?> = fal189
 Enter new 6 char. name of fg 574 ?> = fal190
 Enter new 6 char. name of fg 575 ?> = fal191
 Enter new 6 char. name of fg 576 ?> = frv0
 Enter new 6 char. name of fg 577 ?> = frv1
 Enter new 6 char. name of fg 578 ?> = frv2
 Enter new 6 char. name of fg 579 ?> = frv3
 Enter new 6 char. name of fg 580 ?> = frv4
 Enter new 6 char. name of fg 581 ?> = frv5
 Enter new 6 char. name of fg 582 ?> = frv6
 Enter new 6 char. name of fg 583 ?> = frv7
 Enter new 6 char. name of fg 584 ?> = frv8
 Enter new 6 char. name of fg 585 ?> = frv9
 Enter new 6 char. name of fg 586 ?> = frv10
 Enter new 6 char. name of fg 587 ?> = frv11
 Enter new 6 char. name of fg 588 ?> = frv12
 Enter new 6 char. name of fg 589 ?> = frv13
 Enter new 6 char. name of fg 590 ?> = frv14
 Enter new 6 char. name of fg 591 ?> = frv15
 Enter new 6 char. name of fg 592 ?> = frv16
 Enter new 6 char. name of fg 593 ?> = frv17
 Enter new 6 char. name of fg 594 ?> = frv18

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fg 595 ?> = frv19
 Enter new 6 char. name of fg 596 ?> = frv20
 Enter new 6 char. name of fg 597 ?> = frv21
 Enter new 6 char. name of fg 598 ?> = frv22
 Enter new 6 char. name of fg 599 ?> = frv23
 Enter new 6 char. name of fg 600 ?> = frv24
 Enter new 6 char. name of fg 601 ?> = frv25
 Enter new 6 char. name of fg 602 ?> = frv26
 Enter new 6 char. name of fg 603 ?> = frv27
 Enter new 6 char. name of fg 604 ?> = frv28
 Enter new 6 char. name of fg 605 ?> = frv29
 Enter new 6 char. name of fg 606 ?> = frv30
 Enter new 6 char. name of fg 607 ?> = frv31
 Enter new 6 char. name of fg 608 ?> = frv32
 Enter new 6 char. name of fg 609 ?> = frv33
 Enter new 6 char. name of fg 610 ?> = frv34
 Enter new 6 char. name of fg 611 ?> = frv35
 Enter new 6 char. name of fg 612 ?> = frv36
 Enter new 6 char. name of fg 613 ?> = frv37
 Enter new 6 char. name of fg 614 ?> = frv38
 Enter new 6 char. name of fg 615 ?> = frv39
 Enter new 6 char. name of fg 616 ?> = frv40
 Enter new 6 char. name of fg 617 ?> = frv41
 Enter new 6 char. name of fg 618 ?> = frv42
 Enter new 6 char. name of fg 619 ?> = frv43
 Enter new 6 char. name of fg 620 ?> = frv44
 Enter new 6 char. name of fg 621 ?> = frv45
 Enter new 6 char. name of fg 622 ?> = frv46
 Enter new 6 char. name of fg 623 ?> = frv47
 Enter new 6 char. name of fg 624 ?> = frv48
 Enter new 6 char. name of fg 625 ?> = frv49
 Enter new 6 char. name of fg 626 ?> = frv50
 Enter new 6 char. name of fg 627 ?> = frv51
 Enter new 6 char. name of fg 628 ?> = frv52
 Enter new 6 char. name of fg 629 ?> = frv53
 Enter new 6 char. name of fg 630 ?> = frv54
 Enter new 6 char. name of fg 631 ?> = frv55
 Enter new 6 char. name of fg 632 ?> = frv56
 Enter new 6 char. name of fg 633 ?> = frv57
 Enter new 6 char. name of fg 634 ?> = frv58
 Enter new 6 char. name of fg 635 ?> = frv59
 Enter new 6 char. name of fg 636 ?> = frv60
 Enter new 6 char. name of fg 637 ?> = frv61
 Enter new 6 char. name of fg 638 ?> = frv62
 Enter new 6 char. name of fg 639 ?> = frv63
 Enter new 6 char. name of fg 640 ?> = frv64
 Enter new 6 char. name of fg 641 ?> = frv65
 Enter new 6 char. name of fg 642 ?> = frv66
 Enter new 6 char. name of fg 643 ?> = frv67
 Enter new 6 char. name of fg 644 ?> = frv68

WMC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fg 645 ?> = frv69
Enter new 6 char. name of fg 646 ?> = frv70
Enter new 6 char. name of fg 647 ?> = frv71
Enter new 6 char. name of fg 648 ?> = frv72
Enter new 6 char. name of fg 649 ?> = frv73
Enter new 6 char. name of fg 650 ?> = frv74
Enter new 6 char. name of fg 651 ?> = frv75
Enter new 6 char. name of fg 652 ?> = frv76
Enter new 6 char. name of fg 653 ?> = frv77
Enter new 6 char. name of fg 654 ?> = frv78
Enter new 6 char. name of fg 655 ?> = frv79
Enter new 6 char. name of fg 656 ?> = frv80
Enter new 6 char. name of fg 657 ?> = frv81
Enter new 6 char. name of fg 658 ?> = frv82
Enter new 6 char. name of fg 659 ?> = frv83
Enter new 6 char. name of fg 660 ?> = frv84
Enter new 6 char. name of fg 661 ?> = frv85
Enter new 6 char. name of fg 662 ?> = frv86
Enter new 6 char. name of fg 663 ?> = frv87
Enter new 6 char. name of fg 664 ?> = frv88
Enter new 6 char. name of fg 665 ?> = frv89
Enter new 6 char. name of fg 666 ?> = frv90
Enter new 6 char. name of fg 667 ?> = frv91
Enter new 6 char. name of fg 668 ?> = frv92
Enter new 6 char. name of fg 669 ?> = frv93
Enter new 6 char. name of fg 670 ?> = frv94
Enter new 6 char. name of fg 671 ?> = frv95
Enter new 6 char. name of fg 672 ?> = frv96
Enter new 6 char. name of fg 673 ?> = frv97
Enter new 6 char. name of fg 674 ?> = frv98
Enter new 6 char. name of fg 675 ?> = frv99
Enter new 6 char. name of fg 676 ?> = frv100
Enter new 6 char. name of fg 677 ?> = frv101
Enter new 6 char. name of fg 678 ?> = frv102
Enter new 6 char. name of fg 679 ?> = frv103
Enter new 6 char. name of fg 680 ?> = frv104
Enter new 6 char. name of fg 681 ?> = frv105
Enter new 6 char. name of fg 682 ?> = frv106
Enter new 6 char. name of fg 683 ?> = frv107
Enter new 6 char. name of fg 684 ?> = frv108
Enter new 6 char. name of fg 685 ?> = frv109
Enter new 6 char. name of fg 686 ?> = frv110
Enter new 6 char. name of fg 687 ?> = frv111
Enter new 6 char. name of fg 688 ?> = frv112
Enter new 6 char. name of fg 689 ?> = frv113
Enter new 6 char. name of fg 690 ?> = frv114
Enter new 6 char. name of fg 691 ?> = frv115
Enter new 6 char. name of fg 692 ?> = frv116
Enter new 6 char. name of fg 693 ?> = frv117
Enter new 6 char. name of fg 694 ?> = frv118

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fg 695 ?> = frv119
 Enter new 6 char. name of fg 696 ?> = frv120
 Enter new 6 char. name of fg 697 ?> = frv121
 Enter new 6 char. name of fg 698 ?> = frv122
 Enter new 6 char. name of fg 699 ?> = frv123
 Enter new 6 char. name of fg 700 ?> = frv124
 Enter new 6 char. name of fg 701 ?> = frv125
 Enter new 6 char. name of fg 702 ?> = frv126
 Enter new 6 char. name of fg 703 ?> = frv127
 Enter new 6 char. name of fg 704 ?> = frv128
 Enter new 6 char. name of fg 705 ?> = frv129
 Enter new 6 char. name of fg 706 ?> = frv130
 Enter new 6 char. name of fg 707 ?> = frv131
 Enter new 6 char. name of fg 708 ?> = frv132
 Enter new 6 char. name of fg 709 ?> = frv133
 Enter new 6 char. name of fg 710 ?> = frv134
 Enter new 6 char. name of fg 711 ?> = frv135
 Enter new 6 char. name of fg 712 ?> = frv136
 Enter new 6 char. name of fg 713 ?> = frv137
 Enter new 6 char. name of fg 714 ?> = frv138
 Enter new 6 char. name of fg 715 ?> = frv139
 Enter new 6 char. name of fg 716 ?> = frv140
 Enter new 6 char. name of fg 717 ?> = frv141
 Enter new 6 char. name of fg 718 ?> = frv142
 Enter new 6 char. name of fg 719 ?> = frv143
 Enter new 6 char. name of fg 720 ?> = frv144
 Enter new 6 char. name of fg 721 ?> = frv145
 Enter new 6 char. name of fg 722 ?> = frv146
 Enter new 6 char. name of fg 723 ?> = frv147
 Enter new 6 char. name of fg 724 ?> = frv148
 Enter new 6 char. name of fg 725 ?> = frv149
 Enter new 6 char. name of fg 726 ?> = frv150
 Enter new 6 char. name of fg 727 ?> = frv151
 Enter new 6 char. name of fg 728 ?> = frv152
 Enter new 6 char. name of fg 729 ?> = frv153
 Enter new 6 char. name of fg 730 ?> = frv154
 Enter new 6 char. name of fg 731 ?> = frv155
 Enter new 6 char. name of fg 732 ?> = frv156
 Enter new 6 char. name of fg 733 ?> = frv157
 Enter new 6 char. name of fg 734 ?> = frv158
 Enter new 6 char. name of fg 735 ?> = frv159
 Enter new 6 char. name of fg 736 ?> = frv160
 Enter new 6 char. name of fg 737 ?> = frv161
 Enter new 6 char. name of fg 738 ?> = frv162
 Enter new 6 char. name of fg 739 ?> = frv163
 Enter new 6 char. name of fg 740 ?> = frv164
 Enter new 6 char. name of fg 741 ?> = frv165
 Enter new 6 char. name of fg 742 ?> = frv166
 Enter new 6 char. name of fg 743 ?> = frv167
 Enter new 6 char. name of fg 744 ?> = frv168

```

Enter new 6 char. name of fg 745 ?> = frv169
Enter new 6 char. name of fg 746 ?> = frv170
Enter new 6 char. name of fg 747 ?> = frv171
Enter new 6 char. name of fg 748 ?> = frv172
Enter new 6 char. name of fg 749 ?> = frv173
Enter new 6 char. name of fg 750 ?> = frv174
Enter new 6 char. name of fg 751 ?> = frv175
Enter new 6 char. name of fg 752 ?> = frv176
Enter new 6 char. name of fg 753 ?> = frv177
Enter new 6 char. name of fg 754 ?> = frv178
Enter new 6 char. name of fg 755 ?> = frv179
Enter new 6 char. name of fg 756 ?> = frv180
Enter new 6 char. name of fg 757 ?> = frv181
Enter new 6 char. name of fg 758 ?> = frv182
Enter new 6 char. name of fg 759 ?> = frv183
Enter new 6 char. name of fg 760 ?> = frv184
Enter new 6 char. name of fg 761 ?> = frv185
Enter new 6 char. name of fg 762 ?> = frv186
Enter new 6 char. name of fg 763 ?> = frv187
Enter new 6 char. name of fg 764 ?> = frv188
Enter new 6 char. name of fg 765 ?> = frv189
Enter new 6 char. name of fg 766 ?> = frv190
Enter new 6 char. name of fg 767 ?> = frv191
Enter new 6 char. name of fg 768 ?> = fdisp
Enter new 6 char. name of fg 769 ?> = fhold
Enter new 6 char. name of fg 770 ?> = fning
Enter new 6 char. name of fg 771 ?> = fabort
Enter new 6 char. name of fg 772 ?> = fstart
Enter new 6 char. name of fg 773 ?> = ftrip0
Enter new 6 char. name of fg 774 ?> = ftrip1
Enter new 6 char. name of fg 775 ?> = ftrip2
Enter new 6 char. name of fg 776 ?> = fspar0
Enter new 6 char. name of fg 777 ?> = fspar1
Enter new 6 char. name of fg 778 ?> = fspar2
Enter new 6 char. name of fg 779 ?> = fspar3
Enter new 6 char. name of fg 780 ?> = fspar4
Enter new 6 char. name of fg 781 ?> = fspar5
Enter new 6 char. name of fg 782 ?> = fspar6
Enter new 6 char. name of fg 783 ?> = fspar7
MUPF?> = add 306 in
Enter new 6 char. name of in 0 ?> = line0
Enter new 6 char. name of in 1 ?> = line1
Enter new 6 char. name of in 2 ?> = line2
Enter new 6 char. name of in 3 ?> = line3
Enter new 6 char. name of in 4 ?> = line4
Enter new 6 char. name of in 5 ?> = line5
Enter new 6 char. name of in 6 ?> = line6
Enter new 6 char. name of in 7 ?> = line7
Enter new 6 char. name of in 8 ?> = line8
Enter new 6 char. name of in 9 ?> = line9

```

```

Enter new 6 char. name of in 10 ?> = line10
Enter new 6 char. name of in 11 ?> = line11
Enter new 6 char. name of in 12 ?> = line12
Enter new 6 char. name of in 13 ?> = line13
Enter new 6 char. name of in 14 ?> = line14
Enter new 6 char. name of in 15 ?> = line15
Enter new 6 char. name of in 16 ?> = line16
Enter new 6 char. name of in 17 ?> = line17
Enter new 6 char. name of in 18 ?> = line18
Enter new 6 char. name of in 19 ?> = line19
Enter new 6 char. name of in 20 ?> = line20
Enter new 6 char. name of in 21 ?> = line21
Enter new 6 char. name of in 22 ?> = line22
Enter new 6 char. name of in 23 ?> = line23
Enter new 6 char. name of in 24 ?> = line24
Enter new 6 char. name of in 25 ?> = line25
Enter new 6 char. name of in 26 ?> = line26
Enter new 6 char. name of in 27 ?> = line27
Enter new 6 char. name of in 28 ?> = line28
Enter new 6 char. name of in 29 ?> = line29
Enter new 6 char. name of in 30 ?> = line30
Enter new 6 char. name of in 31 ?> = line31
Enter new 6 char. name of in 32 ?> = line32
Enter new 6 char. name of in 33 ?> = line33
Enter new 6 char. name of in 34 ?> = line34
Enter new 6 char. name of in 35 ?> = line35
Enter new 6 char. name of in 36 ?> = line36
Enter new 6 char. name of in 37 ?> = line37
Enter new 6 char. name of in 38 ?> = line38
Enter new 6 char. name of in 39 ?> = line39
Enter new 6 char. name of in 40 ?> = line40
Enter new 6 char. name of in 41 ?> = line41
Enter new 6 char. name of in 42 ?> = line42
Enter new 6 char. name of in 43 ?> = line43
Enter new 6 char. name of in 44 ?> = line44
Enter new 6 char. name of in 45 ?> = line45
Enter new 6 char. name of in 46 ?> = line46
Enter new 6 char. name of in 47 ?> = line47
Enter new 6 char. name of in 48 ?> = bitval
Enter new 6 char. name of in 49 ?> = iloop
Enter new 6 char. name of in 50 ?> = jloop
Enter new 6 char. name of in 51 ?> = kloop
Enter new 6 char. name of in 52 ?> = first
Enter new 6 char. name of in 53 ?> = last
Enter new 6 char. name of in 54 ?> = index
Enter new 6 char. name of in 55 ?> = trip
Enter new 6 char. name of in 56 ?> = nadi
Enter new 6 char. name of in 57 ?> = blkval
Enter new 6 char. name of in 58 ?> = sort
Enter new 6 char. name of in 59 ?> = bval

```

Enter new 6 char. name of in 60 ?> = hldstp
Enter new 6 char. name of in 61 ?> = remain
Enter new 6 char. name of in 62 ?> = white
Enter new 6 char. name of in 63 ?> = analog
Enter new 6 char. name of in 64 ?> = errn
Enter new 6 char. name of in 65 ?> = errl
Enter new 6 char. name of in 66 ?> = grafx0
Enter new 6 char. name of in 67 ?> = grafx1
Enter new 6 char. name of in 68 ?> = grafx2
Enter new 6 char. name of in 69 ?> = grafx3
Enter new 6 char. name of in 70 ?> = grafx4
Enter new 6 char. name of in 71 ?> = grafx5
Enter new 6 char. name of in 72 ?> = grafx6
Enter new 6 char. name of in 73 ?> = grafx7
Enter new 6 char. name of in 74 ?> = grafx8
Enter new 6 char. name of in 75 ?> = grafx9
Enter new 6 char. name of in 76 ?> = graf10
Enter new 6 char. name of in 77 ?> = graf11
Enter new 6 char. name of in 78 ?> = graf12
Enter new 6 char. name of in 79 ?> = graf13
Enter new 6 char. name of in 80 ?> = graf14
Enter new 6 char. name of in 81 ?> = graf15
Enter new 6 char. name of in 82 ?> = graf16
Enter new 6 char. name of in 83 ?> = graf17
Enter new 6 char. name of in 84 ?> = graf18
Enter new 6 char. name of in 85 ?> = graf19
Enter new 6 char. name of in 86 ?> = graf20
Enter new 6 char. name of in 87 ?> = graf21
Enter new 6 char. name of in 88 ?> = graf22
Enter new 6 char. name of in 89 ?> = graf23
Enter new 6 char. name of in 90 ?> = graf24
Enter new 6 char. name of in 91 ?> = graf25
Enter new 6 char. name of in 92 ?> = graf26
Enter new 6 char. name of in 93 ?> = graf27
Enter new 6 char. name of in 94 ?> = graf28
Enter new 6 char. name of in 95 ?> = graf29
Enter new 6 char. name of in 96 ?> = graf30
Enter new 6 char. name of in 97 ?> = graf31
Enter new 6 char. name of in 98 ?> = graf32
Enter new 6 char. name of in 99 ?> = graf33
Enter new 6 char. name of in 100 ?> = graf34
Enter new 6 char. name of in 101 ?> = graf35
Enter new 6 char. name of in 102 ?> = graf36
Enter new 6 char. name of in 103 ?> = graf37
Enter new 6 char. name of in 104 ?> = graf38
Enter new 6 char. name of in 105 ?> = graf39
Enter new 6 char. name of in 106 ?> = graf40
Enter new 6 char. name of in 107 ?> = graf41
Enter new 6 char. name of in 108 ?> = graf42
Enter new 6 char. name of in 109 ?> = graf43

Enter new 6 char. name of in 110 ?> = graf44
Enter new 6 char. name of in 111 ?> = graf45
Enter new 6 char. name of in 112 ?> = graf46
Enter new 6 char. name of in 113 ?> = graf47
Enter new 6 char. name of in 114 ?> = int0
Enter new 6 char. name of in 115 ?> = int1
Enter new 6 char. name of in 116 ?> = int2
Enter new 6 char. name of in 117 ?> = int3
Enter new 6 char. name of in 118 ?> = int4
Enter new 6 char. name of in 119 ?> = int5
Enter new 6 char. name of in 120 ?> = int6
Enter new 6 char. name of in 121 ?> = int7
Enter new 6 char. name of in 122 ?> = int8
Enter new 6 char. name of in 123 ?> = int9
Enter new 6 char. name of in 124 ?> = int10
Enter new 6 char. name of in 125 ?> = int11
Enter new 6 char. name of in 126 ?> = int12
Enter new 6 char. name of in 127 ?> = int13
Enter new 6 char. name of in 128 ?> = int14
Enter new 6 char. name of in 129 ?> = int15
Enter new 6 char. name of in 130 ?> = int16
Enter new 6 char. name of in 131 ?> = int17
Enter new 6 char. name of in 132 ?> = int18
Enter new 6 char. name of in 133 ?> = int19
Enter new 6 char. name of in 134 ?> = int20
Enter new 6 char. name of in 135 ?> = int21
Enter new 6 char. name of in 136 ?> = int22
Enter new 6 char. name of in 137 ?> = int23
Enter new 6 char. name of in 138 ?> = int24
Enter new 6 char. name of in 139 ?> = int25
Enter new 6 char. name of in 140 ?> = int26
Enter new 6 char. name of in 141 ?> = int27
Enter new 6 char. name of in 142 ?> = int28
Enter new 6 char. name of in 143 ?> = int29
Enter new 6 char. name of in 144 ?> = int30
Enter new 6 char. name of in 145 ?> = int31
Enter new 6 char. name of in 146 ?> = int32
Enter new 6 char. name of in 147 ?> = int33
Enter new 6 char. name of in 148 ?> = int34
Enter new 6 char. name of in 149 ?> = int35
Enter new 6 char. name of in 150 ?> = int36
Enter new 6 char. name of in 151 ?> = int37
Enter new 6 char. name of in 152 ?> = int38
Enter new 6 char. name of in 153 ?> = int39
Enter new 6 char. name of in 154 ?> = int40
Enter new 6 char. name of in 155 ?> = int41
Enter new 6 char. name of in 156 ?> = int42
Enter new 6 char. name of in 157 ?> = int43
Enter new 6 char. name of in 158 ?> = int44
Enter new 6 char. name of in 159 ?> = int45

Enter new 6 char. name of in 160 ?> = int46
Enter new 6 char. name of in 161 ?> = int47
Enter new 6 char. name of in 162 ?> = int48
Enter new 6 char. name of in 163 ?> = int49
Enter new 6 char. name of in 164 ?> = int50
Enter new 6 char. name of in 165 ?> = int51
Enter new 6 char. name of in 166 ?> = int52
Enter new 6 char. name of in 167 ?> = int53
Enter new 6 char. name of in 168 ?> = int54
Enter new 6 char. name of in 169 ?> = int55
Enter new 6 char. name of in 170 ?> = int56
Enter new 6 char. name of in 171 ?> = int57
Enter new 6 char. name of in 172 ?> = int58
Enter new 6 char. name of in 173 ?> = int59
Enter new 6 char. name of in 174 ?> = int60
Enter new 6 char. name of in 175 ?> = int61
Enter new 6 char. name of in 176 ?> = int62
Enter new 6 char. name of in 177 ?> = int63
Enter new 6 char. name of in 178 ?> = int64
Enter new 6 char. name of in 179 ?> = int65
Enter new 6 char. name of in 180 ?> = int66
Enter new 6 char. name of in 181 ?> = int67
Enter new 6 char. name of in 182 ?> = int68
Enter new 6 char. name of in 183 ?> = int69
Enter new 6 char. name of in 184 ?> = int70
Enter new 6 char. name of in 185 ?> = int71
Enter new 6 char. name of in 186 ?> = int72
Enter new 6 char. name of in 187 ?> = int73
Enter new 6 char. name of in 188 ?> = int74
Enter new 6 char. name of in 189 ?> = int75
Enter new 6 char. name of in 190 ?> = int76
Enter new 6 char. name of in 191 ?> = int77
Enter new 6 char. name of in 192 ?> = int78
Enter new 6 char. name of in 193 ?> = int79
Enter new 6 char. name of in 194 ?> = int80
Enter new 6 char. name of in 195 ?> = int81
Enter new 6 char. name of in 196 ?> = int82
Enter new 6 char. name of in 197 ?> = int83
Enter new 6 char. name of in 198 ?> = int84
Enter new 6 char. name of in 199 ?> = int85
Enter new 6 char. name of in 200 ?> = int86
Enter new 6 char. name of in 201 ?> = int87
Enter new 6 char. name of in 202 ?> = int88
Enter new 6 char. name of in 203 ?> = int89
Enter new 6 char. name of in 204 ?> = int90
Enter new 6 char. name of in 205 ?> = int91
Enter new 6 char. name of in 206 ?> = int92
Enter new 6 char. name of in 207 ?> = int93
Enter new 6 char. name of in 208 ?> = int94
Enter new 6 char. name of in 209 ?> = int95

Enter new 6 char. name of in 210 ?> = int96
Enter new 6 char. name of in 211 ?> = int97
Enter new 6 char. name of in 212 ?> = int98
Enter new 6 char. name of in 213 ?> = int99
Enter new 6 char. name of in 214 ?> = int100
Enter new 6 char. name of in 215 ?> = int101
Enter new 6 char. name of in 216 ?> = int102
Enter new 6 char. name of in 217 ?> = int103
Enter new 6 char. name of in 218 ?> = int104
Enter new 6 char. name of in 219 ?> = int105
Enter new 6 char. name of in 220 ?> = int106
Enter new 6 char. name of in 221 ?> = int107
Enter new 6 char. name of in 222 ?> = int108
Enter new 6 char. name of in 223 ?> = int109
Enter new 6 char. name of in 224 ?> = int110
Enter new 6 char. name of in 225 ?> = int111
Enter new 6 char. name of in 226 ?> = int112
Enter new 6 char. name of in 227 ?> = int113
Enter new 6 char. name of in 228 ?> = int114
Enter new 6 char. name of in 229 ?> = int115
Enter new 6 char. name of in 230 ?> = int116
Enter new 6 char. name of in 231 ?> = int117
Enter new 6 char. name of in 232 ?> = int118
Enter new 6 char. name of in 233 ?> = int119
Enter new 6 char. name of in 234 ?> = int120
Enter new 6 char. name of in 235 ?> = int121
Enter new 6 char. name of in 236 ?> = int122
Enter new 6 char. name of in 237 ?> = int123
Enter new 6 char. name of in 238 ?> = int124
Enter new 6 char. name of in 239 ?> = int125
Enter new 6 char. name of in 240 ?> = int126
Enter new 6 char. name of in 241 ?> = int127
Enter new 6 char. name of in 242 ?> = int128
Enter new 6 char. name of in 243 ?> = int129
Enter new 6 char. name of in 244 ?> = int130
Enter new 6 char. name of in 245 ?> = int131
Enter new 6 char. name of in 246 ?> = int132
Enter new 6 char. name of in 247 ?> = int133
Enter new 6 char. name of in 248 ?> = int134
Enter new 6 char. name of in 249 ?> = int135
Enter new 6 char. name of in 250 ?> = int136
Enter new 6 char. name of in 251 ?> = int137
Enter new 6 char. name of in 252 ?> = int138
Enter new 6 char. name of in 253 ?> = int139
Enter new 6 char. name of in 254 ?> = int140
Enter new 6 char. name of in 255 ?> = int141
Enter new 6 char. name of in 256 ?> = int142
Enter new 6 char. name of in 257 ?> = int143
Enter new 6 char. name of in 258 ?> = int144
Enter new 6 char. name of in 259 ?> = int145

```

Enter new 6 char. name of in 260 ?> = int146
Enter new 6 char. name of in 261 ?> = int147
Enter new 6 char. name of in 262 ?> = int148
Enter new 6 char. name of in 263 ?> = int149
Enter new 6 char. name of in 264 ?> = int150
Enter new 6 char. name of in 265 ?> = int151
Enter new 6 char. name of in 266 ?> = int152
Enter new 6 char. name of in 267 ?> = int153
Enter new 6 char. name of in 268 ?> = int154
Enter new 6 char. name of in 269 ?> = int155
Enter new 6 char. name of in 270 ?> = int156
Enter new 6 char. name of in 271 ?> = int157
Enter new 6 char. name of in 272 ?> = int158
Enter new 6 char. name of in 273 ?> = int159
Enter new 6 char. name of in 274 ?> = int160
Enter new 6 char. name of in 275 ?> = int161
Enter new 6 char. name of in 276 ?> = int162
Enter new 6 char. name of in 277 ?> = int163
Enter new 6 char. name of in 278 ?> = int164
Enter new 6 char. name of in 279 ?> = int165
Enter new 6 char. name of in 280 ?> = int166
Enter new 6 char. name of in 281 ?> = int167
Enter new 6 char. name of in 282 ?> = int168
Enter new 6 char. name of in 283 ?> = int169
Enter new 6 char. name of in 284 ?> = int170
Enter new 6 char. name of in 285 ?> = int171
Enter new 6 char. name of in 286 ?> = int172
Enter new 6 char. name of in 287 ?> = int173
Enter new 6 char. name of in 288 ?> = int174
Enter new 6 char. name of in 289 ?> = int175
Enter new 6 char. name of in 290 ?> = int176
Enter new 6 char. name of in 291 ?> = int177
Enter new 6 char. name of in 292 ?> = int178
Enter new 6 char. name of in 293 ?> = int179
Enter new 6 char. name of in 294 ?> = int180
Enter new 6 char. name of in 295 ?> = int181
Enter new 6 char. name of in 296 ?> = int182
Enter new 6 char. name of in 297 ?> = int183
Enter new 6 char. name of in 298 ?> = int184
Enter new 6 char. name of in 299 ?> = int185
Enter new 6 char. name of in 300 ?> = int186
Enter new 6 char. name of in 301 ?> = int187
Enter new 6 char. name of in 302 ?> = int188
Enter new 6 char. name of in 303 ?> = int189
Enter new 6 char. name of in 304 ?> = int190
Enter new 6 char. name of in 305 ?> = int191
MUPF?> = add 100 fl
Enter new 6 char. name of fl 0 ?> = trip0
Enter new 6 char. name of fl 1 ?> = trip1
Enter new 6 char. name of fl 2 ?> = trip2

```

Enter new 6 char. name of fl 3 ?> = trip3
 Enter new 6 char. name of fl 4 ?> = trip4
 Enter new 6 char. name of fl 5 ?> = trip5
 Enter new 6 char. name of fl 6 ?> = trip6
 Enter new 6 char. name of fl 7 ?> = trip7
 Enter new 6 char. name of fl 8 ?> = trip8
 Enter new 6 char. name of fl 9 ?> = trip9
 Enter new 6 char. name of fl 10 ?> = ai0
 Enter new 6 char. name of fl 11 ?> = ai1
 Enter new 6 char. name of fl 12 ?> = ai2
 Enter new 6 char. name of fl 13 ?> = ai3
 Enter new 6 char. name of fl 14 ?> = ai4
 Enter new 6 char. name of fl 15 ?> = ai5
 Enter new 6 char. name of fl 16 ?> = ai6
 Enter new 6 char. name of fl 17 ?> = ai7
 Enter new 6 char. name of fl 18 ?> = ai8
 Enter new 6 char. name of fl 19 ?> = ai9
 Enter new 6 char. name of fl 20 ?> = ai10
 Enter new 6 char. name of fl 21 ?> = ai11
 Enter new 6 char. name of fl 22 ?> = ai12
 Enter new 6 char. name of fl 23 ?> = ai13
 Enter new 6 char. name of fl 24 ?> = ai14
 Enter new 6 char. name of fl 25 ?> = ai15
 Enter new 6 char. name of fl 26 ?> = ai16
 Enter new 6 char. name of fl 27 ?> = ai17
 Enter new 6 char. name of fl 28 ?> = ai18
 Enter new 6 char. name of fl 29 ?> = ai19
 Enter new 6 char. name of fl 30 ?> = ai20
 Enter new 6 char. name of fl 31 ?> = ai21
 Enter new 6 char. name of fl 32 ?> = ai22
 Enter new 6 char. name of fl 33 ?> = ai23
 Enter new 6 char. name of fl 34 ?> = ai24
 Enter new 6 char. name of fl 35 ?> = ai25
 Enter new 6 char. name of fl 36 ?> = ai26
 Enter new 6 char. name of fl 37 ?> = ai27
 Enter new 6 char. name of fl 38 ?> = ai28
 Enter new 6 char. name of fl 39 ?> = ai29
 Enter new 6 char. name of fl 40 ?> = ai30
 Enter new 6 char. name of fl 41 ?> = ai31
 Enter new 6 char. name of fl 42 ?> = ai32
 Enter new 6 char. name of fl 43 ?> = ai33
 Enter new 6 char. name of fl 44 ?> = ai34
 Enter new 6 char. name of fl 45 ?> = ai35
 Enter new 6 char. name of fl 46 ?> = ai36
 Enter new 6 char. name of fl 47 ?> = ai37
 Enter new 6 char. name of fl 48 ?> = ai38
 Enter new 6 char. name of fl 49 ?> = ai39
 Enter new 6 char. name of fl 50 ?> = ai40
 Enter new 6 char. name of fl 51 ?> = ai41
 Enter new 6 char. name of fl 52 ?> = ai42

```

Enter new 6 char. name of fl 53 ?> = ai43
Enter new 6 char. name of fl 54 ?> = ai44
Enter new 6 char. name of fl 55 ?> = ai45
Enter new 6 char. name of fl 56 ?> = ai46
Enter new 6 char. name of fl 57 ?> = ai47
Enter new 6 char. name of fl 58 ?> = ai48
Enter new 6 char. name of fl 59 ?> = ai49
Enter new 6 char. name of fl 60 ?> = ai50
Enter new 6 char. name of fl 61 ?> = ai51
Enter new 6 char. name of fl 62 ?> = ai52
Enter new 6 char. name of fl 63 ?> = ai53
Enter new 6 char. name of fl 64 ?> = ai54
Enter new 6 char. name of fl 65 ?> = ai55
Enter new 6 char. name of fl 66 ?> = ai56
Enter new 6 char. name of fl 67 ?> = ai57
Enter new 6 char. name of fl 68 ?> = ai58
Enter new 6 char. name of fl 69 ?> = ai59
Enter new 6 char. name of fl 70 ?> = ai60
Enter new 6 char. name of fl 71 ?> = ai61
Enter new 6 char. name of fl 72 ?> = ai62
Enter new 6 char. name of fl 73 ?> = ai63
Enter new 6 char. name of fl 74 ?> = ai64
Enter new 6 char. name of fl 75 ?> = ai65
Enter new 6 char. name of fl 76 ?> = ai66
Enter new 6 char. name of fl 77 ?> = ai67
Enter new 6 char. name of fl 78 ?> = ai68
Enter new 6 char. name of fl 79 ?> = ai69
Enter new 6 char. name of fl 80 ?> = ai70
Enter new 6 char. name of fl 81 ?> = ai71
Enter new 6 char. name of fl 82 ?> = ai72
Enter new 6 char. name of fl 83 ?> = ai73
Enter new 6 char. name of fl 84 ?> = ai74
Enter new 6 char. name of fl 85 ?> = ai75
Enter new 6 char. name of fl 86 ?> = ai76
Enter new 6 char. name of fl 87 ?> = ai77
Enter new 6 char. name of fl 88 ?> = ai78
Enter new 6 char. name of fl 89 ?> = ai79
Enter new 6 char. name of fl 90 ?> = float0
Enter new 6 char. name of fl 91 ?> = float1
Enter new 6 char. name of fl 92 ?> = float2
Enter new 6 char. name of fl 93 ?> = float3
Enter new 6 char. name of fl 94 ?> = float4
Enter new 6 char. name of fl 95 ?> = float5
Enter new 6 char. name of fl 96 ?> = float6
Enter new 6 char. name of fl 97 ?> = float7
Enter new 6 char. name of fl 98 ?> = float8
Enter new 6 char. name of fl 99 ?> = float9
MUPF?> = add 20 pt
Enter new 6 char. name of pt 0 ?> = pdate
Enter new 6 char. name of pt 1 ?> = ptime

```

```

Enter new 6 char. name of pt 2 ?> = pepn
Enter new 6 char. name of pt 3 ?> = pmessg
Enter new 6 char. name of pt 4 ?> = pgrafx
Enter new 6 char. name of pt 5 ?> = palarm
Enter new 6 char. name of pt 6 ?> = palm
Enter new 6 char. name of pt 7 ?> = pstat
Enter new 6 char. name of pt 8 ?> = pnext
Enter new 6 char. name of pt 9 ?> = plast
Enter new 6 char. name of pt 10 ?> = pline
Enter new 6 char. name of pt 11 ?> = ptrip
Enter new 6 char. name of pt 12 ?> = pack
Enter new 6 char. name of pt 13 ?> = pdsabl
Enter new 6 char. name of pt 14 ?> = pback
Enter new 6 char. name of pt 15 ?> = preps
Enter new 6 char. name of pt 16 ?> = ptype
Enter new 6 char. name of pt 17 ?> = prev
Enter new 6 char. name of pt 18 ?> = pspar0
Enter new 6 char. name of pt 19 ?> = pspar1
MUPF?> = add 111 sg
Enter new 6 char. name of sg 0 ?> = date0
Enter max string length for sg 0 ?> = 10
Enter new 6 char. name of sg 1 ?> = date1
Enter max string length for sg 1 ?> = 10
Enter new 6 char. name of sg 2 ?> = date2
Enter max string length for sg 2 ?> = 10
Enter new 6 char. name of sg 3 ?> = date3
Enter max string length for sg 3 ?> = 10
Enter new 6 char. name of sg 4 ?> = date4
Enter max string length for sg 4 ?> = 10
Enter new 6 char. name of sg 5 ?> = time0
Enter max string length for sg 5 ?> = 8
Enter new 6 char. name of sg 6 ?> = time1
Enter max string length for sg 6 ?> = 8
Enter new 6 char. name of sg 7 ?> = time2
Enter max string length for sg 7 ?> = 8
Enter new 6 char. name of sg 8 ?> = time3
Enter max string length for sg 8 ?> = 8
Enter new 6 char. name of sg 9 ?> = time4
Enter max string length for sg 9 ?> = 8
Enter new 6 char. name of sg 10 ?> = epn0
Enter max string length for sg 10 ?> = 10
Enter new 6 char. name of sg 11 ?> = epn1
Enter max string length for sg 11 ?> = 10
Enter new 6 char. name of sg 12 ?> = epn2
Enter max string length for sg 12 ?> = 10
Enter new 6 char. name of sg 13 ?> = epn3
Enter max string length for sg 13 ?> = 10
Enter new 6 char. name of sg 14 ?> = epn4
Enter max string length for sg 14 ?> = 10
Enter new 6 char. name of sg 15 ?> = mssg0

```

Enter max string length for sg 15 ?> = 30
Enter new 6 char. name of sg 16 ?> = mssg1
Enter max string length for sg 16 ?> = 30
Enter new 6 char. name of sg 17 ?> = mssg2
Enter max string length for sg 17 ?> = 30
Enter new 6 char. name of sg 18 ?> = mssg3
Enter max string length for sg 18 ?> = 30
Enter new 6 char. name of sg 19 ?> = mssg4
Enter max string length for sg 19 ?> = 30
Enter new 6 char. name of sg 20 ?> = mssg5
Enter max string length for sg 20 ?> = 30
Enter new 6 char. name of sg 21 ?> = mssg6
Enter max string length for sg 21 ?> = 30
Enter new 6 char. name of sg 22 ?> = mssg7
Enter max string length for sg 22 ?> = 30
Enter new 6 char. name of sg 23 ?> = mssg8
Enter max string length for sg 23 ?> = 30
Enter new 6 char. name of sg 24 ?> = mssg9
Enter max string length for sg 24 ?> = 30
Enter new 6 char. name of sg 25 ?> = mssg10
Enter max string length for sg 25 ?> = 30
Enter new 6 char. name of sg 26 ?> = mssg11
Enter max string length for sg 26 ?> = 30
Enter new 6 char. name of sg 27 ?> = mssg12
Enter max string length for sg 27 ?> = 30
Enter new 6 char. name of sg 28 ?> = mssg13
Enter max string length for sg 28 ?> = 30
Enter new 6 char. name of sg 29 ?> = mssg14
Enter max string length for sg 29 ?> = 30
Enter new 6 char. name of sg 30 ?> = mssg15
Enter max string length for sg 30 ?> = 30
Enter new 6 char. name of sg 31 ?> = mssg16
Enter max string length for sg 31 ?> = 30
Enter new 6 char. name of sg 32 ?> = mssg17
Enter max string length for sg 32 ?> = 30
Enter new 6 char. name of sg 33 ?> = mssg18
Enter max string length for sg 33 ?> = 30
Enter new 6 char. name of sg 34 ?> = mssg19
Enter max string length for sg 34 ?> = 30
Enter new 6 char. name of sg 35 ?> = mssg20
Enter max string length for sg 35 ?> = 30
Enter new 6 char. name of sg 36 ?> = mssg21
Enter max string length for sg 36 ?> = 30
Enter new 6 char. name of sg 37 ?> = mssg22
Enter max string length for sg 37 ?> = 30
Enter new 6 char. name of sg 38 ?> = mssg23
Enter max string length for sg 38 ?> = 30
Enter new 6 char. name of sg 39 ?> = mssg24
Enter max string length for sg 39 ?> = 30
Enter new 6 char. name of sg 40 ?> = mssg25

Enter max string length for sg 40 ?> = 30
 Enter new 6 char. name of sg 41 ?> = mssg26
 Enter max string length for sg 41 ?> = 30
 Enter new 6 char. name of sg 42 ?> = mssg27
 Enter max string length for sg 42 ?> = 30
 Enter new 6 char. name of sg 43 ?> = mssg28
 Enter max string length for sg 43 ?> = 30
 Enter new 6 char. name of sg 44 ?> = mssg29
 Enter max string length for sg 44 ?> = 30
 Enter new 6 char. name of sg 45 ?> = mssg30
 Enter max string length for sg 45 ?> = 30
 Enter new 6 char. name of sg 46 ?> = mssg31
 Enter max string length for sg 46 ?> = 30
 Enter new 6 char. name of sg 47 ?> = mssg32
 Enter max string length for sg 47 ?> = 30
 Enter new 6 char. name of sg 48 ?> = mssg33
 Enter max string length for sg 48 ?> = 30
 Enter new 6 char. name of sg 49 ?> = mssg34
 Enter max string length for sg 49 ?> = 30
 Enter new 6 char. name of sg 50 ?> = mssg35
 Enter max string length for sg 50 ?> = 30
 Enter new 6 char. name of sg 51 ?> = mssg36
 Enter max string length for sg 51 ?> = 30
 Enter new 6 char. name of sg 52 ?> = mssg37
 Enter max string length for sg 52 ?> = 30
 Enter new 6 char. name of sg 53 ?> = mssg38
 Enter max string length for sg 53 ?> = 30
 Enter new 6 char. name of sg 54 ?> = mssg39
 Enter max string length for sg 54 ?> = 30
 Enter new 6 char. name of sg 55 ?> = mssg40
 Enter max string length for sg 55 ?> = 30
 Enter new 6 char. name of sg 56 ?> = mssg41
 Enter max string length for sg 56 ?> = 30
 Enter new 6 char. name of sg 57 ?> = mssg42
 Enter max string length for sg 57 ?> = 30
 Enter new 6 char. name of sg 58 ?> = mssg43
 Enter max string length for sg 58 ?> = 30
 Enter new 6 char. name of sg 59 ?> = mssg44
 Enter max string length for sg 59 ?> = 30
 Enter new 6 char. name of sg 60 ?> = mssg45
 Enter max string length for sg 60 ?> = 30
 Enter new 6 char. name of sg 61 ?> = mssg46
 Enter max string length for sg 61 ?> = 30
 Enter new 6 char. name of sg 62 ?> = mssg47
 Enter max string length for sg 62 ?> = 30
 Enter new 6 char. name of sg 63 ?> = lsg0
 Enter max string length for sg 63 ?> = 30
 Enter new 6 char. name of sg 64 ?> = lsg1
 Enter max string length for sg 64 ?> = 30
 Enter new 6 char. name of sg 65 ?> = lsg2

Enter max string length for sg 65 ?> = 30
Enter new 6 char. name of sg 66 ?> = lsg3
Enter max string length for sg 66 ?> = 30
Enter new 6 char. name of sg 67 ?> = lsg4
Enter max string length for sg 67 ?> = 30
Enter new 6 char. name of sg 68 ?> = lsg5
Enter max string length for sg 68 ?> = 30
Enter new 6 char. name of sg 69 ?> = lsg6
Enter max string length for sg 69 ?> = 30
Enter new 6 char. name of sg 70 ?> = lsg7
Enter max string length for sg 70 ?> = 30
Enter new 6 char. name of sg 71 ?> = lsg8
Enter max string length for sg 71 ?> = 30
Enter new 6 char. name of sg 72 ?> = lsg9
Enter max string length for sg 72 ?> = 30
Enter new 6 char. name of sg 73 ?> = lsg10
Enter max string length for sg 73 ?> = 30
Enter new 6 char. name of sg 74 ?> = lsg11
Enter max string length for sg 74 ?> = 30
Enter new 6 char. name of sg 75 ?> = lsg12
Enter max string length for sg 75 ?> = 30
Enter new 6 char. name of sg 76 ?> = lsg13
Enter max string length for sg 76 ?> = 30
Enter new 6 char. name of sg 77 ?> = lsg14
Enter max string length for sg 77 ?> = 30
Enter new 6 char. name of sg 78 ?> = lsg15
Enter max string length for sg 78 ?> = 30
Enter new 6 char. name of sg 79 ?> = lsg16
Enter max string length for sg 79 ?> = 30
Enter new 6 char. name of sg 80 ?> = lsg17
Enter max string length for sg 80 ?> = 30
Enter new 6 char. name of sg 81 ?> = lsg18
Enter max string length for sg 81 ?> = 30
Enter new 6 char. name of sg 82 ?> = lsg19
Enter max string length for sg 82 ?> = 30
Enter new 6 char. name of sg 83 ?> = lsg20
Enter max string length for sg 83 ?> = 30
Enter new 6 char. name of sg 84 ?> = lsg21
Enter max string length for sg 84 ?> = 30
Enter new 6 char. name of sg 85 ?> = lsg22
Enter max string length for sg 85 ?> = 30
Enter new 6 char. name of sg 86 ?> = lsg23
Enter max string length for sg 86 ?> = 30
Enter new 6 char. name of sg 87 ?> = lsg24
Enter max string length for sg 87 ?> = 30
Enter new 6 char. name of sg 88 ?> = lsg25
Enter max string length for sg 88 ?> = 30
Enter new 6 char. name of sg 89 ?> = lsg26
Enter max string length for sg 89 ?> = 30
Enter new 6 char. name of sg 90 ?> = lsg27

Enter max string length for sg 90 ?> = 30
Enter new 6 char. name of sg 91 ?> = lsg28
Enter max string length for sg 91 ?> = 30
Enter new 6 char. name of sg 92 ?> = lsg29
Enter max string length for sg 92 ?> = 30
Enter new 6 char. name of sg 93 ?> = lsg30
Enter max string length for sg 93 ?> = 30
Enter new 6 char. name of sg 94 ?> = lsg31
Enter max string length for sg 94 ?> = 30
Enter new 6 char. name of sg 95 ?> = lsg32
Enter max string length for sg 95 ?> = 30
Enter new 6 char. name of sg 96 ?> = lsg33
Enter max string length for sg 96 ?> = 30
Enter new 6 char. name of sg 97 ?> = lsg34
Enter max string length for sg 97 ?> = 30
Enter new 6 char. name of sg 98 ?> = lsg35
Enter max string length for sg 98 ?> = 30
Enter new 6 char. name of sg 99 ?> = lsg36
Enter max string length for sg 99 ?> = 30
Enter new 6 char. name of sg 100 ?> = lsg37
Enter max string length for sg 100 ?> = 30
Enter new 6 char. name of sg 101 ?> = lsg38
Enter max string length for sg 101 ?> = 30
Enter new 6 char. name of sg 102 ?> = lsg39
Enter max string length for sg 102 ?> = 30
Enter new 6 char. name of sg 103 ?> = lsg40
Enter max string length for sg 103 ?> = 30
Enter new 6 char. name of sg 104 ?> = lsg41
Enter max string length for sg 104 ?> = 30
Enter new 6 char. name of sg 105 ?> = lsg42
Enter max string length for sg 105 ?> = 30
Enter new 6 char. name of sg 106 ?> = lsg43
Enter max string length for sg 106 ?> = 30
Enter new 6 char. name of sg 107 ?> = lsg44
Enter max string length for sg 107 ?> = 30
Enter new 6 char. name of sg 108 ?> = lsg45
Enter max string length for sg 108 ?> = 30
Enter new 6 char. name of sg 109 ?> = lsg46
Enter max string length for sg 109 ?> = 30
Enter new 6 char. name of sg 110 ?> = lsg47
Enter max string length for sg 110 ?> = 30
MUPF?>
    end of file encountered
UPF?>
    end of file encountered

TNKFRM

```

UPF Version 6.3-2      05-Dec-94 09:42:07
UPF?> = de tnkfrm
UPF?> = cr tnkfrm,*
Enter 4 character name of RPF to associate?> = farm
MUPF?> = add 496 fg
Enter new 6 char. name of fg 0 ?> = faln0
Enter new 6 char. name of fg 1 ?> = faln1
Enter new 6 char. name of fg 2 ?> = faln2
Enter new 6 char. name of fg 3 ?> = faln3
Enter new 6 char. name of fg 4 ?> = faln4
Enter new 6 char. name of fg 5 ?> = faln5
Enter new 6 char. name of fg 6 ?> = faln6
Enter new 6 char. name of fg 7 ?> = faln7
Enter new 6 char. name of fg 8 ?> = faln8
Enter new 6 char. name of fg 9 ?> = faln9
Enter new 6 char. name of fg 10 ?> = faln10
Enter new 6 char. name of fg 11 ?> = faln11
Enter new 6 char. name of fg 12 ?> = faln12
Enter new 6 char. name of fg 13 ?> = faln13
Enter new 6 char. name of fg 14 ?> = faln14
Enter new 6 char. name of fg 15 ?> = faln15
Enter new 6 char. name of fg 16 ?> = faln16
Enter new 6 char. name of fg 17 ?> = faln17
Enter new 6 char. name of fg 18 ?> = faln18
Enter new 6 char. name of fg 19 ?> = faln19
Enter new 6 char. name of fg 20 ?> = faln20
Enter new 6 char. name of fg 21 ?> = faln21
Enter new 6 char. name of fg 22 ?> = faln22
Enter new 6 char. name of fg 23 ?> = faln23
Enter new 6 char. name of fg 24 ?> = faln24
Enter new 6 char. name of fg 25 ?> = faln25
Enter new 6 char. name of fg 26 ?> = faln26
Enter new 6 char. name of fg 27 ?> = faln27
Enter new 6 char. name of fg 28 ?> = faln28
Enter new 6 char. name of fg 29 ?> = faln29
Enter new 6 char. name of fg 30 ?> = faln30
Enter new 6 char. name of fg 31 ?> = faln31
Enter new 6 char. name of fg 32 ?> = faln32
Enter new 6 char. name of fg 33 ?> = faln33
Enter new 6 char. name of fg 34 ?> = faln34
Enter new 6 char. name of fg 35 ?> = faln35
Enter new 6 char. name of fg 36 ?> = faln36
Enter new 6 char. name of fg 37 ?> = faln37
Enter new 6 char. name of fg 38 ?> = faln38
Enter new 6 char. name of fg 39 ?> = faln39
Enter new 6 char. name of fg 40 ?> = faln40
Enter new 6 char. name of fg 41 ?> = faln41
Enter new 6 char. name of fg 42 ?> = faln42
Enter new 6 char. name of fg 43 ?> = faln43
Enter new 6 char. name of fg 44 ?> = faln44

```

Enter new 6 char. name of fg 45 ?> = fal45
Enter new 6 char. name of fg 46 ?> = fal46
Enter new 6 char. name of fg 47 ?> = fal47
Enter new 6 char. name of fg 48 ?> = fal48
Enter new 6 char. name of fg 49 ?> = fal49
Enter new 6 char. name of fg 50 ?> = fal50
Enter new 6 char. name of fg 51 ?> = fal51
Enter new 6 char. name of fg 52 ?> = fal52
Enter new 6 char. name of fg 53 ?> = fal53
Enter new 6 char. name of fg 54 ?> = fal54
Enter new 6 char. name of fg 55 ?> = fal55
Enter new 6 char. name of fg 56 ?> = fal56
Enter new 6 char. name of fg 57 ?> = fal57
Enter new 6 char. name of fg 58 ?> = fal58
Enter new 6 char. name of fg 59 ?> = fal59
Enter new 6 char. name of fg 60 ?> = fal60
Enter new 6 char. name of fg 61 ?> = fal61
Enter new 6 char. name of fg 62 ?> = fal62
Enter new 6 char. name of fg 63 ?> = fal63
Enter new 6 char. name of fg 64 ?> = fal64
Enter new 6 char. name of fg 65 ?> = fal65
Enter new 6 char. name of fg 66 ?> = fal66
Enter new 6 char. name of fg 67 ?> = fal67
Enter new 6 char. name of fg 68 ?> = fal68
Enter new 6 char. name of fg 69 ?> = fal69
Enter new 6 char. name of fg 70 ?> = fal70
Enter new 6 char. name of fg 71 ?> = fal71
Enter new 6 char. name of fg 72 ?> = fal72
Enter new 6 char. name of fg 73 ?> = fal73
Enter new 6 char. name of fg 74 ?> = fal74
Enter new 6 char. name of fg 75 ?> = fal75
Enter new 6 char. name of fg 76 ?> = fal76
Enter new 6 char. name of fg 77 ?> = fal77
Enter new 6 char. name of fg 78 ?> = fal78
Enter new 6 char. name of fg 79 ?> = fal79
Enter new 6 char. name of fg 80 ?> = fal80
Enter new 6 char. name of fg 81 ?> = fal81
Enter new 6 char. name of fg 82 ?> = fal82
Enter new 6 char. name of fg 83 ?> = fal83
Enter new 6 char. name of fg 84 ?> = fal84
Enter new 6 char. name of fg 85 ?> = fal85
Enter new 6 char. name of fg 86 ?> = fal86
Enter new 6 char. name of fg 87 ?> = fal87
Enter new 6 char. name of fg 88 ?> = fal88
Enter new 6 char. name of fg 89 ?> = fal89
Enter new 6 char. name of fg 90 ?> = fal90
Enter new 6 char. name of fg 91 ?> = fal91
Enter new 6 char. name of fg 92 ?> = fal92
Enter new 6 char. name of fg 93 ?> = fal93
Enter new 6 char. name of fg 94 ?> = fal94

Enter new 6 char. name of fg 95 ?> = fal95
Enter new 6 char. name of fg 96 ?> = fal96
Enter new 6 char. name of fg 97 ?> = fal97
Enter new 6 char. name of fg 98 ?> = fal98
Enter new 6 char. name of fg 99 ?> = fal99
Enter new 6 char. name of fg 100 ?> = fal100
Enter new 6 char. name of fg 101 ?> = fal101
Enter new 6 char. name of fg 102 ?> = fal102
Enter new 6 char. name of fg 103 ?> = fal103
Enter new 6 char. name of fg 104 ?> = fal104
Enter new 6 char. name of fg 105 ?> = fal105
Enter new 6 char. name of fg 106 ?> = fal106
Enter new 6 char. name of fg 107 ?> = fal107
Enter new 6 char. name of fg 108 ?> = fal108
Enter new 6 char. name of fg 109 ?> = fal109
Enter new 6 char. name of fg 110 ?> = fal110
Enter new 6 char. name of fg 111 ?> = fal111
Enter new 6 char. name of fg 112 ?> = fpas0
Enter new 6 char. name of fg 113 ?> = fpas1
Enter new 6 char. name of fg 114 ?> = fpas2
Enter new 6 char. name of fg 115 ?> = fpas3
Enter new 6 char. name of fg 116 ?> = fpas4
Enter new 6 char. name of fg 117 ?> = fpas5
Enter new 6 char. name of fg 118 ?> = fpas6
Enter new 6 char. name of fg 119 ?> = fpas7
Enter new 6 char. name of fg 120 ?> = fpas8
Enter new 6 char. name of fg 121 ?> = fpas9
Enter new 6 char. name of fg 122 ?> = fpas10
Enter new 6 char. name of fg 123 ?> = fpas11
Enter new 6 char. name of fg 124 ?> = fpas12
Enter new 6 char. name of fg 125 ?> = fpas13
Enter new 6 char. name of fg 126 ?> = fpas14
Enter new 6 char. name of fg 127 ?> = fpas15
Enter new 6 char. name of fg 128 ?> = fpas16
Enter new 6 char. name of fg 129 ?> = fpas17
Enter new 6 char. name of fg 130 ?> = fpas18
Enter new 6 char. name of fg 131 ?> = fpas19
Enter new 6 char. name of fg 132 ?> = fpas20
Enter new 6 char. name of fg 133 ?> = fpas21
Enter new 6 char. name of fg 134 ?> = fpas22
Enter new 6 char. name of fg 135 ?> = fpas23
Enter new 6 char. name of fg 136 ?> = fpas24
Enter new 6 char. name of fg 137 ?> = fpas25
Enter new 6 char. name of fg 138 ?> = fpas26
Enter new 6 char. name of fg 139 ?> = fpas27
Enter new 6 char. name of fg 140 ?> = fpas28
Enter new 6 char. name of fg 141 ?> = fpas29
Enter new 6 char. name of fg 142 ?> = fpas30
Enter new 6 char. name of fg 143 ?> = fpas31
Enter new 6 char. name of fg 144 ?> = fpas32

WHC-SD-534-CSWL-001 REV 3

Enter new 6 char. name of fg 145 ?> = fpas33
Enter new 6 char. name of fg 146 ?> = fpas34
Enter new 6 char. name of fg 147 ?> = fpas35
Enter new 6 char. name of fg 148 ?> = fpas36
Enter new 6 char. name of fg 149 ?> = fpas37
Enter new 6 char. name of fg 150 ?> = fpas38
Enter new 6 char. name of fg 151 ?> = fpas39
Enter new 6 char. name of fg 152 ?> = fpas40
Enter new 6 char. name of fg 153 ?> = fpas41
Enter new 6 char. name of fg 154 ?> = fpas42
Enter new 6 char. name of fg 155 ?> = fpas43
Enter new 6 char. name of fg 156 ?> = fpas44
Enter new 6 char. name of fg 157 ?> = fpas45
Enter new 6 char. name of fg 158 ?> = fpas46
Enter new 6 char. name of fg 159 ?> = fpas47
Enter new 6 char. name of fg 160 ?> = fpas48
Enter new 6 char. name of fg 161 ?> = fpas49
Enter new 6 char. name of fg 162 ?> = fpas50
Enter new 6 char. name of fg 163 ?> = fpas51
Enter new 6 char. name of fg 164 ?> = fpas52
Enter new 6 char. name of fg 165 ?> = fpas53
Enter new 6 char. name of fg 166 ?> = fpas54
Enter new 6 char. name of fg 167 ?> = fpas55
Enter new 6 char. name of fg 168 ?> = fpas56
Enter new 6 char. name of fg 169 ?> = fpas57
Enter new 6 char. name of fg 170 ?> = fpas58
Enter new 6 char. name of fg 171 ?> = fpas59
Enter new 6 char. name of fg 172 ?> = fpas60
Enter new 6 char. name of fg 173 ?> = fpas61
Enter new 6 char. name of fg 174 ?> = fpas62
Enter new 6 char. name of fg 175 ?> = fpas63
Enter new 6 char. name of fg 176 ?> = fpas64
Enter new 6 char. name of fg 177 ?> = fpas65
Enter new 6 char. name of fg 178 ?> = fpas66
Enter new 6 char. name of fg 179 ?> = fpas67
Enter new 6 char. name of fg 180 ?> = fpas68
Enter new 6 char. name of fg 181 ?> = fpas69
Enter new 6 char. name of fg 182 ?> = fpas70
Enter new 6 char. name of fg 183 ?> = fpas71
Enter new 6 char. name of fg 184 ?> = fpas72
Enter new 6 char. name of fg 185 ?> = fpas73
Enter new 6 char. name of fg 186 ?> = fpas74
Enter new 6 char. name of fg 187 ?> = fpas75
Enter new 6 char. name of fg 188 ?> = fpas76
Enter new 6 char. name of fg 189 ?> = fpas77
Enter new 6 char. name of fg 190 ?> = fpas78
Enter new 6 char. name of fg 191 ?> = fpas79
Enter new 6 char. name of fg 192 ?> = fpas80
Enter new 6 char. name of fg 193 ?> = fpas81
Enter new 6 char. name of fg 194 ?> = fpas82

Enter new 6 char. name of fg 195 ?> = fpas83
Enter new 6 char. name of fg 196 ?> = fpas84
Enter new 6 char. name of fg 197 ?> = fpas85
Enter new 6 char. name of fg 198 ?> = fpas86
Enter new 6 char. name of fg 199 ?> = fpas87
Enter new 6 char. name of fg 200 ?> = fpas88
Enter new 6 char. name of fg 201 ?> = fpas89
Enter new 6 char. name of fg 202 ?> = fpas90
Enter new 6 char. name of fg 203 ?> = fpas91
Enter new 6 char. name of fg 204 ?> = fpas92
Enter new 6 char. name of fg 205 ?> = fpas93
Enter new 6 char. name of fg 206 ?> = fpas94
Enter new 6 char. name of fg 207 ?> = fpas95
Enter new 6 char. name of fg 208 ?> = fpas96
Enter new 6 char. name of fg 209 ?> = fpas97
Enter new 6 char. name of fg 210 ?> = fpas98
Enter new 6 char. name of fg 211 ?> = fpas99
Enter new 6 char. name of fg 212 ?> = fpal00
Enter new 6 char. name of fg 213 ?> = fpal01
Enter new 6 char. name of fg 214 ?> = fpal02
Enter new 6 char. name of fg 215 ?> = fpal03
Enter new 6 char. name of fg 216 ?> = fpal04
Enter new 6 char. name of fg 217 ?> = fpal05
Enter new 6 char. name of fg 218 ?> = fpal06
Enter new 6 char. name of fg 219 ?> = fpal07
Enter new 6 char. name of fg 220 ?> = fpal08
Enter new 6 char. name of fg 221 ?> = fpal09
Enter new 6 char. name of fg 222 ?> = fpal10
Enter new 6 char. name of fg 223 ?> = fpal11
Enter new 6 char. name of fg 224 ?> = fskp0
Enter new 6 char. name of fg 225 ?> = fskp1
Enter new 6 char. name of fg 226 ?> = fskp2
Enter new 6 char. name of fg 227 ?> = fskp3
Enter new 6 char. name of fg 228 ?> = fskp4
Enter new 6 char. name of fg 229 ?> = fskp5
Enter new 6 char. name of fg 230 ?> = fskp6
Enter new 6 char. name of fg 231 ?> = fskp7
Enter new 6 char. name of fg 232 ?> = fskp8
Enter new 6 char. name of fg 233 ?> = fskp9
Enter new 6 char. name of fg 234 ?> = fskp10
Enter new 6 char. name of fg 235 ?> = fskp11
Enter new 6 char. name of fg 236 ?> = fskp12
Enter new 6 char. name of fg 237 ?> = fskp13
Enter new 6 char. name of fg 238 ?> = fskp14
Enter new 6 char. name of fg 239 ?> = fskp15
Enter new 6 char. name of fg 240 ?> = fskp16
Enter new 6 char. name of fg 241 ?> = fskp17
Enter new 6 char. name of fg 242 ?> = fskp18
Enter new 6 char. name of fg 243 ?> = fskp19
Enter new 6 char. name of fg 244 ?> = fskp20

Enter new 6 char. name of fg 245 ?> = fskp21
Enter new 6 char. name of fg 246 ?> = fskp22
Enter new 6 char. name of fg 247 ?> = fskp23
Enter new 6 char. name of fg 248 ?> = fskp24
Enter new 6 char. name of fg 249 ?> = fskp25
Enter new 6 char. name of fg 250 ?> = fskp26
Enter new 6 char. name of fg 251 ?> = fskp27
Enter new 6 char. name of fg 252 ?> = fskp28
Enter new 6 char. name of fg 253 ?> = fskp29
Enter new 6 char. name of fg 254 ?> = fskp30
Enter new 6 char. name of fg 255 ?> = fskp31
Enter new 6 char. name of fg 256 ?> = fskp32
Enter new 6 char. name of fg 257 ?> = fskp33
Enter new 6 char. name of fg 258 ?> = fskp34
Enter new 6 char. name of fg 259 ?> = fskp35
Enter new 6 char. name of fg 260 ?> = fskp36
Enter new 6 char. name of fg 261 ?> = fskp37
Enter new 6 char. name of fg 262 ?> = fskp38
Enter new 6 char. name of fg 263 ?> = fskp39
Enter new 6 char. name of fg 264 ?> = fskp40
Enter new 6 char. name of fg 265 ?> = fskp41
Enter new 6 char. name of fg 266 ?> = fskp42
Enter new 6 char. name of fg 267 ?> = fskp43
Enter new 6 char. name of fg 268 ?> = fskp44
Enter new 6 char. name of fg 269 ?> = fskp45
Enter new 6 char. name of fg 270 ?> = fskp46
Enter new 6 char. name of fg 271 ?> = fskp47
Enter new 6 char. name of fg 272 ?> = fskp48
Enter new 6 char. name of fg 273 ?> = fskp49
Enter new 6 char. name of fg 274 ?> = fskp50
Enter new 6 char. name of fg 275 ?> = fskp51
Enter new 6 char. name of fg 276 ?> = fskp52
Enter new 6 char. name of fg 277 ?> = fskp53
Enter new 6 char. name of fg 278 ?> = fskp54
Enter new 6 char. name of fg 279 ?> = fskp55
Enter new 6 char. name of fg 280 ?> = fskp56
Enter new 6 char. name of fg 281 ?> = fskp57
Enter new 6 char. name of fg 282 ?> = fskp58
Enter new 6 char. name of fg 283 ?> = fskp59
Enter new 6 char. name of fg 284 ?> = fskp60
Enter new 6 char. name of fg 285 ?> = fskp61
Enter new 6 char. name of fg 286 ?> = fskp62
Enter new 6 char. name of fg 287 ?> = fskp63
Enter new 6 char. name of fg 288 ?> = fskp64
Enter new 6 char. name of fg 289 ?> = fskp65
Enter new 6 char. name of fg 290 ?> = fskp66
Enter new 6 char. name of fg 291 ?> = fskp67
Enter new 6 char. name of fg 292 ?> = fskp68
Enter new 6 char. name of fg 293 ?> = fskp69
Enter new 6 char. name of fg 294 ?> = fskp70

Enter new 6 char. name of fg 295 ?> = fskp71
Enter new 6 char. name of fg 296 ?> = fskp72
Enter new 6 char. name of fg 297 ?> = fskp73
Enter new 6 char. name of fg 298 ?> = fskp74
Enter new 6 char. name of fg 299 ?> = fskp75
Enter new 6 char. name of fg 300 ?> = fskp76
Enter new 6 char. name of fg 301 ?> = fskp77
Enter new 6 char. name of fg 302 ?> = fskp78
Enter new 6 char. name of fg 303 ?> = fskp79
Enter new 6 char. name of fg 304 ?> = fskp80
Enter new 6 char. name of fg 305 ?> = fskp81
Enter new 6 char. name of fg 306 ?> = fskp82
Enter new 6 char. name of fg 307 ?> = fskp83
Enter new 6 char. name of fg 308 ?> = fskp84
Enter new 6 char. name of fg 309 ?> = fskp85
Enter new 6 char. name of fg 310 ?> = fskp86
Enter new 6 char. name of fg 311 ?> = fskp87
Enter new 6 char. name of fg 312 ?> = fskp88
Enter new 6 char. name of fg 313 ?> = fskp89
Enter new 6 char. name of fg 314 ?> = fskp90
Enter new 6 char. name of fg 315 ?> = fskp91
Enter new 6 char. name of fg 316 ?> = fskp92
Enter new 6 char. name of fg 317 ?> = fskp93
Enter new 6 char. name of fg 318 ?> = fskp94
Enter new 6 char. name of fg 319 ?> = fskp95
Enter new 6 char. name of fg 320 ?> = fskp96
Enter new 6 char. name of fg 321 ?> = fskp97
Enter new 6 char. name of fg 322 ?> = fskp98
Enter new 6 char. name of fg 323 ?> = fskp99
Enter new 6 char. name of fg 324 ?> = fsk100
Enter new 6 char. name of fg 325 ?> = fsk101
Enter new 6 char. name of fg 326 ?> = fsk102
Enter new 6 char. name of fg 327 ?> = fsk103
Enter new 6 char. name of fg 328 ?> = fsk104
Enter new 6 char. name of fg 329 ?> = fsk105
Enter new 6 char. name of fg 330 ?> = fsk106
Enter new 6 char. name of fg 331 ?> = fsk107
Enter new 6 char. name of fg 332 ?> = fsk108
Enter new 6 char. name of fg 333 ?> = fsk109
Enter new 6 char. name of fg 334 ?> = fsk110
Enter new 6 char. name of fg 335 ?> = fsk111
Enter new 6 char. name of fg 336 ?> = fx0
Enter new 6 char. name of fg 337 ?> = fx1
Enter new 6 char. name of fg 338 ?> = fx2
Enter new 6 char. name of fg 339 ?> = fx3
Enter new 6 char. name of fg 340 ?> = fx4
Enter new 6 char. name of fg 341 ?> = fx5
Enter new 6 char. name of fg 342 ?> = fx6
Enter new 6 char. name of fg 343 ?> = fx7
Enter new 6 char. name of fg 344 ?> = fx8

Enter new 6 char. name of fg 345 ?> = fx9
 Enter new 6 char. name of fg 346 ?> = fx10
 Enter new 6 char. name of fg 347 ?> = fx11
 Enter new 6 char. name of fg 348 ?> = fx12
 Enter new 6 char. name of fg 349 ?> = fx13
 Enter new 6 char. name of fg 350 ?> = fx14
 Enter new 6 char. name of fg 351 ?> = fx15
 Enter new 6 char. name of fg 352 ?> = fx16
 Enter new 6 char. name of fg 353 ?> = fx17
 Enter new 6 char. name of fg 354 ?> = fx18
 Enter new 6 char. name of fg 355 ?> = fx19
 Enter new 6 char. name of fg 356 ?> = fx20
 Enter new 6 char. name of fg 357 ?> = fx21
 Enter new 6 char. name of fg 358 ?> = fx22
 Enter new 6 char. name of fg 359 ?> = fx23
 Enter new 6 char. name of fg 360 ?> = fx24
 Enter new 6 char. name of fg 361 ?> = fx25
 Enter new 6 char. name of fg 362 ?> = fx26
 Enter new 6 char. name of fg 363 ?> = fx27
 Enter new 6 char. name of fg 364 ?> = fx28
 Enter new 6 char. name of fg 365 ?> = fx29
 Enter new 6 char. name of fg 366 ?> = fx30
 Enter new 6 char. name of fg 367 ?> = fx31
 Enter new 6 char. name of fg 368 ?> = fx32
 Enter new 6 char. name of fg 369 ?> = fx33
 Enter new 6 char. name of fg 370 ?> = fx34
 Enter new 6 char. name of fg 371 ?> = fx35
 Enter new 6 char. name of fg 372 ?> = fx36
 Enter new 6 char. name of fg 373 ?> = fx37
 Enter new 6 char. name of fg 374 ?> = fx38
 Enter new 6 char. name of fg 375 ?> = fx39
 Enter new 6 char. name of fg 376 ?> = fx40
 Enter new 6 char. name of fg 377 ?> = fx41
 Enter new 6 char. name of fg 378 ?> = fx42
 Enter new 6 char. name of fg 379 ?> = fx43
 Enter new 6 char. name of fg 380 ?> = fx44
 Enter new 6 char. name of fg 381 ?> = fx45
 Enter new 6 char. name of fg 382 ?> = fx46
 Enter new 6 char. name of fg 383 ?> = fx47
 Enter new 6 char. name of fg 384 ?> = fx48
 Enter new 6 char. name of fg 385 ?> = fx49
 Enter new 6 char. name of fg 386 ?> = fx50
 Enter new 6 char. name of fg 387 ?> = fx51
 Enter new 6 char. name of fg 388 ?> = fx52
 Enter new 6 char. name of fg 389 ?> = fx53
 Enter new 6 char. name of fg 390 ?> = fx54
 Enter new 6 char. name of fg 391 ?> = fx55
 Enter new 6 char. name of fg 392 ?> = fx56
 Enter new 6 char. name of fg 393 ?> = fx57
 Enter new 6 char. name of fg 394 ?> = fx58

Enter new 6 char. name of fg 395 ?> = fx59
Enter new 6 char. name of fg 396 ?> = fx60
Enter new 6 char. name of fg 397 ?> = fx61
Enter new 6 char. name of fg 398 ?> = fx62
Enter new 6 char. name of fg 399 ?> = fx63
Enter new 6 char. name of fg 400 ?> = fx64
Enter new 6 char. name of fg 401 ?> = fx65
Enter new 6 char. name of fg 402 ?> = fx66
Enter new 6 char. name of fg 403 ?> = fx67
Enter new 6 char. name of fg 404 ?> = fx68
Enter new 6 char. name of fg 405 ?> = fx69
Enter new 6 char. name of fg 406 ?> = fx70
Enter new 6 char. name of fg 407 ?> = fx71
Enter new 6 char. name of fg 408 ?> = fx72
Enter new 6 char. name of fg 409 ?> = fx73
Enter new 6 char. name of fg 410 ?> = fx74
Enter new 6 char. name of fg 411 ?> = fx75
Enter new 6 char. name of fg 412 ?> = fx76
Enter new 6 char. name of fg 413 ?> = fx77
Enter new 6 char. name of fg 414 ?> = fx78
Enter new 6 char. name of fg 415 ?> = fx79
Enter new 6 char. name of fg 416 ?> = fx80
Enter new 6 char. name of fg 417 ?> = fx81
Enter new 6 char. name of fg 418 ?> = fx82
Enter new 6 char. name of fg 419 ?> = fx83
Enter new 6 char. name of fg 420 ?> = fx84
Enter new 6 char. name of fg 421 ?> = fx85
Enter new 6 char. name of fg 422 ?> = fx86
Enter new 6 char. name of fg 423 ?> = fx87
Enter new 6 char. name of fg 424 ?> = fx88
Enter new 6 char. name of fg 425 ?> = fx89
Enter new 6 char. name of fg 426 ?> = fx90
Enter new 6 char. name of fg 427 ?> = fx91
Enter new 6 char. name of fg 428 ?> = fx92
Enter new 6 char. name of fg 429 ?> = fx93
Enter new 6 char. name of fg 430 ?> = fx94
Enter new 6 char. name of fg 431 ?> = fx95
Enter new 6 char. name of fg 432 ?> = fx96
Enter new 6 char. name of fg 433 ?> = fx97
Enter new 6 char. name of fg 434 ?> = fx98
Enter new 6 char. name of fg 435 ?> = fx99
Enter new 6 char. name of fg 436 ?> = fx100
Enter new 6 char. name of fg 437 ?> = fx101
Enter new 6 char. name of fg 438 ?> = fx102
Enter new 6 char. name of fg 439 ?> = fx103
Enter new 6 char. name of fg 440 ?> = fx104
Enter new 6 char. name of fg 441 ?> = fx105
Enter new 6 char. name of fg 442 ?> = fx106
Enter new 6 char. name of fg 443 ?> = fx107
Enter new 6 char. name of fg 444 ?> = fx108

Enter new 6 char. name of fg 445 ?> = fx109
 Enter new 6 char. name of fg 446 ?> = fx110
 Enter new 6 char. name of fg 447 ?> = fleak0
 Enter new 6 char. name of fg 448 ?> = fleak1
 Enter new 6 char. name of fg 449 ?> = fleak2
 Enter new 6 char. name of fg 450 ?> = fleak3
 Enter new 6 char. name of fg 451 ?> = fleak4
 Enter new 6 char. name of fg 452 ?> = fleak5
 Enter new 6 char. name of fg 453 ?> = fleak6
 Enter new 6 char. name of fg 454 ?> = fleak7
 Enter new 6 char. name of fg 455 ?> = fsn0
 Enter new 6 char. name of fg 456 ?> = fsn1
 Enter new 6 char. name of fg 457 ?> = fsn2
 Enter new 6 char. name of fg 458 ?> = fsn3
 Enter new 6 char. name of fg 459 ?> = fsn4
 Enter new 6 char. name of fg 460 ?> = fsn5
 Enter new 6 char. name of fg 461 ?> = fsn6
 Enter new 6 char. name of fg 462 ?> = fsn7
 Enter new 6 char. name of fg 463 ?> = fs10
 Enter new 6 char. name of fg 464 ?> = fs11
 Enter new 6 char. name of fg 465 ?> = fs12
 Enter new 6 char. name of fg 466 ?> = fs13
 Enter new 6 char. name of fg 467 ?> = fs14
 Enter new 6 char. name of fg 468 ?> = fs15
 Enter new 6 char. name of fg 469 ?> = fs16
 Enter new 6 char. name of fg 470 ?> = fs17
 Enter new 6 char. name of fg 471 ?> = ffarm
 Enter new 6 char. name of fg 472 ?> = fspit
 Enter new 6 char. name of fg 473 ?> = fvlpit
 Enter new 6 char. name of fg 474 ?> = fhvac
 Enter new 6 char. name of fg 475 ?> = f271
 Enter new 6 char. name of fg 476 ?> = fflush
 Enter new 6 char. name of fg 477 ?> = fleak
 Enter new 6 char. name of fg 478 ?> = fdisp
 Enter new 6 char. name of fg 479 ?> = fhold
 Enter new 6 char. name of fg 480 ?> = fning
 Enter new 6 char. name of fg 481 ?> = fabort
 Enter new 6 char. name of fg 482 ?> = fstart
 Enter new 6 char. name of fg 483 ?> = fldalm
 Enter new 6 char. name of fg 484 ?> = fpass
 Enter new 6 char. name of fg 485 ?> = ftrip0
 Enter new 6 char. name of fg 486 ?> = ftrip1
 Enter new 6 char. name of fg 487 ?> = ftrip2
 Enter new 6 char. name of fg 488 ?> = ftrip3
 Enter new 6 char. name of fg 489 ?> = ftrip4
 Enter new 6 char. name of fg 490 ?> = reset
 Enter new 6 char. name of fg 491 ?> = ack
 Enter new 6 char. name of fg 492 ?> = falmon
 Enter new 6 char. name of fg 493 ?> = fack
 Enter new 6 char. name of fg 494 ?> = fclear

```

Enter new 6 char. name of fg 495 ?> = flgack
MUPF?> = add 418 in
Enter new 6 char. name of in 0 ?> = bitval
Enter new 6 char. name of in 1 ?> = lmpval
Enter new 6 char. name of in 2 ?> = iloop
Enter new 6 char. name of in 3 ?> = jloop
Enter new 6 char. name of in 4 ?> = kloop
Enter new 6 char. name of in 5 ?> = lloop
Enter new 6 char. name of in 6 ?> = mloop
Enter new 6 char. name of in 7 ?> = index
Enter new 6 char. name of in 8 ?> = last
Enter new 6 char. name of in 9 ?> = curent
Enter new 6 char. name of in 10 ?> = nxt
Enter new 6 char. name of in 11 ?> = analog
Enter new 6 char. name of in 12 ?> = white
Enter new 6 char. name of in 13 ?> = stat
Enter new 6 char. name of in 14 ?> = remain
Enter new 6 char. name of in 15 ?> = trip
Enter new 6 char. name of in 16 ?> = star
Enter new 6 char. name of in 17 ?> = skdkey
Enter new 6 char. name of in 18 ?> = unitt
Enter new 6 char. name of in 19 ?> = adi
Enter new 6 char. name of in 20 ?> = nadi
Enter new 6 char. name of in 21 ?> = blkval
Enter new 6 char. name of in 22 ?> = sort
Enter new 6 char. name of in 23 ?> = bval
Enter new 6 char. name of in 24 ?> = hldstp
Enter new 6 char. name of in 25 ?> = errn
Enter new 6 char. name of in 26 ?> = errl
Enter new 6 char. name of in 27 ?> = int0
Enter new 6 char. name of in 28 ?> = int1
Enter new 6 char. name of in 29 ?> = int2
Enter new 6 char. name of in 30 ?> = int3
Enter new 6 char. name of in 31 ?> = int4
Enter new 6 char. name of in 32 ?> = int5
Enter new 6 char. name of in 33 ?> = int6
Enter new 6 char. name of in 34 ?> = int7
Enter new 6 char. name of in 35 ?> = int8
Enter new 6 char. name of in 36 ?> = int9
Enter new 6 char. name of in 37 ?> = int10
Enter new 6 char. name of in 38 ?> = int11
Enter new 6 char. name of in 39 ?> = int12
Enter new 6 char. name of in 40 ?> = int13
Enter new 6 char. name of in 41 ?> = int14
Enter new 6 char. name of in 42 ?> = int15
Enter new 6 char. name of in 43 ?> = int16
Enter new 6 char. name of in 44 ?> = int17
Enter new 6 char. name of in 45 ?> = int18
Enter new 6 char. name of in 46 ?> = int19
Enter new 6 char. name of in 47 ?> = int20

```

Enter new 6 char. name of in 48 ?> = int21
Enter new 6 char. name of in 49 ?> = int22
Enter new 6 char. name of in 50 ?> = int23
Enter new 6 char. name of in 51 ?> = int24
Enter new 6 char. name of in 52 ?> = int25
Enter new 6 char. name of in 53 ?> = int26
Enter new 6 char. name of in 54 ?> = int27
Enter new 6 char. name of in 55 ?> = int28
Enter new 6 char. name of in 56 ?> = int29
Enter new 6 char. name of in 57 ?> = int30
Enter new 6 char. name of in 58 ?> = int31
Enter new 6 char. name of in 59 ?> = int32
Enter new 6 char. name of in 60 ?> = int33
Enter new 6 char. name of in 61 ?> = int34
Enter new 6 char. name of in 62 ?> = int35
Enter new 6 char. name of in 63 ?> = int36
Enter new 6 char. name of in 64 ?> = int37
Enter new 6 char. name of in 65 ?> = int38
Enter new 6 char. name of in 66 ?> = int39
Enter new 6 char. name of in 67 ?> = int40
Enter new 6 char. name of in 68 ?> = int41
Enter new 6 char. name of in 69 ?> = int42
Enter new 6 char. name of in 70 ?> = int43
Enter new 6 char. name of in 71 ?> = int44
Enter new 6 char. name of in 72 ?> = int45
Enter new 6 char. name of in 73 ?> = int46
Enter new 6 char. name of in 74 ?> = int47
Enter new 6 char. name of in 75 ?> = int48
Enter new 6 char. name of in 76 ?> = int49
Enter new 6 char. name of in 77 ?> = int50
Enter new 6 char. name of in 78 ?> = int51
Enter new 6 char. name of in 79 ?> = int52
Enter new 6 char. name of in 80 ?> = int53
Enter new 6 char. name of in 81 ?> = int54
Enter new 6 char. name of in 82 ?> = int55
Enter new 6 char. name of in 83 ?> = int56
Enter new 6 char. name of in 84 ?> = int57
Enter new 6 char. name of in 85 ?> = int58
Enter new 6 char. name of in 86 ?> = int59
Enter new 6 char. name of in 87 ?> = int60
Enter new 6 char. name of in 88 ?> = int61
Enter new 6 char. name of in 89 ?> = int62
Enter new 6 char. name of in 90 ?> = int63
Enter new 6 char. name of in 91 ?> = int64
Enter new 6 char. name of in 92 ?> = int65
Enter new 6 char. name of in 93 ?> = int66
Enter new 6 char. name of in 94 ?> = int67
Enter new 6 char. name of in 95 ?> = int68
Enter new 6 char. name of in 96 ?> = int69
Enter new 6 char. name of in 97 ?> = int70

Enter new 6 char. name of in 98 ?> = int71
Enter new 6 char. name of in 99 ?> = int72
Enter new 6 char. name of in 100 ?> = int73
Enter new 6 char. name of in 101 ?> = int74
Enter new 6 char. name of in 102 ?> = int75
Enter new 6 char. name of in 103 ?> = int76
Enter new 6 char. name of in 104 ?> = int77
Enter new 6 char. name of in 105 ?> = int78
Enter new 6 char. name of in 106 ?> = int79
Enter new 6 char. name of in 107 ?> = int80
Enter new 6 char. name of in 108 ?> = int81
Enter new 6 char. name of in 109 ?> = int82
Enter new 6 char. name of in 110 ?> = int83
Enter new 6 char. name of in 111 ?> = int84
Enter new 6 char. name of in 112 ?> = int85
Enter new 6 char. name of in 113 ?> = int86
Enter new 6 char. name of in 114 ?> = int87
Enter new 6 char. name of in 115 ?> = int88
Enter new 6 char. name of in 116 ?> = int89
Enter new 6 char. name of in 117 ?> = int90
Enter new 6 char. name of in 118 ?> = int91
Enter new 6 char. name of in 119 ?> = int92
Enter new 6 char. name of in 120 ?> = int93
Enter new 6 char. name of in 121 ?> = int94
Enter new 6 char. name of in 122 ?> = int95
Enter new 6 char. name of in 123 ?> = int96
Enter new 6 char. name of in 124 ?> = int97
Enter new 6 char. name of in 125 ?> = int98
Enter new 6 char. name of in 126 ?> = int99
Enter new 6 char. name of in 127 ?> = ialm0
Enter new 6 char. name of in 128 ?> = ialm1
Enter new 6 char. name of in 129 ?> = ialm2
Enter new 6 char. name of in 130 ?> = ialm3
Enter new 6 char. name of in 131 ?> = ialm4
Enter new 6 char. name of in 132 ?> = ialm5
Enter new 6 char. name of in 133 ?> = ialm6
Enter new 6 char. name of in 134 ?> = ialm7
Enter new 6 char. name of in 135 ?> = ialm8
Enter new 6 char. name of in 136 ?> = ialm9
Enter new 6 char. name of in 137 ?> = ialm10
Enter new 6 char. name of in 138 ?> = ialm11
Enter new 6 char. name of in 139 ?> = ialm12
Enter new 6 char. name of in 140 ?> = ialm13
Enter new 6 char. name of in 141 ?> = ialm14
Enter new 6 char. name of in 142 ?> = ialm15
Enter new 6 char. name of in 143 ?> = ialm16
Enter new 6 char. name of in 144 ?> = ialm17
Enter new 6 char. name of in 145 ?> = ialm18
Enter new 6 char. name of in 146 ?> = ialm19
Enter new 6 char. name of in 147 ?> = ialm20

Enter new 6 char. name of in 148 ?> = ialm21
Enter new 6 char. name of in 149 ?> = ialm22
Enter new 6 char. name of in 150 ?> = ialm23
Enter new 6 char. name of in 151 ?> = ialm24
Enter new 6 char. name of in 152 ?> = ialm25
Enter new 6 char. name of in 153 ?> = ialm26
Enter new 6 char. name of in 154 ?> = ialm27
Enter new 6 char. name of in 155 ?> = ialm28
Enter new 6 char. name of in 156 ?> = ialm29
Enter new 6 char. name of in 157 ?> = ialm30
Enter new 6 char. name of in 158 ?> = ialm31
Enter new 6 char. name of in 159 ?> = ialm32
Enter new 6 char. name of in 160 ?> = ialm33
Enter new 6 char. name of in 161 ?> = ialm34
Enter new 6 char. name of in 162 ?> = ialm35
Enter new 6 char. name of in 163 ?> = ialm36
Enter new 6 char. name of in 164 ?> = ialm37
Enter new 6 char. name of in 165 ?> = ialm38
Enter new 6 char. name of in 166 ?> = ialm39
Enter new 6 char. name of in 167 ?> = ialm40
Enter new 6 char. name of in 168 ?> = ialm41
Enter new 6 char. name of in 169 ?> = ialm42
Enter new 6 char. name of in 170 ?> = ialm43
Enter new 6 char. name of in 171 ?> = ialm44
Enter new 6 char. name of in 172 ?> = ialm45
Enter new 6 char. name of in 173 ?> = ialm46
Enter new 6 char. name of in 174 ?> = ialm47
Enter new 6 char. name of in 175 ?> = ialm48
Enter new 6 char. name of in 176 ?> = ialm49
Enter new 6 char. name of in 177 ?> = ialm50
Enter new 6 char. name of in 178 ?> = ialm51
Enter new 6 char. name of in 179 ?> = ialm52
Enter new 6 char. name of in 180 ?> = ialm53
Enter new 6 char. name of in 181 ?> = ialm54
Enter new 6 char. name of in 182 ?> = ialm55
Enter new 6 char. name of in 183 ?> = ialm56
Enter new 6 char. name of in 184 ?> = ialm57
Enter new 6 char. name of in 185 ?> = ialm58
Enter new 6 char. name of in 186 ?> = ialm59
Enter new 6 char. name of in 187 ?> = ialm60
Enter new 6 char. name of in 188 ?> = ialm61
Enter new 6 char. name of in 189 ?> = ialm62
Enter new 6 char. name of in 190 ?> = ialm63
Enter new 6 char. name of in 191 ?> = ialm64
Enter new 6 char. name of in 192 ?> = ialm65
Enter new 6 char. name of in 193 ?> = ialm66
Enter new 6 char. name of in 194 ?> = ialm67
Enter new 6 char. name of in 195 ?> = ialm68
Enter new 6 char. name of in 196 ?> = ialm69
Enter new 6 char. name of in 197 ?> = ialm70

Enter new 6 char. name of in 198 ?> = ialm71
Enter new 6 char. name of in 199 ?> = ialm72
Enter new 6 char. name of in 200 ?> = ialm73
Enter new 6 char. name of in 201 ?> = ialm74
Enter new 6 char. name of in 202 ?> = ialm75
Enter new 6 char. name of in 203 ?> = ialm76
Enter new 6 char. name of in 204 ?> = ialm77
Enter new 6 char. name of in 205 ?> = ialm78
Enter new 6 char. name of in 206 ?> = ialm79
Enter new 6 char. name of in 207 ?> = ialm80
Enter new 6 char. name of in 208 ?> = ialm81
Enter new 6 char. name of in 209 ?> = ialm82
Enter new 6 char. name of in 210 ?> = ialm83
Enter new 6 char. name of in 211 ?> = ialm84
Enter new 6 char. name of in 212 ?> = ialm85
Enter new 6 char. name of in 213 ?> = ialm86
Enter new 6 char. name of in 214 ?> = ialm87
Enter new 6 char. name of in 215 ?> = ialm88
Enter new 6 char. name of in 216 ?> = ialm89
Enter new 6 char. name of in 217 ?> = ialm90
Enter new 6 char. name of in 218 ?> = ialm91
Enter new 6 char. name of in 219 ?> = ialm92
Enter new 6 char. name of in 220 ?> = ialm93
Enter new 6 char. name of in 221 ?> = ialm94
Enter new 6 char. name of in 222 ?> = ialm95
Enter new 6 char. name of in 223 ?> = ialm96
Enter new 6 char. name of in 224 ?> = ialm97
Enter new 6 char. name of in 225 ?> = ialm98
Enter new 6 char. name of in 226 ?> = ialm99
Enter new 6 char. name of in 227 ?> = skid0
Enter new 6 char. name of in 228 ?> = skid1
Enter new 6 char. name of in 229 ?> = skid2
Enter new 6 char. name of in 230 ?> = skid3
Enter new 6 char. name of in 231 ?> = skid4
Enter new 6 char. name of in 232 ?> = skid5
Enter new 6 char. name of in 233 ?> = skid6
Enter new 6 char. name of in 234 ?> = skid7
Enter new 6 char. name of in 235 ?> = skid8
Enter new 6 char. name of in 236 ?> = skid9
Enter new 6 char. name of in 237 ?> = skid10
Enter new 6 char. name of in 238 ?> = skid11
Enter new 6 char. name of in 239 ?> = skid12
Enter new 6 char. name of in 240 ?> = skid13
Enter new 6 char. name of in 241 ?> = skid14
Enter new 6 char. name of in 242 ?> = skid15
Enter new 6 char. name of in 243 ?> = skid16
Enter new 6 char. name of in 244 ?> = skid17
Enter new 6 char. name of in 245 ?> = skid18
Enter new 6 char. name of in 246 ?> = skid19
Enter new 6 char. name of in 247 ?> = skid20

Enter new 6 char. name of in 248 ?> = skid21
Enter new 6 char. name of in 249 ?> = skid22
Enter new 6 char. name of in 250 ?> = skid23
Enter new 6 char. name of in 251 ?> = skid24
Enter new 6 char. name of in 252 ?> = skid25
Enter new 6 char. name of in 253 ?> = skid26
Enter new 6 char. name of in 254 ?> = skid27
Enter new 6 char. name of in 255 ?> = skid28
Enter new 6 char. name of in 256 ?> = skid29
Enter new 6 char. name of in 257 ?> = skid30
Enter new 6 char. name of in 258 ?> = skid31
Enter new 6 char. name of in 259 ?> = skid32
Enter new 6 char. name of in 260 ?> = skid33
Enter new 6 char. name of in 261 ?> = skid34
Enter new 6 char. name of in 262 ?> = skid35
Enter new 6 char. name of in 263 ?> = skid36
Enter new 6 char. name of in 264 ?> = skid37
Enter new 6 char. name of in 265 ?> = skid38
Enter new 6 char. name of in 266 ?> = skid39
Enter new 6 char. name of in 267 ?> = skid40
Enter new 6 char. name of in 268 ?> = skid41
Enter new 6 char. name of in 269 ?> = skid42
Enter new 6 char. name of in 270 ?> = skid43
Enter new 6 char. name of in 271 ?> = skid44
Enter new 6 char. name of in 272 ?> = skid45
Enter new 6 char. name of in 273 ?> = skid46
Enter new 6 char. name of in 274 ?> = skid47
Enter new 6 char. name of in 275 ?> = skid48
Enter new 6 char. name of in 276 ?> = skid49
Enter new 6 char. name of in 277 ?> = skid50
Enter new 6 char. name of in 278 ?> = skid51
Enter new 6 char. name of in 279 ?> = skid52
Enter new 6 char. name of in 280 ?> = skid53
Enter new 6 char. name of in 281 ?> = skid54
Enter new 6 char. name of in 282 ?> = skid55
Enter new 6 char. name of in 283 ?> = skid56
Enter new 6 char. name of in 284 ?> = skid57
Enter new 6 char. name of in 285 ?> = skid58
Enter new 6 char. name of in 286 ?> = skid59
Enter new 6 char. name of in 287 ?> = skid60
Enter new 6 char. name of in 288 ?> = skid61
Enter new 6 char. name of in 289 ?> = skid62
Enter new 6 char. name of in 290 ?> = skid63
Enter new 6 char. name of in 291 ?> = skid64
Enter new 6 char. name of in 292 ?> = skid65
Enter new 6 char. name of in 293 ?> = skid66
Enter new 6 char. name of in 294 ?> = skid67
Enter new 6 char. name of in 295 ?> = skid68
Enter new 6 char. name of in 296 ?> = skid69
Enter new 6 char. name of in 297 ?> = skid70

Enter new 6 char. name of in 298 ?> = skid71
Enter new 6 char. name of in 299 ?> = skid72
Enter new 6 char. name of in 300 ?> = skid73
Enter new 6 char. name of in 301 ?> = skid74
Enter new 6 char. name of in 302 ?> = skid75
Enter new 6 char. name of in 303 ?> = skid76
Enter new 6 char. name of in 304 ?> = skid77
Enter new 6 char. name of in 305 ?> = skid78
Enter new 6 char. name of in 306 ?> = skid79
Enter new 6 char. name of in 307 ?> = skid80
Enter new 6 char. name of in 308 ?> = skid81
Enter new 6 char. name of in 309 ?> = skid82
Enter new 6 char. name of in 310 ?> = skid83
Enter new 6 char. name of in 311 ?> = skid84
Enter new 6 char. name of in 312 ?> = skid85
Enter new 6 char. name of in 313 ?> = skid86
Enter new 6 char. name of in 314 ?> = skid87
Enter new 6 char. name of in 315 ?> = skid88
Enter new 6 char. name of in 316 ?> = skid89
Enter new 6 char. name of in 317 ?> = skid90
Enter new 6 char. name of in 318 ?> = skid91
Enter new 6 char. name of in 319 ?> = skid92
Enter new 6 char. name of in 320 ?> = skid93
Enter new 6 char. name of in 321 ?> = skid94
Enter new 6 char. name of in 322 ?> = skid95
Enter new 6 char. name of in 323 ?> = skid96
Enter new 6 char. name of in 324 ?> = skid97
Enter new 6 char. name of in 325 ?> = skid98
Enter new 6 char. name of in 326 ?> = skid99
Enter new 6 char. name of in 327 ?> = dev0
Enter new 6 char. name of in 328 ?> = dev1
Enter new 6 char. name of in 329 ?> = dev2
Enter new 6 char. name of in 330 ?> = dev3
Enter new 6 char. name of in 331 ?> = dev4
Enter new 6 char. name of in 332 ?> = dev5
Enter new 6 char. name of in 333 ?> = dev6
Enter new 6 char. name of in 334 ?> = dev7
Enter new 6 char. name of in 335 ?> = dev8
Enter new 6 char. name of in 336 ?> = dev9
Enter new 6 char. name of in 337 ?> = dev10
Enter new 6 char. name of in 338 ?> = dev11
Enter new 6 char. name of in 339 ?> = dev12
Enter new 6 char. name of in 340 ?> = dev13
Enter new 6 char. name of in 341 ?> = dev14
Enter new 6 char. name of in 342 ?> = dev15
Enter new 6 char. name of in 343 ?> = dev16
Enter new 6 char. name of in 344 ?> = dev17
Enter new 6 char. name of in 345 ?> = dev18
Enter new 6 char. name of in 346 ?> = dev19
Enter new 6 char. name of in 347 ?> = farm

WHC-SD-534-CSWL-001 REV 3

```

Enter new 6 char. name of in 348 ?> = serpit
Enter new 6 char. name of in 349 ?> = vlvpit
Enter new 6 char. name of in 350 ?> = hvac
Enter new 6 char. name of in 351 ?> = bld271
Enter new 6 char. name of in 352 ?> = flspit
Enter new 6 char. name of in 353 ?> = lekpit
Enter new 6 char. name of in 354 ?> = drapit
Enter new 6 char. name of in 355 ?> = wspit
Enter new 6 char. name of in 356 ?> = yspit
Enter new 6 char. name of in 357 ?> = rspit
Enter new 6 char. name of in 358 ?> = wvlvpt
Enter new 6 char. name of in 359 ?> = yvlvpt
Enter new 6 char. name of in 360 ?> = rvlvpt
Enter new 6 char. name of in 361 ?> = whvac
Enter new 6 char. name of in 362 ?> = yhvac
Enter new 6 char. name of in 363 ?> = rhvac
Enter new 6 char. name of in 364 ?> = wbld
Enter new 6 char. name of in 365 ?> = ybld
Enter new 6 char. name of in 366 ?> = rbld
Enter new 6 char. name of in 367 ?> = wpump
Enter new 6 char. name of in 368 ?> = ypump
Enter new 6 char. name of in 369 ?> = rpump
Enter new 6 char. name of in 370 ?> = leak0
Enter new 6 char. name of in 371 ?> = leak1
Enter new 6 char. name of in 372 ?> = leak2
Enter new 6 char. name of in 373 ?> = leak3
Enter new 6 char. name of in 374 ?> = leak4
Enter new 6 char. name of in 375 ?> = leak5
Enter new 6 char. name of in 376 ?> = hour
Enter new 6 char. name of in 377 ?> = min
Enter new 6 char. name of in 378 ?> = sec
Enter new 6 char. name of in 379 ?> = dom
Enter new 6 char. name of in 380 ?> = mon
Enter new 6 char. name of in 381 ?> = year
Enter new 6 char. name of in 382 ?> = alm0
Enter new 6 char. name of in 383 ?> = alm1
Enter new 6 char. name of in 384 ?> = alm2
Enter new 6 char. name of in 385 ?> = alm3
Enter new 6 char. name of in 386 ?> = alm4
Enter new 6 char. name of in 387 ?> = alm5
Enter new 6 char. name of in 388 ?> = alm6
Enter new 6 char. name of in 389 ?> = alm7
Enter new 6 char. name of in 390 ?> = alm8
Enter new 6 char. name of in 391 ?> = alm9
Enter new 6 char. name of in 392 ?> = ack0
Enter new 6 char. name of in 393 ?> = ack1
Enter new 6 char. name of in 394 ?> = ack2
Enter new 6 char. name of in 395 ?> = ack3
Enter new 6 char. name of in 396 ?> = ack4
Enter new 6 char. name of in 397 ?> = ack5

```

Enter new 6 char. name of in 398 ?> = ack6
 Enter new 6 char. name of in 399 ?> = ack7
 Enter new 6 char. name of in 400 ?> = ack8
 Enter new 6 char. name of in 401 ?> = ack9
 Enter new 6 char. name of in 402 ?> = color0
 Enter new 6 char. name of in 403 ?> = color1
 Enter new 6 char. name of in 404 ?> = color2
 Enter new 6 char. name of in 405 ?> = color3
 Enter new 6 char. name of in 406 ?> = color4
 Enter new 6 char. name of in 407 ?> = color5
 Enter new 6 char. name of in 408 ?> = color6
 Enter new 6 char. name of in 409 ?> = color7
 Enter new 6 char. name of in 410 ?> = grafx0
 Enter new 6 char. name of in 411 ?> = grafx1
 Enter new 6 char. name of in 412 ?> = grafx2
 Enter new 6 char. name of in 413 ?> = grafx3
 Enter new 6 char. name of in 414 ?> = grafx4
 Enter new 6 char. name of in 415 ?> = grafx5
 Enter new 6 char. name of in 416 ?> = grafx6
 Enter new 6 char. name of in 417 ?> = grafx7
 MUPF?> = add 20 fl
 Enter new 6 char. name of fl 0 ?> = ai0
 Enter new 6 char. name of fl 1 ?> = ai1
 Enter new 6 char. name of fl 2 ?> = ai2
 Enter new 6 char. name of fl 3 ?> = ai3
 Enter new 6 char. name of fl 4 ?> = ai4
 Enter new 6 char. name of fl 5 ?> = ai5
 Enter new 6 char. name of fl 6 ?> = ai6
 Enter new 6 char. name of fl 7 ?> = ai7
 Enter new 6 char. name of fl 8 ?> = ai8
 Enter new 6 char. name of fl 9 ?> = ai9
 Enter new 6 char. name of fl 10 ?> = ai10
 Enter new 6 char. name of fl 11 ?> = ai11
 Enter new 6 char. name of fl 12 ?> = ai12
 Enter new 6 char. name of fl 13 ?> = ai13
 Enter new 6 char. name of fl 14 ?> = ai14
 Enter new 6 char. name of fl 15 ?> = ai15
 Enter new 6 char. name of fl 16 ?> = ai16
 Enter new 6 char. name of fl 17 ?> = ai17
 Enter new 6 char. name of fl 18 ?> = ai18
 Enter new 6 char. name of fl 19 ?> = ai19
 MUPF?> = add 13 pt
 Enter new 6 char. name of pt 0 ?> = pepn
 Enter new 6 char. name of pt 1 ?> = pdin
 Enter new 6 char. name of pt 2 ?> = pdev
 Enter new 6 char. name of pt 3 ?> = pai
 Enter new 6 char. name of pt 4 ?> = pskid
 Enter new 6 char. name of pt 5 ?> = pstat
 Enter new 6 char. name of pt 6 ?> = ptype
 Enter new 6 char. name of pt 7 ?> = prev

```

Enter new 6 char. name of pt 8 ?> = psn
Enter new 6 char. name of pt 9 ?> = pmsg0
Enter new 6 char. name of pt 10 ?> = pmsg1
Enter new 6 char. name of pt 11 ?> = ppass
Enter new 6 char. name of pt 12 ?> = pack
MUPF?> = add 61 sg
Enter new 6 char. name of sg 0 ?> = swhen0
Enter max string length for sg 0 ?> = 8
Enter new 6 char. name of sg 1Enter new 6 char. name of sg 1 ?> = swhen1
Enter max string length for sg 1 ?> = 8
Enter new 6 char. name of sg 2 ?> = swhen2
Enter max string length for sg 2 ?> = 8
Enter new 6 char. name of sg 3 ?> = swhen3
Enter max string length for sg 3 ?> = 8
Enter new 6 char. name of sg 4 ?> = swhen4
Enter max string length for sg 4 ?> = 8
Enter new 6 char. name of sg 5 ?> = swhen5
Enter max string length for sg 5 ?> = 8
Enter new 6 char. name of sg 6 ?> = swhen6
Enter max string length for sg 6 ?> = 8
Enter new 6 char. name of sg 7 ?> = swhen7
Enter max string length for sg 7 ?> = 8
Enter new 6 char. name of sg 8 ?> = ssng0
Enter max string length for sg 8 ?> = 8
Enter new 6 char. name of sg 9 ?> = ssng1
Enter max string length for sg 9 ?> = 8
Enter new 6 char. name of sg 10 ?> = ssng2
Enter max string length for sg 10 ?> = 8
Enter new 6 char. name of sg 11 ?> = ssng3
Enter max string length for sg 11 ?> = 8
Enter new 6 char. name of sg 12 ?> = ssng4
Enter max string length for sg 12 ?> = 8
Enter new 6 char. name of sg 13 ?> = ssng5
Enter max string length for sg 13 ?> = 8
Enter new 6 char. name of sg 14 ?> = ssng6
Enter max string length for sg 14 ?> = 8
Enter new 6 char. name of sg 15 ?> = ssng7
Enter max string length for sg 15 ?> = 8
Enter new 6 char. name of sg 16 ?> = sdate0
Enter max string length for sg 16 ?> = 10
Enter new 6 char. name of sg 17 ?> = sdate1
Enter max string length for sg 17 ?> = 10
Enter new 6 char. name of sg 18 ?> = sdate2
Enter max string length for sg 18 ?> = 10
Enter new 6 char. name of sg 19 ?> = sdate3
Enter max string length for sg 19 ?> = 10
Enter new 6 char. name of sg 20 ?> = sdate4
Enter max string length for sg 20 ?> = 10
Enter new 6 char. name of sg 21 ?> = sdate5
Enter max string length for sg 21 ?> = 10

```

Enter new 6 char. name of sg 22 ?> = sdate6
Enter max string length for sg 22 ?> = 10
Enter new 6 char. name of sg 23 ?> = sdate7
Enter max string length for sg 23 ?> = 10
Enter new 6 char. name of sg 24 ?> = epns0
Enter max string length for sg 24 ?> = 10
Enter new 6 char. name of sg 25 ?> = epns1
Enter max string length for sg 25 ?> = 10
Enter new 6 char. name of sg 26 ?> = epns2
Enter max string length for sg 26 ?> = 10
Enter new 6 char. name of sg 27 ?> = epns3
Enter max string length for sg 27 ?> = 10
Enter new 6 char. name of sg 28 ?> = epns4
Enter max string length for sg 28 ?> = 10
Enter new 6 char. name of sg 29 ?> = epns5
Enter max string length for sg 29 ?> = 10
Enter new 6 char. name of sg 30 ?> = epns6
Enter max string length for sg 30 ?> = 10
Enter new 6 char. name of sg 31 ?> = epns7
Enter max string length for sg 31 ?> = 10
Enter new 6 char. name of sg 32 ?> = salm0
Enter max string length for sg 32 ?> = 20
Enter new 6 char. name of sg 33 ?> = salm1
Enter max string length for sg 33 ?> = 20
Enter new 6 char. name of sg 34 ?> = salm2
Enter max string length for sg 34 ?> = 20
Enter new 6 char. name of sg 35 ?> = salm3
Enter max string length for sg 35 ?> = 20
Enter new 6 char. name of sg 36 ?> = salm4
Enter max string length for sg 36 ?> = 20
Enter new 6 char. name of sg 37 ?> = salm5
Enter max string length for sg 37 ?> = 20
Enter new 6 char. name of sg 38 ?> = salm6
Enter max string length for sg 38 ?> = 20
Enter new 6 char. name of sg 39 ?> = salm7
Enter max string length for sg 39 ?> = 20
Enter new 6 char. name of sg 40 ?> = msg0
Enter max string length for sg 40 ?> = 30
Enter new 6 char. name of sg 41 ?> = msg1
Enter max string length for sg 41 ?> = 30
Enter new 6 char. name of sg 42 ?> = msg2
Enter max string length for sg 42 ?> = 30
Enter new 6 char. name of sg 43 ?> = msg3
Enter max string length for sg 43 ?> = 30
Enter new 6 char. name of sg 44 ?> = msg4
Enter max string length for sg 44 ?> = 30
Enter new 6 char. name of sg 45 ?> = msg5
Enter max string length for sg 45 ?> = 30
Enter new 6 char. name of sg 46 ?> = msg6
Enter max string length for sg 46 ?> = 30


```

Enter new 6 char. name of sg 47 ?> = msg7
Enter max string length for sg 47 ?> = 30
Enter new 6 char. name of sg 48 ?> = lsg0
Enter max string length for sg 48 ?> = 30
Enter new 6 char. name of sg 49 ?> = lsg1
Enter max string length for sg 49 ?> = 30
Enter new 6 char. name of sg 50 ?> = lsg2
Enter max string length for sg 50 ?> = 30
Enter new 6 char. name of sg 51 ?> = lsg3
Enter max string length for sg 51 ?> = 30
Enter new 6 char. name of sg 52 ?> = lsg4
Enter max string length for sg 52 ?> = 30
Enter new 6 char. name of sg 53 ?> = lsg5
Enter max string length for sg 53 ?> = 30
Enter new 6 char. name of sg 54 ?> = lsg6
Enter max string length for sg 54 ?> = 30
Enter new 6 char. name of sg 55 ?> = lsg7
Enter max string length for sg 55 ?> = 30
Enter new 6 char. name of sg 56 ?> = strng0
Enter max string length for sg 56 ?> = 30
Enter new 6 char. name of sg 57 ?> = strng1
Enter max string length for sg 57 ?> = 30
Enter new 6 char. name of sg 58 ?> = strng2
Enter max string length for sg 58 ?> = 30
Enter new 6 char. name of sg 59 ?> = strng3
Enter max string length for sg 59 ?> = 30
Enter new 6 char. name of sg 60 ?> = strng4
Enter max string length for sg 60 ?> = 30
MUPF?> =
UPF?>
    end of file encountered

```

RMIS View/Print Document Cover Sheet

This document was retrieved from the Documentation and Records Management (DRM) ISEARCH System. It is intended for Information only and may not be the most recent or updated version. Contact a Document Service Center (see Hanford Info for locations) if you need additional retrieval information.

Accession #: D196079535

Document #: SD-534-CSWL-001

Title/Desc:

SEQUENCE & BATCH LANGUAGE PROGRAMS & ALARM RELATED
C PROGRAMS FOR THE 242A MCS [SEC 3 OF 6 PAGE 253
TO 383]

Pages: 133

This document was too large to scan as a whole document, therefore it required breaking into smaller sections.

Document number: SD-534-CSWL-001

Section 3 of 6

Title: Sequence and Batch Language programs
and Alarm Related "C" Programs for the
242-A MCS

Date: 4/15/96 Revision: 3

Originator: Berger, J.F

Co: _____

Recipient: _____

Co: _____

References: ECN-627504

WHC-SD-534-CSWL-001 REV 3

BLANKK

PAGE 253

```

UPF Version 6.3-2      05-Dec-94 09:39:00
UPF?> = de blankk
UPF?> = cr blankk,*
Enter 4 character name of RPF to associate?> = empty
MUPF?> = add 192 fg
Enter new 6 char. name of fg 0 ?> = fal m0
Enter new 6 char. name of fg 1 ?> = fal m1
Enter new 6 char. name of fg 2 ?> = fal m2
Enter new 6 char. name of fg 3 ?> = fal m3
Enter new 6 char. name of fg 4 ?> = fal m4
Enter new 6 char. name of fg 5 ?> = fal m5
Enter new 6 char. name of fg 6 ?> = fal m6
Enter new 6 char. name of fg 7 ?> = fal m7
Enter new 6 char. name of fg 8 ?> = fal m8
Enter new 6 char. name of fg 9 ?> = fal m9
Enter new 6 char. name of fg 10 ?> = fal m10
Enter new 6 char. name of fg 11 ?> = fal m11
Enter new 6 char. name of fg 12 ?> = fal m12
Enter new 6 char. name of fg 13 ?> = fal m13
Enter new 6 char. name of fg 14 ?> = fal m14
Enter new 6 char. name of fg 15 ?> = fal m15
Enter new 6 char. name of fg 16 ?> = fal m16
Enter new 6 char. name of fg 17 ?> = fal m17
Enter new 6 char. name of fg 18 ?> = fal m18
Enter new 6 char. name of fg 19 ?> = fal m19
Enter new 6 char. name of fg 20 ?> = fal m20
Enter new 6 char. name of fg 21 ?> = fal m21
Enter new 6 char. name of fg 22 ?> = fal m22
Enter new 6 char. name of fg 23 ?> = fal m23
Enter new 6 char. name of fg 24 ?> = fal m24
Enter new 6 char. name of fg 25 ?> = fal m25
Enter new 6 char. name of fg 26 ?> = fal m26
Enter new 6 char. name of fg 27 ?> = fal m27
Enter new 6 char. name of fg 28 ?> = fal m28
Enter new 6 char. name of fg 29 ?> = fal m29
Enter new 6 char. name of fg 30 ?> = fal m30
Enter new 6 char. name of fg 31 ?> = fal m31
Enter new 6 char. name of fg 32 ?> = fal m32
Enter new 6 char. name of fg 33 ?> = fal m33
Enter new 6 char. name of fg 34 ?> = fal m34
Enter new 6 char. name of fg 35 ?> = fal m35
Enter new 6 char. name of fg 36 ?> = fal m36
Enter new 6 char. name of fg 37 ?> = fal m37
Enter new 6 char. name of fg 38 ?> = fal m38
Enter new 6 char. name of fg 39 ?> = fal m39
Enter new 6 char. name of fg 40 ?> = fal m40
Enter new 6 char. name of fg 41 ?> = fal m41
Enter new 6 char. name of fg 42 ?> = fal m42
Enter new 6 char. name of fg 43 ?> = fal m43
Enter new 6 char. name of fg 44 ?> = fal m44

```

Enter new 6 char. name of fg 45 ?> = fal45
 Enter new 6 char. name of fg 46 ?> = fal46
 Enter new 6 char. name of fg 47 ?> = fal47
 Enter new 6 char. name of fg 48 ?> = fal48
 Enter new 6 char. name of fg 49 ?> = fal49
 Enter new 6 char. name of fg 50 ?> = fal50
 Enter new 6 char. name of fg 51 ?> = fal51
 Enter new 6 char. name of fg 52 ?> = fal52
 Enter new 6 char. name of fg 53 ?> = fal53
 Enter new 6 char. name of fg 54 ?> = fal54
 Enter new 6 char. name of fg 55 ?> = fal55
 Enter new 6 char. name of fg 56 ?> = fal56
 Enter new 6 char. name of fg 57 ?> = fal57
 Enter new 6 char. name of fg 58 ?> = fal58
 Enter new 6 char. name of fg 59 ?> = fal59
 Enter new 6 char. name of fg 60 ?> = fal60
 Enter new 6 char. name of fg 61 ?> = fal61
 Enter new 6 char. name of fg 62 ?> = fal62
 Enter new 6 char. name of fg 63 ?> = fal63
 Enter new 6 char. name of fg 64 ?> = fal64
 Enter new 6 char. name of fg 65 ?> = fal65
 Enter new 6 char. name of fg 66 ?> = fal66
 Enter new 6 char. name of fg 67 ?> = fal67
 Enter new 6 char. name of fg 68 ?> = fal68
 Enter new 6 char. name of fg 69 ?> = fal69
 Enter new 6 char. name of fg 70 ?> = fal70
 Enter new 6 char. name of fg 71 ?> = fal71
 Enter new 6 char. name of fg 72 ?> = fal72
 Enter new 6 char. name of fg 73 ?> = fal73
 Enter new 6 char. name of fg 74 ?> = fal74
 Enter new 6 char. name of fg 75 ?> = fal75
 Enter new 6 char. name of fg 76 ?> = fal76
 Enter new 6 char. name of fg 77 ?> = fal77
 Enter new 6 char. name of fg 78 ?> = fal78
 Enter new 6 char. name of fg 79 ?> = fal79
 Enter new 6 char. name of fg 80 ?> = flg0
 Enter new 6 char. name of fg 81 ?> = flg1
 Enter new 6 char. name of fg 82 ?> = flg2
 Enter new 6 char. name of fg 83 ?> = flg3
 Enter new 6 char. name of fg 84 ?> = flg4
 Enter new 6 char. name of fg 85 ?> = flg5
 Enter new 6 char. name of fg 86 ?> = flg6
 Enter new 6 char. name of fg 87 ?> = flg7
 Enter new 6 char. name of fg 88 ?> = flg8
 Enter new 6 char. name of fg 89 ?> = flg9
 Enter new 6 char. name of fg 90 ?> = flg10
 Enter new 6 char. name of fg 91 ?> = flg11
 Enter new 6 char. name of fg 92 ?> = flg12
 Enter new 6 char. name of fg 93 ?> = flg13
 Enter new 6 char. name of fg 94 ?> = flg14

Enter new 6 char. name of fg 95 ?> = flg15
Enter new 6 char. name of fg 96 ?> = flg16
Enter new 6 char. name of fg 97 ?> = flg17
Enter new 6 char. name of fg 98 ?> = flg18
Enter new 6 char. name of fg 99 ?> = flg19
Enter new 6 char. name of fg 100 ?> = flg20
Enter new 6 char. name of fg 101 ?> = flg21
Enter new 6 char. name of fg 102 ?> = flg22
Enter new 6 char. name of fg 103 ?> = flg23
Enter new 6 char. name of fg 104 ?> = flg24
Enter new 6 char. name of fg 105 ?> = flg25
Enter new 6 char. name of fg 106 ?> = flg26
Enter new 6 char. name of fg 107 ?> = flg27
Enter new 6 char. name of fg 108 ?> = flg28
Enter new 6 char. name of fg 109 ?> = flg29
Enter new 6 char. name of fg 110 ?> = flg30
Enter new 6 char. name of fg 111 ?> = flg31
Enter new 6 char. name of fg 112 ?> = flg32
Enter new 6 char. name of fg 113 ?> = flg33
Enter new 6 char. name of fg 114 ?> = flg34
Enter new 6 char. name of fg 115 ?> = flg35
Enter new 6 char. name of fg 116 ?> = flg36
Enter new 6 char. name of fg 117 ?> = flg37
Enter new 6 char. name of fg 118 ?> = flg38
Enter new 6 char. name of fg 119 ?> = flg39
Enter new 6 char. name of fg 120 ?> = flg40
Enter new 6 char. name of fg 121 ?> = flg41
Enter new 6 char. name of fg 122 ?> = flg42
Enter new 6 char. name of fg 123 ?> = flg43
Enter new 6 char. name of fg 124 ?> = flg44
Enter new 6 char. name of fg 125 ?> = flg45
Enter new 6 char. name of fg 126 ?> = flg46
Enter new 6 char. name of fg 127 ?> = flg47
Enter new 6 char. name of fg 128 ?> = flg48
Enter new 6 char. name of fg 129 ?> = flg49
Enter new 6 char. name of fg 130 ?> = flg50
Enter new 6 char. name of fg 131 ?> = flg51
Enter new 6 char. name of fg 132 ?> = flg52
Enter new 6 char. name of fg 133 ?> = flg53
Enter new 6 char. name of fg 134 ?> = flg54
Enter new 6 char. name of fg 135 ?> = flg55
Enter new 6 char. name of fg 136 ?> = flg56
Enter new 6 char. name of fg 137 ?> = flg57
Enter new 6 char. name of fg 138 ?> = flg58
Enter new 6 char. name of fg 139 ?> = flg59
Enter new 6 char. name of fg 140 ?> = flg60
Enter new 6 char. name of fg 141 ?> = flg61
Enter new 6 char. name of fg 142 ?> = flg62
Enter new 6 char. name of fg 143 ?> = flg63
Enter new 6 char. name of fg 144 ?> = flg64

```

Enter new 6 char. name of fg 145 ?> = flg65
Enter new 6 char. name of fg 146 ?> = flg66
Enter new 6 char. name of fg 147 ?> = flg67
Enter new 6 char. name of fg 148 ?> = flg68
Enter new 6 char. name of fg 149 ?> = flg69
Enter new 6 char. name of fg 150 ?> = flg70
Enter new 6 char. name of fg 151 ?> = flg71
Enter new 6 char. name of fg 152 ?> = flg72
Enter new 6 char. name of fg 153 ?> = flg73
Enter new 6 char. name of fg 154 ?> = flg74
Enter new 6 char. name of fg 155 ?> = flg75
Enter new 6 char. name of fg 156 ?> = flg76
Enter new 6 char. name of fg 157 ?> = flg77
Enter new 6 char. name of fg 158 ?> = flg78
Enter new 6 char. name of fg 159 ?> = flg79
Enter new 6 char. name of fg 160 ?> = fx0
Enter new 6 char. name of fg 161 ?> = fx1
Enter new 6 char. name of fg 162 ?> = fx2
Enter new 6 char. name of fg 163 ?> = fx3
Enter new 6 char. name of fg 164 ?> = fx4
Enter new 6 char. name of fg 165 ?> = fx5
Enter new 6 char. name of fg 166 ?> = fx6
Enter new 6 char. name of fg 167 ?> = fx7
Enter new 6 char. name of fg 168 ?> = fs10
Enter new 6 char. name of fg 169 ?> = fs11
Enter new 6 char. name of fg 170 ?> = fs12
Enter new 6 char. name of fg 171 ?> = fs13
Enter new 6 char. name of fg 172 ?> = fs14
Enter new 6 char. name of fg 173 ?> = fs15
Enter new 6 char. name of fg 174 ?> = fs16
Enter new 6 char. name of fg 175 ?> = fs17
Enter new 6 char. name of fg 176 ?> = fhold
Enter new 6 char. name of fg 177 ?> = fning
Enter new 6 char. name of fg 178 ?> = fabort
Enter new 6 char. name of fg 179 ?> = fstart
Enter new 6 char. name of fg 180 ?> = fpass
Enter new 6 char. name of fg 181 ?> = ftrip0
Enter new 6 char. name of fg 182 ?> = ftrip1
Enter new 6 char. name of fg 183 ?> = ftrip2
Enter new 6 char. name of fg 184 ?> = ftrip3
Enter new 6 char. name of fg 185 ?> = ftrip4
Enter new 6 char. name of fg 186 ?> = fspar0
Enter new 6 char. name of fg 187 ?> = fspar1
Enter new 6 char. name of fg 188 ?> = fspar2
Enter new 6 char. name of fg 189 ?> = fspar3
Enter new 6 char. name of fg 190 ?> = fspar4
Enter new 6 char. name of fg 191 ?> = fspar5
MUPF?> = add 92 in
Enter new 6 char. name of in 0 ?> = bitval
Enter new 6 char. name of in 1 ?> = iloop

```


Enter new 6 char. name of in 2 ?> = index
Enter new 6 char. name of in 3 ?> = remain
Enter new 6 char. name of in 4 ?> = trip
Enter new 6 char. name of in 5 ?> = nadi
Enter new 6 char. name of in 6 ?> = blkval
Enter new 6 char. name of in 7 ?> = sort
Enter new 6 char. name of in 8 ?> = bval
Enter new 6 char. name of in 9 ?> = hldstp
Enter new 6 char. name of in 10 ?> = errn
Enter new 6 char. name of in 11 ?> = errl
Enter new 6 char. name of in 12 ?> = int0
Enter new 6 char. name of in 13 ?> = int1
Enter new 6 char. name of in 14 ?> = int2
Enter new 6 char. name of in 15 ?> = int3
Enter new 6 char. name of in 16 ?> = int4
Enter new 6 char. name of in 17 ?> = int5
Enter new 6 char. name of in 18 ?> = int6
Enter new 6 char. name of in 19 ?> = int7
Enter new 6 char. name of in 20 ?> = int8
Enter new 6 char. name of in 21 ?> = int9
Enter new 6 char. name of in 22 ?> = int10
Enter new 6 char. name of in 23 ?> = int11
Enter new 6 char. name of in 24 ?> = int12
Enter new 6 char. name of in 25 ?> = int13
Enter new 6 char. name of in 26 ?> = int14
Enter new 6 char. name of in 27 ?> = int15
Enter new 6 char. name of in 28 ?> = int16
Enter new 6 char. name of in 29 ?> = int17
Enter new 6 char. name of in 30 ?> = int18
Enter new 6 char. name of in 31 ?> = int19
Enter new 6 char. name of in 32 ?> = int20
Enter new 6 char. name of in 33 ?> = int21
Enter new 6 char. name of in 34 ?> = int22
Enter new 6 char. name of in 35 ?> = int23
Enter new 6 char. name of in 36 ?> = int24
Enter new 6 char. name of in 37 ?> = int25
Enter new 6 char. name of in 38 ?> = int26
Enter new 6 char. name of in 39 ?> = int27
Enter new 6 char. name of in 40 ?> = int28
Enter new 6 char. name of in 41 ?> = int29
Enter new 6 char. name of in 42 ?> = int30
Enter new 6 char. name of in 43 ?> = int31
Enter new 6 char. name of in 44 ?> = int32
Enter new 6 char. name of in 45 ?> = int33
Enter new 6 char. name of in 46 ?> = int34
Enter new 6 char. name of in 47 ?> = int35
Enter new 6 char. name of in 48 ?> = int36
Enter new 6 char. name of in 49 ?> = int37
Enter new 6 char. name of in 50 ?> = int38
Enter new 6 char. name of in 51 ?> = int39

```

Enter new 6 char. name of in 52 ?> = int40
Enter new 6 char. name of in 53 ?> = int41
Enter new 6 char. name of in 54 ?> = int42
Enter new 6 char. name of in 55 ?> = int43
Enter new 6 char. name of in 56 ?> = int44
Enter new 6 char. name of in 57 ?> = int45
Enter new 6 char. name of in 58 ?> = int46
Enter new 6 char. name of in 59 ?> = int47
Enter new 6 char. name of in 60 ?> = int48
Enter new 6 char. name of in 61 ?> = int49
Enter new 6 char. name of in 62 ?> = int50
Enter new 6 char. name of in 63 ?> = int51
Enter new 6 char. name of in 64 ?> = int52
Enter new 6 char. name of in 65 ?> = int53
Enter new 6 char. name of in 66 ?> = int54
Enter new 6 char. name of in 67 ?> = int55
Enter new 6 char. name of in 68 ?> = int56
Enter new 6 char. name of in 69 ?> = int57
Enter new 6 char. name of in 70 ?> = int58
Enter new 6 char. name of in 71 ?> = int59
Enter new 6 char. name of in 72 ?> = int60
Enter new 6 char. name of in 73 ?> = int61
Enter new 6 char. name of in 74 ?> = int62
Enter new 6 char. name of in 75 ?> = int63
Enter new 6 char. name of in 76 ?> = int64
Enter new 6 char. name of in 77 ?> = int65
Enter new 6 char. name of in 78 ?> = int66
Enter new 6 char. name of in 79 ?> = int67
Enter new 6 char. name of in 80 ?> = int68
Enter new 6 char. name of in 81 ?> = int69
Enter new 6 char. name of in 82 ?> = int70
Enter new 6 char. name of in 83 ?> = int71
Enter new 6 char. name of in 84 ?> = int72
Enter new 6 char. name of in 85 ?> = int73
Enter new 6 char. name of in 86 ?> = int74
Enter new 6 char. name of in 87 ?> = int75
Enter new 6 char. name of in 88 ?> = int76
Enter new 6 char. name of in 89 ?> = int77
Enter new 6 char. name of in 90 ?> = int78
Enter new 6 char. name of in 91 ?> = int79
MUPF?> = add 30 fl
Enter new 6 char. name of fl 0 ?> = float0
Enter new 6 char. name of fl 1 ?> = float1
Enter new 6 char. name of fl 2 ?> = float2
Enter new 6 char. name of fl 3 ?> = float3
Enter new 6 char. name of fl 4 ?> = float4
Enter new 6 char. name of fl 5 ?> = float5
Enter new 6 char. name of fl 6 ?> = float6
Enter new 6 char. name of fl 7 ?> = float7
Enter new 6 char. name of fl 8 ?> = float8

```

```

Enter new 6 char. name of fl 9 ?> = float9
Enter new 6 char. name of fl 10 ?> = ai0
Enter new 6 char. name of fl 11 ?> = ai1
Enter new 6 char. name of fl 12 ?> = ai2
Enter new 6 char. name of fl 13 ?> = ai3
Enter new 6 char. name of fl 14 ?> = ai4
Enter new 6 char. name of fl 15 ?> = ai5
Enter new 6 char. name of fl 16 ?> = ai6
Enter new 6 char. name of fl 17 ?> = ai7
Enter new 6 char. name of fl 18 ?> = ai8
Enter new 6 char. name of fl 19 ?> = ai9
Enter new 6 char. name of fl 20 ?> = ai10
Enter new 6 char. name of fl 21 ?> = ai11
Enter new 6 char. name of fl 22 ?> = ai12
Enter new 6 char. name of fl 23 ?> = ai13
Enter new 6 char. name of fl 24 ?> = ai14
Enter new 6 char. name of fl 25 ?> = ai15
Enter new 6 char. name of fl 26 ?> = ai16
Enter new 6 char. name of fl 27 ?> = ai17
Enter new 6 char. name of fl 28 ?> = ai18
Enter new 6 char. name of fl 29 ?> = ai19
MUPF?> = add 6 pt
Enter new 6 char. name of pt 0 ?> = pepn
Enter new 6 char. name of pt 1 ?> = pdin
Enter new 6 char. name of pt 2 ?> = pdev
Enter new 6 char. name of pt 3 ?> = pai
Enter new 6 char. name of pt 4 ?> = palarm
Enter new 6 char. name of pt 5 ?> = pspar0
MUPF?> = add 38 sg
Enter new 6 char. name of sg 0 ?> = lsg0
Enter max string length for sg 0 ?> = 30
Enter new 6 char. name of sg 1 ?> = lsg1
Enter max string length for sg 1 ?> = 30
Enter new 6 char. name of sg 2 ?> = lsg2
Enter max string length for sg 2 ?> = 30
Enter new 6 char. name of sg 3 ?> = lsg3
Enter max string length for sg 3 ?> = 30
Enter new 6 char. name of sg 4 ?> = lsg4
Enter max string length for sg 4 ?> = 30
Enter new 6 char. name of sg 5 ?> = lsg5
Enter max string length for sg 5 ?> = 30
Enter new 6 char. name of sg 6 ?> = lsg6
Enter max string length for sg 6 ?> = 30
Enter new 6 char. name of sg 7 ?> = lsg7
Enter max string length for sg 7 ?> = 30
Enter new 6 char. name of sg 8 ?> = unit0
Enter max string length for sg 8 ?> = 30
Enter new 6 char. name of sg 9 ?> = unit1
Enter max string length for sg 9 ?> = 30
Enter new 6 char. name of sg 10 ?> = unit2

```

Enter max string length for sg 10 ?> = 30
Enter new 6 char. name of sg 11 ?> = unit3
Enter max string length for sg 11 ?> = 30
Enter new 6 char. name of sg 12 ?> = unit4
Enter max string length for sg 12 ?> = 30
Enter new 6 char. name of sg 13 ?> = unit5
Enter max string length for sg 13 ?> = 30
Enter new 6 char. name of sg 14 ?> = unit6
Enter max string length for sg 14 ?> = 30
Enter new 6 char. name of sg 15 ?> = unit7
Enter max string length for sg 15 ?> = 30
Enter new 6 char. name of sg 16 ?> = unit8
Enter max string length for sg 16 ?> = 30
Enter new 6 char. name of sg 17 ?> = unit9
Enter max string length for sg 17 ?> = 30
Enter new 6 char. name of sg 18 ?> = unit10
Enter max string length for sg 18 ?> = 30
Enter new 6 char. name of sg 19 ?> = unit11
Enter max string length for sg 19 ?> = 30
Enter new 6 char. name of sg 20 ?> = unit12
Enter max string length for sg 20 ?> = 30
Enter new 6 char. name of sg 21 ?> = unit13
Enter max string length for sg 21 ?> = 30
Enter new 6 char. name of sg 22 ?> = unit14
Enter max string length for sg 22 ?> = 30
Enter new 6 char. name of sg 23 ?> = unit15
Enter max string length for sg 23 ?> = 30
Enter new 6 char. name of sg 24 ?> = unit16
Enter max string length for sg 24 ?> = 30
Enter new 6 char. name of sg 25 ?> = unit17
Enter max string length for sg 25 ?> = 30
Enter new 6 char. name of sg 26 ?> = unit18
Enter max string length for sg 26 ?> = 30
Enter new 6 char. name of sg 27 ?> = unit19
Enter max string length for sg 27 ?> = 30
Enter new 6 char. name of sg 28 ?> = unit20
Enter max string length for sg 28 ?> = 30
Enter new 6 char. name of sg 29 ?> = unit21
Enter max string length for sg 29 ?> = 30
Enter new 6 char. name of sg 30 ?> = unit22
Enter max string length for sg 30 ?> = 30
Enter new 6 char. name of sg 31 ?> = unit23
Enter max string length for sg 31 ?> = 30
Enter new 6 char. name of sg 32 ?> = unit24
Enter max string length for sg 32 ?> = 30
Enter new 6 char. name of sg 33 ?> = unit25
Enter max string length for sg 33 ?> = 30
Enter new 6 char. name of sg 34 ?> = unit26
Enter max string length for sg 34 ?> = 30
Enter new 6 char. name of sg 35 ?> = unit27

WMC-SD-534-CSWL-001 REV 3

Enter max string length for sg 35 ?> = 30
Enter new 6 char. name of sg 36 ?> = unit28
Enter max string length for sg 36 ?> = 30
Enter new 6 char. name of sg 37 ?> = unit29
Enter max string length for sg 37 ?> = 30
MUPF?>
end of file encountered
UPF?>
end of file encountered

APPENDIX E - SEQUENCE AND BATCH LANGUAGE (SABL)

224BATCH

```

1      :      2 4 4 B A T C H . S E Q                                VERSION 4
2      :
3      :
4      :      Function:      Sequence program for Lamp function,
5      :                      message display on Graphic number 340,
6      :                      and printer messages.
7      :
8      :      Written By      : R. N. Kersey
9      :      Date           : 05-31-92
10     :      Facility       : 242-A Evaporator Bldg.
11     :      Machine        : VAX 4104-A
12     :      Op. System     : VAX/VMS 5.5-2AH
13     :      D/3 Version    : 6.3-2 BASELINE
14     :      Compiler       : SIC
15     :      Build command  : @[320,20]244BATCH
16     :
17     :      UNIT = 1, SKDKEY = 19
18     :
19     :      Assoc. UPF:      [300,1]_SHOTR0.____    SHOTR0.LST
20     :      Assoc. RPF:      [300,1] FARM.____    FARM.LST
21     :      SDB:            [300,1]PCMOASDB.____    PCMOASDB.CMD
22     :
23     :      POINTER      SDB VAR.      FUNCTION
24     :      -----
25     :      PSTAT      INTn      Current EPN alarm state and is integer tied to
26     :                      graphic flashing string of value:
27     :                      (n=0-100)      0 = previously not alarmed
28     :                      1 = previously alarmed, not ack
29     :                      2 = previously alarmed, is ack
30     :                      3 = alarm is inhibited
31     :                      4 = input is off scan
32     :                      5 = input is bad
33     :      LEAKn      Special INT variable for multiple white alarms on an EPN
34     :      'EPN'
35     :      PTYPEn      alarm type 0 = none, 1 = white, 2 = yellow,
36     :                      3 = yellow and yellow, 4 = red,
37     :                      6 = yellow and red
38     :      PREV      FRVn      Reverse acting (not fail safe) flag (1=set)
39     :      PSKIDn      Used for analog alarms
40     :      PPASn      Turn lamp off when alarm is ACKnowledged
41     :      PALARM      FALMn      Analog alarm flag
42     :      ---      FNWNG      Unused error flag
43     :
44     :      MODIFICATION LOG:
45     :
46     :      09-JUN-93      Added comments and baselined for      M. Teats
47     :                      D/3 version 3.3.  CCN# 93JUN01-01.
48     :
49     :      15-DEC-94      Revised header to indicate the new VMS      J. Berger
50     :                      system. Rebase line document. Removed
51     :                      obsolete Select() statements.
52     :
53     :      10-MAY-95      Revised program to delete EPNs P-244-A2, J. Berger
54     :                      II-P244A2, YS-P244A2, YS244A2I1. See
55     :                      ECN 622501 and EE-94-00929. (JFB01)

```

MHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:00:31 08-Jan-96

Page 2

```

56      ;
57      ;      20-OCT-95      Revised program to delete EPNs for 244A      J. Berger
58      ;      ;      that were never used. See ECN 627411 and
59      ;      ;      EE-94-00929. (JFB02)
60      ;
61      ;      08-JAN-96      Deleted RSH-SP & PSH-FP per MCS-96-001      J. Berger
62      ;
63      ;      END MOD LOG
64      ;
65      UNIT      A244
66      0000      NOP
67      0001      STAR = 20      ; Start message
68      0002      SKDKEY =19      ; SKID key
69      0003      ADI = 0      ; Alarm Destination Index = 0
70      0004      FSN0 = 0      ; No alarm yet
71      0005      TIMEROFF STMRO      ; Alarm timer off
72      0006      TIMERCLR STMRO      ; Alarm timer clear
73      0007      STRNG0 = " "      ; Blank 30 Character string
74      0008      ERRN = 0      ; Set error number to zero
75      0009      ERRL = 0      ; Set error line to zero
76      0010      ASSIGN PDEV = '244ALARM' ; Alarm device
77      0011      LET FTRIP4 = 0      ; FTRIP4 = 1 when audible horn is to be turned on
78      0012      GETUNIT TRIP      ; Which sequence unit is program running in
79      0013      IF TRIP I= 1 THEN      ; Abort if running in wrong seq. unit
80      BEGIN
81      0014      FSTART = 1
82      0015      ILOOP = STAR + 1
83      0016      GOSUB ACKK
84      0017      FSTART = 1
85      0018      PRINT ADI," >>>> ATTEMPT TO RUN 244BATCH IN WRONG UNIT"
86      0019      GOTO TERM
87      END
88      ;
89      ;      B E G I N   I N I T I A L I Z I N G
90      ;
91      0020      SSMGO = "INIT"
92      0021      INIT:      LET FABORT = 0      ; If FABORT = 1 then abort program
93      0022      LET FSTART = 0      ; Start flag = 0
94      0023      LAMP ADI,SKDKEY,0      ; Turn off the annunciator key
95      0024      LET LSG0 = " "      ; Clear the string variables
96      0025      LET LSG1 = " "
97      0026      LET LSG2 = " "
98      0027      LET LSG3 = " "
99      0028      LET LSG4 = " "
100     0029      PDEV:DEV_TMPO = 1      ; Set alarm device to non-alarm state
101     0030      IF AND(PDEV:DEV_ALMCT,16) THEN      ; If alarm device is inhibited, remove the inhibit
102     0031      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
103     0032      IF AND(PDEV:DEV_ALMCT,32) THEN      ; If alarm device is acknowledged, clear it
104     0033      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
105     ;
106     ;      First, set all flags and integers:
107     ;
108     ;      EPN      POINT      INT      TYPE      FRV      WHITE
109     ;      -----      -      -      -      -      -
110     ;      244ALARM      DEV      -      -      -      -

```



```

111      : P-244-A1      DEV      -      -      -
112      : SPARE        DEV      -      -      -
113      : P-244-A3      DEV      -      -      -
114      : SPARE        DEV      -      -      -
115      : SPARE        DEV      -      -      -
116      : SPARE        DEV      -      -      -
117      : SPARE        DEV      -      -      -
118      : SPARE        DEV      -      -      -
119      : SPARE        DEV      -      -      -
120      :
121      : YS244A11L    DIM      0      2      0
122      : YS-P244A1    DIM      1      0      -
123      : SPARE        DIM      2      -      -
124      : SPARE        DIM      3      -      -
125      : YS-P244A3    DIM      4      0      -
126      : SPARE        DIM      5      0      -
127      : ZS-MOV-2F    DIM      6      0      -
128      : ZS-MOV2DR    DIM      7      0      -
129      : SPARE        DIM      8      -      -
130      : SPARE        DIM      9      -      -
131      : SPARE        DIM     10      -      -
132      : SPARE        DIM     11      -      -
133      : SPARE        DIM     12      -      -
134      : SPARE        DIM     13      -      -
135      : SPARE        DIM     14      -      -
136      : PSH-SN215    DIM     15      2      0
137      : PSH-SN216    DIM     16      2      0
138      : LDS-PP DIM    17      2      0
139      : SPARE DIM     18      -      -
140      : PSH-FP{SPARE} DIM     19      -      -
141      :
142      : I1-P244A1    AI       20      -      -
143      : SPARE        AI       21      -      -
144      : I1-P244A3    AI       22      -      -
145      : D1-244-SP    AI       23      -      -
146      : WRI-244SP    AI       24      -      -
147      : SPARE        AI       25      -      -
148      : LI-244TK     AI       26      -      -
149      : D1-244-TK    AI       27      -      -
150      : WFI-244TK    AI       28      -      -
151      : I1-244-TK    AI       29      -      -
152      : P1-244-TK    AI       30      -      -
153      : I1-244-TK    AI       31      -      -
154      :
155      : ASSIGN PDIM = 'YS244A11L'
156      : ASSIGN PSTAT = INTO
157      : ASSIGN PTY = TYPE0
158      : ASSIGN PREV = ERVO
159      : ASSIGN PSKID = SKID00
160      : ASSIGN PPASS = FPAS0
161      :
162      : FOR ILOOP = 0 TO 31
163      : LET PPASS = 0
164      : LET PREV = 0
165      : LET PTY = 0

```

166	0044	LET PSTAT = 0	
167	0045	LET PSKID = 0	
168	0046	IF ILOOP = 2 THEN	: JFB01
169		BEGIN	: JFB01
170	0047	STEP PDIN BY 1	: JFB01
171	0048	GOTO SKP	: JFB01
172		END	: JFB01
173	0049	IF ILOOP = 3 THEN	: JFB01
174		BEGIN	: JFB01
175	0050	STEP PDIN BY 1	: JFB01
176	0051	GOTO SKP	: JFB01
177		END	: JFB01
178	0052	IF ILOOP = 5 THEN	: JFB02
179		BEGIN	: JFB02
180	0053	STEP PDIN BY 1	: JFB02
181	0054	GOTO SKP	: JFB02
182		END	: JFB02
183	0055	IF ILOOP = 8 THEN	: JFB02
184		BEGIN	: JFB02
185	0056	STEP PDIN BY 1	: JFB02
186	0057	GOTO SKP	: JFB02
187		END	: JFB02
188	0058	IF ILOOP = 9 THEN	: JFB02
189		BEGIN	: JFB02
190	0059	STEP PDIN BY 1	: JFB02
191	0060	GOTO SKP	: JFB02
192		END	: JFB02
193	0061	IF ILOOP = 10 THEN	: JFB02
194		BEGIN	: JFB02
195	0062	STEP PDIN BY 1	: JFB02
196	0063	GOTO SKP	: JFB02
197		END	: JFB02
198	0064	IF ILOOP = 11 THEN	: JFB02
199		BEGIN	: JFB02
200	0065	STEP PDIN BY 1	: JFB02
201	0066	GOTO SKP	: JFB02
202		END	: JFB02
203	0067	IF ILOOP = 12 THEN	: JFB02
204		BEGIN	: JFB02
205	0068	STEP PDIN BY 1	: JFB02
206	0069	GOTO SKP	: JFB02
207		END	: JFB02
208	0070	IF ILOOP = 13 THEN	: JFB02
209		BEGIN	: JFB02
210	0071	STEP PDIN BY 1	: JFB02
211	0072	GOTO SKP	: JFB02
212		END	: JFB02
213	0073	IF ILOOP = 14 THEN	: JFB02
214		BEGIN	: JFB02
215	0074	STEP PDIN BY 1	: JFB02
216	0075	GOTO SKP	: JFB02
217		END	: JFB02
218	0076	IF ILOOP = 18 THEN	: JFB03
219		BEGIN	: JFB03
220	0077	STEP PDIN BY 1	: JFB03

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:00:31 08-Jan-96

Page 5

```

221 0078      GOTO SKP                                ; JFB03
222          END                                    ; JFB03
223 0079      IF ILOOP = 19 THEN                      ; JFB03
224          BEGIN                                ; JFB03
225 0080      STEP PDIN BY 1                          ; JFB03
226 0081      GOTO SKP                                ; JFB03
227          END                                    ; JFB03
228 0082      IF ILOOP = 21 THEN                      ; JFB02
229          BEGIN                                ; JFB02
230 0083      GOTO SKP                                ; JFB02
231          END                                    ; JFB02
232 0084      IF ILOOP = 25 THEN                      ; JFB02
233          BEGIN                                ; JFB02
234 0085      GOTO SKP                                ; JFB02
235          END                                    ; JFB02
236 0086      IF ILOOP < 20 THEN                      ; For DINs
237          BEGIN
238 0087      IF (AND(PDIN:DIN_DFLG,3)) THEN          ; Check for alarms (must be red or yellow)
239          BEGIN
240 0088          LET PTYPE = 2                        ; Default to yellow for DINs
241 0090          IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1 ; Reverse acting
242 0092          IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4 ; Set to critical alarm
243          END
244 0093          STEP PDIN BY 1                      ; Point to next DIN
245          END
246 0095      IF ILOOP > 30 THEN LET PTYPE = 1
247 0096      SKP: STEP PSTAT BY 1
248 0097          STEP PTYPE BY 1
249 0098          STEP PREV BY 1
250 0099          STEP PPASS BY 1
251 0100          STEP PSKID BY 1
252 0101          NEXT ILOOP
253      ;
254 0102      LET TYPE0 = 2                          ; YS244A11L
255      ;                                           ; YS244A21L      JFB01
256 0103      LET TYPE15 = 0                        ; This EPN now alarms in Unit XYZFRM on Graphic 321
257 0104      LET TYPE16 = 0                        ; This EPN now alarms in Unit XYZFRM on Graphic 321
258 0105      LET TYPE17 = 2                        ; LDS-PP
259      ;                                           ; RSH-SP
260      ;                                           ; PSH-FP
261 0106      LET TYPE24 = 2                        ; WFI-244SP HI
262 0107      LET TYPE26 = 6                        ; LI-244TK LOLO, LO, HI, HIHI
263 0108      LET TYPE28 = 6                        ; WFI-244TK LOLO, LO, HI, HIHI
264      ;
265 0109      LET FRV15 = 0                        ; This EPN now alarms in Unit XYZFRM on Graphic 321
266 0110      LET FRV16 = 0                        ; This EPN now alarms in Unit XYZFRM on Graphic 321
267 0111      LET FRV17 = 0                        ; LDS-PP
268      ;                                           ; RSH-SP
269      ;                                           ; PSH-FP
270      ;
271 0112      LET A10 = 200.0 ; TI-244-TK TEMP HI
272      ;
273      ; END INITIALIZING
274      ;
275      ; SEND PROGRAM START MESSAGE

```

```

276      ;
277 0113      PTYPE = 0
278 0114      ILOOP = STAR
279 0115      FSTART = 1
280 0116      GOSUB ACKK
281 0117      ON ERROR GOSUB ERRSUB
282      ;
283      ;      BEGIN PROGRAM LOOP
284      ;
285 0118  AGAIN: MARK STEP "WTUNTIL",0,0
286      ;      WAITUNTIL SELECT() ; FABORT      ; Wait until micro is selected or program aborted
287      ;      IF (FABORT) THEN GOTO TERM      ; Terminate program if FABORT flag is set
288 0119      MARK STEP "RNNING",0,0
289 0120      WAIT 1
290 0121      IF (FSMO) THEN      ; If alarm flag set
291 0122      IF STMRO > 60 THEN      ; If alarm flag active for 1 minute
292      BEGIN      ; Reset timer and turn off flag
293 0123      TIMEROFF STMRO
294 0124      TIMERCLR STMRO
295 0125      LET FSMO = 0
296 0126      LET ERRN = 0
297 0127      LET ERRL = 0
298      END
299      ;
300      ;      If 244ALARM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
301      ;
302 0128      IF (PDEV:DEV_TMP0 = 2 ) & AND(PDEV:DEV_ALMCT,32) THEN
303      BEGIN
304 0129      PDEV:DEV_TMP0 = 1      ; Put device in acknowledge state
305 0130      LET FTRIP4 = 1      ; Set for Unit alarm acknowledgement, system to graphic
306 0131      LET ACK = 1      ; Set for Unit alarm acknowledgement, graphic to system
307      ;      CRIT
308 0132      LAMP ADI,SKDKEY,1
309      END
310      ;
311      ;      BEGIN DINS:
312      ;
313 0133      ASSIGN PDIN = 'YS244A1IL'
314 0134      ASSIGN PTYPE = TYPED
315 0135      ASSIGN PREV = FRVO
316 0136      ASSIGN PSTAT = INTO
317 0137      ASSIGN PPASS = FPASO
318 0138      CURENT = 0      ; Used to set LAMP status
319 0139      LAST = 0
320 0140      NXT = 0
321 0141      LMPVAR = 0
322 0142      BITVAL = 0
323      ;
324 0143  LOP:  FOR ILOOP = 0 TO 19
325      ;      IF !SELECT() THEN GOTO AGAIN
326 0145      IF ILOOP = 2 THEN GOTO LOP1      ; JF801
327 0147      IF ILOOP = 3 THEN GOTO LOP1      ; JF801
328 0149      IF ILOOP = 5 THEN GOTO LOP1      ; JF801
329 0151      IF ILOOP = 8 THEN GOTO LOP1      ; JF802
330 0153      IF ILOOP = 9 THEN GOTO LOP1      ; JF802

```

```

331 0155      IF ILOOP = 10 THEN GOTO LOP1      ; JFB02
332 0157      IF ILOOP = 11 THEN GOTO LOP1      ; JFB02
333 0159      IF ILOOP = 12 THEN GOTO LOP1      ; JFB02
334 0161      IF ILOOP = 13 THEN GOTO LOP1      ; JFB02
335 0163      IF ILOOP = 14 THEN GOTO LOP1      ; JFB02
336 0165      IF ILOOP = 18 THEN GOTO LOP1      ; JFB02
337 0167      IF ILOOP = 19 THEN GOTO LOP1      ; JFB03
338 0168      IF ( PTYPE > 0 ) THEN              ; If some type of alarm
339          BEGIN                                ; be1
340 0170          IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3 ; If EPW inhibited
341 0172          IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5 ; If EPW is bad input
342 0174          IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4 ; If EPW is disabled
343 0175          IF ( PSTAT < 3 ) THEN            ; If not inhibited
344          BEGIN                                ; be2
345 0176          IF PSTAT = 1 THEN
346          BEGIN                                ; be3
347 0177          IF PTYPE != 1 THEN
348 0179          IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
349 0181          IF (FTRIP4) THEN GOSUB ACK1
350          END                                ; en3
351 0184          IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
352 0185          IF (PSTAT) THEN
353          BEGIN                                ; be4
354 0187          IF PDIN:DIN_VAL != PREV THEN GOSUB CLR
355 0188          NXT = PSTAT
356 0190          IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
357 0191          LAST = OR(LAST,NXT)
358          END                                ; en4
359          END                                ; en2
360 0193      ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0
361          END                                ; en1
362 0195      LOP1:  IF ILOOP < 19 THEN STEP PDIN BY 1
363 0196      STEP PTYPE BY 1                    ; JFB01
364 0197      STEP PREV BY 1
365 0198      STEP PSTAT BY 1
366 0199      STEP PPASS BY 1
367 0200      NEXT ILOOP
368      ;
369      ;
370      ;
371      ;
372      ;
373 0201      ANALOG = 0
374 0202      ASSIGN PAI = 'I1-P244A1'
375 0203      ASSIGN PSKID = SKID20
376 0204      ASSIGN PTYPE = TYPE20
377 0205      ASSIGN PREV = FRV20
378 0206      FOR JLOOP = 20 TO 30
379 0208      IF JLOOP = 21 THEN GOTO LOP2      ; JFB01
380 0210      IF JLOOP = 25 THEN GOTO LOP2      ; JFB02
381      ;
382 0211      IF ISELECT() THEN GOTO AGAIN
383      IF (PTYPE) THEN                        ; If some alarm exists
384      BEGIN                                ; be1
385 0212      BITVAL = PAI:AI_ALMCT
386 0213      IF !AND(BITVAL,16) THEN

```

```

386          BEGIN
387 0214      IF AND(BITVAL,15) THEN ; be2
388          BEGIN ; If alarm exists and it is current
389 0216      IF AND(BITVAL,32) THEN NXT = 2 ; be3 ; If alarm is acknowledged
390          ELSE
391          BEGIN ; be4
392 0217      IF PSKID !=BITVAL THEN
393          BEGIN ; be5
394 0219      IF BITVAL < 3 THEN FALMON = 1
395 0220      LET PSKID = BITVAL
396 0221      LET PDEV:DEV_TMPO = 2
397 0222      NXT = 1
398 0223      LAMP AD1,SKDKEY,2
399          CRIT
400          END ; en5
401 0224      ELSE NXT = 1
402          END ; en4
403          END ; en3
404          ELSE ; Else there is no alarm
405          BEGIN
406 0225      NXT = 0
407 0226      LET PSKID = 0
408          END
409          END ; en2
410          ELSE ; Else alarm is inhibited
411          BEGIN
412 0227      NXT = 0
413 0228      LET PSKID = 0
414          END
415 0229      ANALOG = OR(ANALOG,NXT)
416          END ; en1
417 0230 LOP2: STEP PSKID BY 1
418 0231      STEP PTYPE BY 1 ; JFB01
419 0232      STEP PREV BY 1
420 0233      STEP PAI BY 1
421 0234      NEXT JLOOP
422      ;
423      ;
424      ;
425      ;
426      ;
427 0235 WHT: WHITE = 0
428 0236      SSWGO = "RUN"
429 0237      ASSIGN PSTAT = INT31
430 0238      ASSIGN PTYPE = TYPE31
431 0239      ASSIGN PREV = FRV31
432 0240      ASSIGN PPASS = FPAS31
433 0241      ASSIGN PALARM = A10
434 0242      ILOOP = 31
435 0243      ASSIGN PAI = 'T1-244-TK'
436 0244      GOSUB ANLG
437      ;
438      ;
439      ;
440 0245      MARK STEP "BELOW",0,0

```

```

441 LET FTMP4,4 = 0
442 IF LAST > 3 THEN PRINT AD1," LAMP VARIABLE IS GREATER THAN 3!"
443
444 CURRENT = OR(LAST,WHITE)
445 CURRENT = OR(CURRENT,ANALOG)
446
447 IF AND(CURRENT,1) THEN LMPVAR = 2
448 ELSE IF AND(CURRENT,2) THEN LMPVAR = 1
449 ELSE LMPVAR = 0
450
451 LAMP AD1,SKDKEY,LMPVAR
452 MARK STEP "BOTTOM",0,0
453 MOCRIT
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

```

496      BEGIN                                ; be2
497      0279      IF (PSTAT = 0) THEN          ; If not previously alarmed
498                  BEGIN                        ; be3
499      0281      IF PA1:AI_INVL <= PALARM THEN GOSUB ALM
500                  END                        ; en3
501      ELSE
502      BEGIN
503      0282      IF PSTAT = 1 THEN                ; If previously alarmed
504                  BEGIN                        ; be5
505      0284      IF (FTRIP4) THEN GOSUB ACK1      ; If ACK flag set, then ack it
506      0287      IF PTYPY != 1 THEN IF AND(PA1:AI_ALMCT,32) THEN GOSUB ACK1
507                  END                        ; en5
508      0289      IF PA1:AI_INVL >= PALARM + PA1:AI_DEAD THEN GOSUB CLR
509                  END                        ; en4
510      END                                    ; en2
511      ELSE                                ; If fail-safe, HIHI, or HI
512      BEGIN                                ; be2.1
513      0290      IF PSTAT = 0 THEN
514                  BEGIN
515      0292      IF PA1:AI_INVL >= PALARM THEN GOSUB ALM
516                  END
517      ELSE
518      BEGIN
519      0293      IF PSTAT = 1 THEN
520                  BEGIN
521      0295      IF (FTRIP4) THEN GOSUB ACK1
522      0298      IF PTYPY != 1 THEN IF AND(PA1:AI_ALMCT,32) THEN GOSUB ACK1
523                  END
524      0300      IF PA1:AI_INVL <= PALARM - PA1:AI_DEAD THEN GOSUB CLR
525                  END
526      END                                    ; en2.1
527      0302      IF PSTAT = 3 THEN PSTAT = 0      ; If INHA, then there is no alarm
528      0304      IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
529      0305      ELSE NXT = PSTAT
530      END
531      ELSE
532      BEGIN
533      0306      LET PSTAT = 3
534      0307      NXT = 0
535      END
536      0308      WHITE = OR(WHITE,NXT)
537      END
538      0309      ELSE LET PSTAT = 0
539      0310      STEP PALARM BY 1
540      0311      STEP PSTAT BY 1
541      0312      STEP PTYPY BY 1
542      0313      STEP PREV BY 1
543      0314      STEP PPASS BY 1
544      0315      ILOOP = ILOOP + 1
545      0316      RETURN
546      ;
547      ;
548      ;
549      ;
550      ;

```

END ANALOG WHITE ALARM SUBROUTINE

BEGIN CLEAR, ALARM, ACKNOWLEDGE, MESSAGE, ERROR, AND TERMINATE SUBROUTINES:


```

551 0317 CLR:    LET PSTAT = 0                                ; Int set to zero
552          ;
553          ; If not white then unack:
554          ;
555 0320          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
556 0321          LET FCLEAR = 1                                ; Clear flag
557 0322          GOSUB ACKK                                    ; Print message
558 0323          RETURN
559          ;
560 0324 ALM:    LET PSTAT = 1                                ; Int set to one
561          ;
562          ; If not white then unack:
563          ;
564 0327          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
565          ;
566          ; Turn audible alarm on if white or yellow:
567          ;
568 0329          IF AND(PTYPE,3) THEN FALMON = 1
569 0330          LET PDEV:DEV_TMPO = 2
570          ;
571 0331          LAMP ADI,SKDKEY,2                                ; Blink the LAMP
572 0332          GOSUB ACKK                                    ; Print message
573 0333          RETURN
574          ;
575 0334 ACK1:   LET PSTAT = 2                                ; Acknowledge alarm
576 0335          LET FACK = 1                                  ; Set ACK flag for C program
577          ;
578          ; ALARM TYPER STRINGS:
579          ;
580 0337 ACKK:   IF ILOOP = 0 THEN STRNG0 = "244A-P1 HRDWR INTRLK "
581          ;
582 0339          IF ILOOP = 2 THEN STRNG0 = "244A-P2 HRDWR INTRLK "
583          ;
584          IF ILOOP = 17 THEN STRNG0 = "244A PUMP PIT LEAK "
585          IF ILOOP = 18 THEN STRNG0 = "244A SER PIT RADN HI "
586          IF ILOOP = 19 THEN STRNG0 = "244A FLSH PIT PRES HI "
587          IF ILOOP = 20 THEN STRNG0 = "244BATCH PGM STARTING "
588          IF ILOOP = 21 THEN STRNG0 = "244BATCH WRONG UNIT "
589          IF ILOOP = 22 THEN STRNG0 = "244BATCH TERMINATING "
590          IF ILOOP = 31 THEN STRNG0 = "244A LIQUID TEMP HI "
591          ;
592          ; If acknowledging alarm, then:
593          ;
594 0348 AMSG:   IF (FACK) THEN
595          BEGIN
596          ;
597          ; If white, print it is acknowledged and reset the flag:
598          ;
599          IF PTYPE = 1 THEN PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNG0
600          LET FACK = 0
601          GOTO NXT
602          END
603          ;
604          ; If CLEAR flag is set, print it is clear and reset the flag:
605          ;
606 0353          IF (FCLEAR) THEN
607          BEGIN

```


244HVAC

```

1      ;      2 4 4 H V A C . S E Q      VERSION 2
2      ;
3      ;      Function:      Sequence program for Lamp function,
4      ;                      message display on Graphic number 341,
5      ;                      and printer messages.
6      ;
7      ;      Written By      : R. M. Kersey
8      ;      Date           : 05-31-92
9      ;      Facility       : 242-A Evaporator Bldg.
10     ;      Machine        : VAX 4104-A
11     ;      Op. System     : VAX/VMS 5.5-2AH
12     ;      D/3 Version    : 6.3-2 BASELINE
13     ;      Compiler       : SIC
14     ;      Build command  : @[320,20]244HVAC
15     ;
16     ;      UNIT = 2, SKDKEY = 18
17     ;
18     ;      Assoc. UPF:      [300,1]_SHORT0.____      SHORT0.LST
19     ;      Assoc. RPF:      [300,1]_FARM.____        FARM.LST
20     ;      SDB:            [300,1]PCMOASDB.____      PCMOASDB.CMD
21     ;
22     ;      POINTER      SDB VAR.      FUNCTION
23     ;      -----
24     ;      PSTAT      INTn      Current EPN alarm state is an integer tied to
25     ;                      graphic flashing string of value:
26     ;                      .      (n=0-100)      0 = previously not alarmed
27     ;                      .                      1 = previously alarmed, not ack
28     ;                      .                      2 = previously alarmed, is ack
29     ;                      .                      3 = alarm is inhibited
30     ;                      .                      4 = input is off scan
31     ;                      .                      5 = input is bad
32     ;
33     ;      PDIN      LEAKn      Special INT variable for multiple white alarms on an EPN
34     ;      PTYPEn      'EPN'      ---
35     ;                      alarm type 0 = none, 1 = white, 2 = yellow,
36     ;                      3 = yellow and yellow, 4 = red,
37     ;                      6 = yellow and red
38     ;      PREV      FRWn      Reverse acting (not fail safe) flag (1=set)
39     ;      PSKID      SKIDn      Used for analog alarms
40     ;      PPASS      FPAIn      Turn lamp off when alarm is ACKnowledged
41     ;      PALARM      FALMn      Analog alarm flag
42     ;      ---      FNING      Unused error flag
43     ;
44     ;      MODIFICATION LOG:
45     ;
46     ;      09-JUN-93      Added comments and baselined for      M. Teats
47     ;                      D/3 version 3.3. CCN# 93JUN01-01.
48     ;
49     ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
50     ;                      system. Rebase line document. Removed
51     ;                      obsolete Select() statements.
52     ;
53     ;      END MOD LOG
54     ;
55     ;      UNIT      HVC244
56     ;

```

```

56 0000      STAR = 8                      ; Start message
57 0001      SKDKEY = 18                   ; SKID Key
58 0002      ADI = 0                       ; Alarm destination index
59 0003      FSN0 = 0                      ; No alarm yet
60 0004      TIMEROFF STMRO                ; Alarm timer off
61 0005      TIMERCLR STMRO               ; Alarm timer clear
62 0006      STRNG0 = " "                 ; Blank 30 character string
63 0007      ERRN = 0                      ; Set error number to zero
64 0008      ERRL = 0                      ; Set error line to zero
65 0009      ASSIGN PDEV = 'HVAC244'       ; Alarm device
66 0010      FTRIP4 = 0                    ; FTRIPP4 = 1 when audible horn is to be turned on
67 0011      GETUNIT TRIP                  ; Which sequence unit is program running in
68 0012      IF TRIP != 2 THEN             ; Abort if running in wrong seq. unit
69          BEGIN
70          FSTART = 1
71          ILOOP = STAR + 1
72          GOSUB ACKK
73          FSTART = 1
74          PRINT ADI," >>>>> ATTEMPT TO RUN 244BATCH IN WRONG UNIT"
75          GOTO TERM
76      END
77      ; IF !SELECT() THEN GOTO AGAIN
78      ;
79      ; BEGIN INITIALIZING
80      ;
81 0019      SMSG0 = "INIT"
82 0020      INIT: FABORT = 0                ; If FABORT = 1 then abort program
83 0021      FSTART = 0                     ; Program start flag = 0
84 0022      LAMP ADI,SKDKEY,0             ; Turn off the annunciator key
85 0023      LSG0 = " "                    ; Clear the string variables
86 0024      LSG1 = " "
87 0025      LSG2 = " "
88 0026      LSG3 = " "
89 0027      LSG4 = " "
90 0028      PDEV:DEV TMO0 = 1               ; Set alarm device to non-alarm state
91 0029      IF AND(PDEV:DEV_ALMCT,16) THEN ; If alarm device is inhibited, remove the inhibit
92 0030      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
93 0031      IF AND(PDEV:DEV_ALMCT,32) THEN ; If alarm device is acknowledged, clear it
94 0032      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
95      ;
96      ; First, set all flags and integers:
97      ;
98      ;
99      ;
100      ;
101      ;
102      ;
103      ;
104      ;
105      ;
106      ;
107      ;
108      ;
109      ;
110      ;

```

EPN	TYPE	INT	FRV	LIMIT	DESCRIPTOR
PDSL-EF-2	2	0	1		244-A HEPA DP FAN FAIL EXH RADN
PDSL-EF-1	2	1	0		244-A HEPA DELTA P HIGH
RSH-A	4	2	0		244-A ANNULUS RADN HIGH
RSH-EXH-1	4	3	0		244-A EXH STAK RADN HIGH
QXS-EXH-1	2	4	0		244-A STACK MON SYS FAIL
RI-244ANN	4	5	-		244-A LIFT STA ANNULUS RADN
FI-244EXH	0	6	-		244-A EXHAUST STACK FLOW
RI-244EXH	4	7	-		244-A LIFT STA EXH STACK RADN
FI-244EXH	1	8	1	100 LO	244-A EXHAUST STACK FLOW

```

111      ; FI-244EXH      1      9      0      200 HI  244-A EXHAUST STACK FLOW
112      ;
113      0033      ASSIGN PDIN = 'PDSL-EF-2'      ; Assign pointer to first DIN
114      0034      ASSIGN PSTAT = INTO
115      0035      ASSIGN PTYPE = TYPE0
116      0036      ASSIGN PREV = FRV0
117      0037      ASSIGN PPASS = FPAS0
118      0038      ASSIGN PSKID = SKID0
119      0039      ASSIGN PALARM = FDIS0
120      ;
121      0040      FOR ILOOP = 0 TO 9
122      0041      PPASS = 0
123      0042      PREV = 0
124      0043      PTYPE = 2
125      0044      PSTAT = 0
126      0045      PSKID = 0
127      0046      IF ILOOP < 5 THEN
128      BEGIN
129      0047      IF (AND(PDIN:DIN_DFLG,3)) THEN      ; Check for alarms (must be red or yellow)
130      BEGIN
131      0048      PTYPE = 2      ; Default to yellow
132      0050      IF AND(PDIN:DIN_DFLG,1) THEN PREV = 1      ; Reverse acting
133      0052      IF AND(PDIN:DIN_DFLG,4) THEN PTYPE = 4      ; Set to critical alarm
134      END
135      0054      IF ILOOP < 4 THEN STEP PDIN BY 1      ; Point to next DIN
136      END
137      0056      IF ILOOP > 7 THEN PTYPE = 1
138      0057      SKP: STEP PSTAT BY 1
139      0058      STEP PTYPE BY 1
140      0059      STEP PREV BY 1
141      0060      STEP PPASS BY 1
142      0061      STEP PSKID BY 1
143      0062      STEP PALARM BY 1
144      0063      NEXT ILOOP
145      ;
146      0064      FRV0 = 1      ; PDSL-EF-2
147      0065      FRV1 = 0      ; PSDH-EF-1
148      0066      FRV2 = 0      ; RSH-A
149      0067      FRV3 = 0      ; RSH-EXH-1
150      0068      FRV4 = 0      ; QXS-EXH-1
151      0069      FRV8 = 1      ; FI-244EXH LO
152      ;
153      0070      TYPE5 = 4      ; RI-244ANM
154      0071      TYPE6 = 0      ; RI-244EXH
155      0072      TYPE7 = 4      ; RI-244EXH
156      ;
157      0073      A10 = 100.0      ; FI-244EXH LO
158      0074      A11 = 200.0      ; FI-244EXH HI
159      0075      ANALOG = 0
160      ;
161      ;
162      ;
163      ;
164      ;
165      ;

```

```

166 0076      ILOOP = STAR
167 0077      FSTART = 1
168 0078      GOSUB ACKK
169 0079      ON ERROR GOSUB ERRSUB
170      ;
171      ;
172      ;      B E G I N   P R O G R A M   L O O P
173 0080  AGAIN:  MARK STEP "AGAIN",0,0
174 0081      SNGO = "WAIT"
175      ;      WAITUNTIL SELECT() ! FABORT      ; Wait until micro is selected or program aborted
176      ;      IF (FABORT) THEN GOTO TERM      ; Terminate program if FABORT flag is set
177 0082      WAIT 1
178 0083      IF (FSNO) THEN      ; If alarm flag set
179 0084      IF STMRO > 60 THEN      ; If alarm flag active for 1 minute
180      BEGIN      ; Reset timer and turn off flag
181 0085      TIMEROFF STMRO
182 0086      TIMERCLR STMRO
183 0087      FSNO = 0
184 0088      ERNR = 0
185 0089      ERRL = 0
186      END
187      ;
188      ;
189      ;      If HVAC244 $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
190 0090      IF (PDEV:DEV_TMPO = 2) & AND(PDEV:DEV_ALMCT,32) THEN
191      BEGIN
192 0091      PDEV:DEV_TMPO = 1      ; Put device in ack state
193 0092      FTRIP4 = 1      ; Set for unit alarm ack, system to graphic
194 0093      ACK = 1      ; Set for unit alarm ack, graphic to system
195 0094      CRIT
196 0095      LAMP ADI,SKDKEY,1
197      END
198      ;
199      ;
200      ;      BEGIN DINs:
201 0096      ASSIGN PDIN = 'PDSL-EF-2'
202 0097      ASSIGN PTYPE = TYPEO
203 0098      ASSIGN PREV = FRVO
204 0099      ASSIGN PSTAT = INTO
205 0100      ASSIGN PPASS = FPASO
206 0101      CURENT = 0      ; Used to set LAMP status
207 0102      LAST = 0
208 0103      ANALOG = 0
209 0104      NKT = 0
210 0105      LMPVAR = 0
211 0106      BITVAL = 0
212      ;
213 0107  LOP:   FOR ILOOP = 0 TO 4
214 0108      IF ( PTYPE > 0 ) THEN      ; If some type of alarm
215      BEGIN
216 0110      IF AND(PDIN:DIN_ALMCT,16) THEN PSTAT = 3      ; If EPN inhibited
217 0112      IF AND(PDIN:DIN_ALMCT,4) THEN PSTAT = 5      ; If EPN is bad input
218 0114      IF !AND(PDIN:DIN_FLG,8) THEN PSTAT = 4      ; If EPN is disabled
219 0115      IF ( PSTAT < 3 ) THEN      ; If not inhibited
220      BEGIN      ;be2

```

```

221 0116          IF PSTAT = 1 THEN
222              BEGIN                                ;be3
223              IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
224 0120          IF (FTRIP4) THEN GOSUB ACK1
225              END                                  ;en3
226 0123          IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
227 0124          IF(PSTAT) THEN
228              BEGIN                                ;be4
229              IF PDIN:DIN_VAL != PREV THEN GOSUB CLR
230 0127          NXT = PSTAT
231 0129          IF (PPASS) & (PSTAT = 2) THEN NXT = 0
232 0130          LAST = OR(LAST,NXT)
233              END                                  ;en4
234              END                                  ;en2
235 0132          ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8)THEN PSTAT = 0
236              END                                  ;en1
237 0134          IF ILOOP < 4 THEN STEP PDIN BY 1
238 0135          STEP PTYPE BY 1
239 0136          STEP PREV BY 1
240 0137          STEP PSTAT BY 1
241 0138          STEP PPASS BY 1
242 0139          NEXT ILOOP
243
244          :
245          :
246          :
247          :
248          :
249          :
250          :
251          :
252          :
253          :
254          :
255          :
256          :
257          :
258          :
259          :
260          :
261          :
262          :
263          :
264          :
265          :
266          :
267          :
268          :
269          :
270          :
271          :
272          :
273          :
274          :
275          :

```

END DIMS:

BEGIN ANALOG RED AND YELLOWS:

```

248 0140          ASSIGN PSKID = SKID5
249 0141          ASSIGN PTYPE = TYPE5
250 0142          ASSIGN PREV = FRV5
251 0143          ASSIGN PAI = 'R1-244ANN'
252 0144          FOR JLOOP = 5 TO 7
253 0145          IF (PTYPE) THEN                                ; If some alarm exists
254              BEGIN                                ;be1
255 0146          BITVAL = PAI:A1_ALMCT
256 0147          IF !AND(BITVAL,16) THEN
257              BEGIN                                ;be2
258 0148          IF AND(BITVAL,15) THEN                                ; If alarm exists and alarm is current
259              BEGIN                                ;be3
260 0150          IF AND(BITVAL,32) THEN NXT = 2                                ; If alarm is ack
261              ELSE
262              BEGIN                                ;be4
263 0151          IF PSKID != BITVAL THEN
264              BEGIN                                ;be5
265 0152          PSKID = BITVAL
266 0153          PDEV:DEV_TMPO = 2
267 0155          IF BITVAL < 3 THEN FALMON = 1
268 0156          NXT = 1
269 0157          LAMP AD1,SKDKEY,2
270 0158          CRIT
271              END                                  ;en5
272 0159          ELSE NXT = 1
273              END                                  ;en4
274              END                                  ;en3
275          ELSE                                ; Else there is no alarm

```



```

276          BEGIN
277 0160      NXT = 0
278 0161      PSKID = 0
279          END
280          END ;en2
281      ELSE ; Else alarm is inhibited
282          BEGIN
283 0162      NXT = 0
284 0163      PSKID = 0
285          END
286 0164      ANALOG = OR(ANALOG,NXT)
287          END ;en1
288 0165      STEP PSKID BY 1
289 0166      STEP PTYPE BY 1
290 0167      STEP PREV BY 1
291 0169      IF JLOOP < 3 THEN STEP PAI BY 1
292 0170      NEXT JLOOP
293      ;
294      ;
295      ;
296      ;
297      ;
298 0171  WHT:  WHITE = 0
299 0172      SNGO = "RUN"
300 0173      ASSIGN PALARM = A10
301 0174      ASSIGN PSTAT = INT8
302 0175      ASSIGN PTYPE = TYPE8
303 0176      ASSIGN PREV = FRV8
304 0177      ASSIGN PPASS = FPAS8
305 0178      ILOOP = 8
306 0179      ASSIGN PAI = 'FI-244EXH'
307 0180      GOSUB ANLG
308 0181      GOSUB ANLG
309 0182      CURENT = OR(CURENT,ANALOG)
310 0183      MARK STEP "BELOW",0,0
311      ;
312      ; Use LEAK0 with string packet 78 in graphic:
313      ;
314 0185      IF ((INT9)&INT9<3) THEN LEAK0 = INT9 + 4
315 0187      ELSE IF((INT8) & (INT8 < 3)) THEN LEAK0 = INT8 + 2
316 0189      ELSE IF INT9 = 3 | INT8 = 3 THEN LEAK0 = 9
317 0190      ELSE LEAK0 = 0
318      ;
319      ;
320      ;
321 0191      FTRIP4 = 0
322 0193      IF CURENT > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
323 0194      CURENT = OR(LAST,WHITE)
324 0195      CURENT = OR(CURENT,ANALOG)
325 0197      IF AND(CURENT,1)THEN LMPVAR = 2
326 0199      ELSE IF AND(CURENT,2)THEN LMPVAR = 1
327 0200      ELSE LMPVAR = 0
328 0201      LAMP ADI,SKDKEY,LMPVAR
329 0202      MARK STEP "BOTTOM",0,0
330 0203      NOCRIT

```

```

331 0204      GOTO AGAIN
332      ;
333      ;      E N D   P R O G R A M   L O O P
334      ;
335      ;      BEGIN ANALOG WHITE ALARM SUBROUTINE:
336      ;
337 0205  ANLG:  IF PTYPE = 1 THEN                      ; If some alarm
338              BEGIN                                ;be0
339              BITVAL = PAI:AI_ALMCT
340 0207      IF !(AND(BITVAL,16) & (PTYPE)) THEN      ; If not inhibited and alarm enabled
341              BEGIN                                ;be1
342 0209      IF PSTAT = 3 THEN PSTAT = 0
343 0210      IF (PREV) THEN                          ; If rev acting, LOLO, or LO
344              BEGIN                                ;be2
345 0211      IF !(PSTAT) THEN                          ; If not previously alarmed
346              BEGIN                                ;be3
347 0213      IF PAI:AI_INVL <= PALARM THEN GOSUB ALM
348              END                                  ;en3
349      ELSE
350              BEGIN                                ;be4
351 0214      IF PSTAT = 1 THEN                          ; If previously alarmed
352              BEGIN                                ;be5
353 0216      IF (FTRIP4) THEN GOSUB ACK1                ; If ACK flag set, then ack it
354 0219      IF PTYPE !=1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
355              END                                  ;en5
356 0221      IF PAI:AI_INVL >= PALARM + PAI:AI_DEAD THEN GOSUB CLR
357              END                                  ;en4
358              END                                  ;en2
359      ELSE
360              BEGIN                                ;be2.1
361 0222      IF PSTAT = 0 THEN
362              BEGIN
363 0224      IF PAI:AI_INVL >= PALARM THEN GOSUB ALM
364              END
365      ELSE
366              BEGIN
367 0225      IF PSTAT = 1 THEN
368              BEGIN
369 0227      IF (FTRIP4) THEN GOSUB ACK1
370 0230      IF PTYPE != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
371              END
372 0232      IF PAI:AI_INVL <= PALARM - PAI:AI_DEAD THEN GOSUB CLR
373              END
374              END                                  ;en2.1
375 0234      IF PSTAT = 3 THEN PSTAT = 0                ; If INHA then there is no alarm
376 0236      IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
377 0237      ELSE NXT = PSTAT
378 0238      NXT = PSTAT
379      END
380      ELSE
381      BEGIN
382 0239      PSTAT = 3
383 0240      NXT = 0
384      END
385 0241      WHITE = OR(WHITE,NXT)

```

```

386      END
387 0242      ELSE PSTAT = 0
388 0243      STEP PALARM BY 1
389 0244      STEP PSTAT BY 1
390 0245      STEP PTYPE BY 1
391 0246      STEP PREV BY 1
392 0247      STEP PPASS BY 1
393 0248      ILOOP = ILOOP + 1
394 0249      RETURN
395      ;
396      ;      END ANALOG WHITE ALARM SUBROUTINE
397      ;
398      ;      BEGIN CLEAR, ALARM, ACKNOWLEDGE, MESSAGE, ERROR, AND TERMINATE SUBROUTINES:
399      ;
400 0250 CLR:   PSTAT = 0 ; Int set to zero
401 0253      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
402 0254      FCLEAR = 1 ; Clear flag
403 0256      IF ILOOP = 9 THEN INT8 = 0 ; Reset the other white alarm
404 0257      GOSUB ACKK ; Print message
405 0258      RETURN
406      ;
407 0259 ALM:   PSTAT = 1 ; Int set to one
408      ;
409      ;      If not white then unack:
410      ;
411 0262      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
412      ;
413      ;      Turn audible alarm on if white or yellow:
414      ;
415 0264      IF AND(PTYPE,3) THEN FALMON = 1
416 0265      PDEV:DEV_TMD0 = 2
417 0266      CRIT
418 0267      LAMP ADI,SKDKEY,2 ; Blink the LAMP
419 0268      GOSUB ACKK ; Print message
420 0269      RETURN
421      ;
422 0270 ACK1:  PSTAT = 2 ; Ack the alarm
423 0271      FACK = 1 ; Set ACK flag for C program
424      ;
425      ;
426      ;      ALARM TYPER STRINGS:
427      ;
428 0273 ACKK:  IF ILOOP = 0 THEN STRNG0 = "DP FAN/FAIL EXH RADN "
429 0275      IF ILOOP = 1 THEN STRNG0 = "244-A HEPA DLTA P H1 "
430 0277      IF ILOOP = 2 THEN STRNG0 = "244-A ANN EXH RAD H1 "
431 0279      IF ILOOP = 3 THEN STRNG0 = "244-A EXH STK RAD H1 "
432 0281      IF ILOOP = 4 THEN STRNG0 = "244-A STK MN SYS FAIL "
433 0283      IF ILOOP = 8 THEN STRNG0 = "244-A EXH STK FLO LO "
434 0285      IF ILOOP = 9 THEN STRNG0 = "244-A EXH STK FLO HI "
435 0287      IF ILOOP = 20 THEN STRNG0 = "START PGM 244HVAC.SEO "
436 0289      IF ILOOP = 21 THEN STRNG0 = "WRONG UNIT 244HVAC.SEO"
437 0291      IF ILOOP = 22 THEN STRNG0 = "244HVAC TERMINATED "
438      ;
439      ;      If acknowledging the alarm, then:
440 0292 AMSG:  IF (FACK) THEN

```

```

441 BEGIN
442 ;
443 ; If white, print it is ack and reset the flag:
444 ;
445 0293 PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
446 0294 FACK = 0
447 0295 GOTO NXT
448 END
449
450 ; If CLEAR flag is set, print it is clear and reset the flag:
451 ;
452 0296 IF (FCLEAR) THEN
453 BEGIN
454 0297 PRINT ADI,"%p ALARM CLEAR",STRNGO
455 0298 FCLEAR = 0
456 0299 GOTO NXT
457 END
458 REMAIN= TIMSTR(SSNGO)
459 ; IF REMAIN != 8 THEN SSNGO = "TIMEFALL"
460 REMAIN= APPENDSTRNGI,STRNGO,SSNGO)
461 ; IF REMAIN < 0 THEN STRNGI = "APPEND FAILED!"
462 ;
463 ; If white alarm, print:
464 ;
465 0301 IF PTYPE = 1 THEN PRINT ADI,"WHITE ALARM %p",STRNGI
466 ;
467 ; If STARTUP, then print startup message:
468 ;
469 0302 NXT: IF FSTART = 1 THEN
470 BEGIN
471 0303 FSTART = 0
472 0304 PRINT ADI,"%p",STRNGO
473 END
474 0305 RETURN
475 ;
476 ; If ERROR - print message, start Btimer, and set flag:
477 ;
478 0307 ERRSUB: IF(ERNUM() = -187) THEN ERKSET
479 ELSE
480 BEGIN
481 IF ERN != ERNUM() THEN
482 BEGIN
483 0309 TIMRON STRMO
484 0310 TIMRCLR STRMO
485 0311 FSNO = 1
486 0312 ERN = ERNUM()
487 0313 ERRL = ERRLIN()
488 0314 PRINT ADI,"ERROR # %i AT LINE %i",ERN,ERRL
489 0315 ERCLR
490 END
491 END
492 GOTO AGAIN
493 ;
494 ; If FABORT = 1, then abort the program:
495 ;

```

```

496      ; TERM:  FABORT = 0
497      ; TERM:  STRINGD = "244BATCH TERMINATED"
498      ; TERM:  LOOP = STAR + 2
499      ; TERM:  FIRM = "FIRM"
500      ; TERM:  SSGCD = "SSGCD"
501      ; TERM:  GOSUB ACK
502      END

Total Errors Detected 0      Total Warnings 0
Sequence Program Size = 6048 Bytes, 378 Pages
22 Forward Reference(s)

```

WSTXFR

```

1      ; W S T X F R . S E Q                                VERSION 4
2      ;
3      ;
4      ; Function:      Sequence program for Lamp function,
5      ;                message display on Graphic number 344,
6      ;                and printer messages.
7      ;
8      ; Written By    : R. M. Kersey
9      ; Date         : 05-31-92
10     ; Facility      : 242-A Evaporator Bldg.
11     ; Machine       : VAX 4104-A
12     ; Op. System    : VAX/VMS 5.5-2AH
13     ; D/3 Version   : 6.3-3
14     ; Compiler      : SIC
15     ; Build command : @[320,20]WSTXFR
16     ;
17     ; UNIT = 3, SKDKEY = 16
18     ;
19     ; Assoc. UPF:    [300,1] SHORT0.____   SHORT0.LST
20     ; Assoc. RPF:    [300,1] FARM.____     FARM.LST
21     ; SDB:           [300,1] PCMOASDB.____  PCMOASDB.CMD
22     ;
23     ; POINTER      SDB VAR.      FUNCTION
24     ; -----
25     ; PSTAT        INTn          Current EPN alarm state is an integer tied to
26     ;                graphinc flashing string of value:
27     ;                (n=0-100)    0 = previously not alarmed
28     ;                .            1 = previously alarmed, not ack
29     ;                .            2 = previously alarmed, is ack
30     ;                .            3 = alarm is inhibited
31     ;                .            4 = input is off scan
32     ;                .            5 = input is bad
33     ;                LEAKn        Special INT variable for multiple white alarms on an EPN
34     ;                'EPN'        ---
35     ;                PTYPE        alarm type 0 = none, 1 = white, 2 = yellow,
36     ;                .            3 = yellow and yellow, 4 = red,
37     ;                .            6 = yellow and red
38     ;                PREV        FRVn        Reverse acting (not fail safe) flag (1=set)
39     ;                PSKID        SKIDn       Used for analog alarms
40     ;                PPASS        FPASn       Turn lamp off when alarm is ACKnowledged
41     ;                PALARM        FALMn       Analog alarm flag
42     ;                ---          FNING       Unused error flag
43     ;
44     ; MODIFICATION LOG:
45     ;
46     ; 04-AUG-92      Set FRV9 and FRV10 to 0 for Fail Safe          M. Teats
47     ;                per ECN 186678.
48     ;
49     ; 30-MAR-93      Modified for redundancy.                      M. Teats
50     ;
51     ; 09-JUN-93      Added comments, CCN# 93JUN01-01.             M. Teats
52     ;
53     ; -mct1+ 13-JUL-93 Add LDS216A40 PCM1 DIN section, and CLR2,   M. Teats
54     ;                ALM2, ACK2 subroutines per CCN 93JUL13-01.
55     ;                Baselined for D/3 version 3.3.

```

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

07:23:08 24-Oct-95

Page 2

```

56      ;
57      ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
58      ;      system. Rebase line document. Removed
59      ;      obsolete Select() statements.
60      ;
61      ;      25-JUL-95      Revised program to remove EPN LDS216A40      J. Berger
62      ;      from program. See ECN 623935 and package
63      ;      EE-95-00006. (JFB01)
64      ;
65      ;      20-OCT-95      Revised program to remove EPNs that are      J. Berger
66      ;      no longer used. See ECN 627411 and package
67      ;      EE-94-00929. (JFB02)
68      ;
69      ;      END MOD LOG
70      ;
71      UNIT      WASTE      ; Is in PCMO
72      COMMON      WHOLE1      ; Is in PCM1, -mct1+
73      ;
74      ;      LDS1 == 'LDS216A40'      ; -mct1+ (JFB01)
75      ;
76      0000      NOP
77      0001      STAR = 20      ; Start message
78      0002      SKDKEY = 16      ; SKID Key
79      0003      ADI = 0      ; Alarm Destination Index
80      0004      FSWO = 0      ; No alarm yet
81      0005      TIMEROFF STMW0      ; Alarm timer off
82      0006      TIMERCLR STMW0      ; Alarm timer clear
83      0007      STRNG0 = " "      ; Blank 30 character string
84      0008      ERRN = 0      ; Set error number to zero
85      0009      ERRL = 0      ; Set error line to zero
86      0010      ASSIGN PDEV = 'WSTXFR'      ; Alarm Device
87      0011      FTRIP4 = 0      ; FTRIP4 = 1 when audible horn is to be turned on
88      0012      GETUNIT TRIP      ; Which sequence unit is program running in
89      0013      IF TRIP != 3 THEN      ; Abort if running in the wrong seq. unit
90      BEGIN
91      0014      FSTART = 1
92      0015      ILOOP = STAR + 1
93      0016      FSTART = 1
94      0017      GOSUB ACKK
95      0018      PRINT ADI," >>>> ATTEMPT TO RUN WSTXFR IN WRONG UNIT"
96      0019      GOTO TERM
97      END
98      ;
99      ;      BEGIN INITIALIZING
100     ;
101     0020     SSWGO = "INIT"
102     0021     INIT: LET FABORT = 0      ; If FABORT = 1 then abort program
103     0022     LET FSTART = 0      ; Start Flag = 0
104     0023     LAMP ADI,SKDKEY,0      ; Turn off the annunciator key
105     0024     LET LSG0 = " "      ; Clear the string variables
106     0025     LET LSG1 = " "
107     0026     LET LSG2 = " "
108     0027     LET LSG3 = " "
109     0028     LET LSG4 = " "
110     0029     PDEV:DEV TMPO = 1      ; Set alarm device to non-alarm state
111     0030     IF AND(PDEV:DEV_ALMCT,16) THEN      ; If alarm device is inhibited, remove the inhibit

```



```

111 0031      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
112 0032      IF AND(PDEV:DEV_ALMCT,32) THEN      ; If alarm device is acknowledged, clear it
113 0033      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
114      ;
115      ; First, set all flags and integers:
116      ;
117      ;
118      ;
119      ;
120      ;
121      ;
122      ;
123      ;
124      ;
125      ;
126      ;
127      ;
128      ;
129      ;
130      ;
131      ;
132      ;
133      ;
134      ;
135      ;
136      ;
137      ;
138 0034      ASSIGN PDIN = 'LDS-AR151'      ; Assign pointer to first DIN
139 0035      ASSIGN PSTAT = INTO
140 0036      ASSIGN PTYPE = TYPE0
141 0037      ASSIGN PREV = FRV0
142 0038      ASSIGN PPASS = FPAS0
143      ;
144 0039      FOR ILOOP = 0 TO 17
145 0040      LET PPASS = 0
146 0041      LET PREV = 0
147 0042      LET PTYPE = 0
148 0043      LET PSTAT = 0
149 0044      IF ILOOP = 4 THEN      ; JFB02
150      BEGIN      ; JFB02
151      STEP PDIN BY 1      ; JFB02
152 0046      GOTO SKP      ; JFB02
153      END      ; JFB02
154 0047      IF ILOOP = 5 THEN      ; JFB02
155      BEGIN      ; JFB02
156 0048      STEP PDIN BY 1      ; JFB02
157 0049      GOTO SKP      ; JFB02
158      END      ; JFB02
159 0050      IF ILOOP = 6 THEN      ; JFB02
160      BEGIN      ; JFB02
161      STEP PDIN BY 1      ; JFB02
162 0052      GOTO SKP      ; JFB02
163      END      ; JFB02
164 0053      IF ILOOP = 7 THEN      ; JFB02
165      BEGIN      ; JFB02

```

```

166 0054      STEP PDIN BY 1          ; JFB02
167 0055      GOTO SKP                ; JFB02
168          END                      ; JFB02
169 0056      IF ILOOP = 11 THEN      ; JFB02
170          BEGIN                    ; JFB02
171 0057      STEP PDIN BY 1          ; JFB02
172 0058      GOTO SKP                ; JFB02
173          END                      ; JFB02
174 0059      IF ILOOP = 12 THEN      ; JFB02
175          BEGIN                    ; JFB02
176 0060      STEP PDIN BY 1          ; JFB02
177 0061      GOTO SKP                ; JFB02
178          END                      ; JFB02
179 0062      IF ILOOP = 13 THEN      ; JFB02
180          BEGIN                    ; JFB02
181 0063      STEP PDIN BY 1          ; JFB02
182 0064      GOTO SKP                ; JFB02
183          END                      ; JFB02
184 0065      IF ILOOP = 14 THEN      ; JFB02
185          BEGIN                    ; JFB02
186 0066      STEP PDIN BY 1          ; JFB02
187 0067      GOTO SKP                ; JFB02
188          END                      ; JFB02
189 0068      IF (AND(PDIN:DIN_DFLG,3)) THEN ; Check for alarms (must be red or yellow)
190          BEGIN                    ;
191 0069          LET PTYPE = 2        ; Default to yellow
192 0071          IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1 ; Reverse acting
193 0073          IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4 ; Set to critical alarm
194          END                      ;
195 0075      IF ILOOP < 17 THEN STEP PDIN BY 1 ; Point to next DIN
196 0076      SKP: STEP PSTAT BY 1
197 0077          STEP PTYPE BY 1
198 0078          STEP PREV BY 1
199 0079          STEP PPASS BY 1
200 0080          NEXT ILOOP
201          ;
202          ; LET TYPE4 = 0          ; ZSER15212
203          ; LET TYPE5 = 0          ; ZSER15211
204          ; LET TYPE6 = 0          ; PSER15212
205          ; LET TYPE7 = 0          ; PSER15211
206          ; LET TYPE11 = 0         ; ZSER15312
207          ; LET TYPE12 = 0         ; ZSER15311
208          ; LET TYPE13 = 0         ; PSER15312
209          ; LET TYPE14 = 0         ; PSER15311
210          ; LET TYPE18 = 2         ; LDS216A40, -mct1+ (JFB01)
211          ;
212 0081      LET FRV0 = 0            ; LDS-AR151
213 0082      LET FRV1 = 0            ; LDS-ER151
214 0083      LET FRV2 = 0            ; PSHSMV229
215 0084      LET FRV3 = 0            ; LDS-ER152
216 0085      LET FRV8 = 0            ; LDS-ER153
217 0086      LET FRV9 = 0            ; PSHPSN232
218 0087      LET FRV10 = 0           ; PSHPSN234
219 0088      LET FRV15 = 0           ; LDS-AX155
220 0089      LET FRV16 = 0           ; LDS-DR713

```

```

221 0090      LET FRV17 = 0          ; LDS-DR717
222      ;      LET FRV18 = 0          ; LDS216A40, -mct1+ (JFB01)
223      ;
224      ;      LET INT18 = 0          ; LDS216A40, -mct1+ (JFB01)
225      ;      LET FPAS18 = 0        ; LDS216A40, -mct1+ (JFB01)
226      ;
227      ;      END INITIALIZING
228      ;
229      ;      SEND PROGRAM START MESSAGE
230      ;
231 0091      ANALOG = 0
232 0092      ILOOP = STAR
233 0093      FSTART = 1
234 0094      GOSUB ACKK
235 0095      ON ERROR GOSUB ERRSUB
236      ;
237      ;      BEGIN PROGRAM LOOP
238      ;
239 0096  AGAIN:  MARK STEP "WTUNTL",0,0
240      ;      WAITUNTIL SELECT() ! FABORT          ; Wait until micro is selected or program aborted
241      ;      IF (FABORT) THEN GOTO TERM          ; Terminate program if FABORT flag is set
242      ;      MARK STEP "RNNING",0,0
243 0097      WAIT 1
244 0098      IF (FSNO) THEN          ; If alarm flag set
245 0099      IF STMRO > 60 THEN      ; If alarm flag active for 1 minute
246      BEGIN                      ; Reset timer and turn off flag
247 0101      TIMEROFF STMRO
248 0102      TIMERCLR STMRO
249 0103      LET FSNO = 0
250 0104      LET ERRN = 0
251 0105      LET ERRL = 0
252      END
253      ;
254      ;      If WSTXFR $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
255      ;
256 0106      IF (PDEV:DEV_TMPO = 2 ) & AND(PDEV:DEV_ALMCT,32) THEN
257      BEGIN
258 0107      PDEV:DEV_TMPO = 1          ; Put device in acknowledge state
259 0108      LET FTRIP4 = 1          ; Set for unit alarm acknowledgement, system to graphic
260 0109      LET ACK = 1              ; Set for unit alarm acknowledgement, graphic to system
261      ;      CRIT
262 0110      LAMP ADI,SKDKEY,1
263      END
264      ;
265      ;      BEGIN DINs:
266      ;
267 0111      ASSIGN PDIN = 'LDS-AR151'
268 0112      ASSIGN PSTAT = INTO
269 0113      ASSIGN PTYPE = TYPEO
270 0114      ASSIGN PREV = FRVO
271 0115      ASSIGN PPASS = FPASO
272 0116      CURRENT = 0              ; Used to set LAMP status
273 0117      LAST = 0
274 0118      ANALOG = 0
275 0119      NXT = 0

```

```

276 0120      WHITE = 0
277 0121      LMPVAR = 0
278 0122      BITVAL = 0
279
280 0123      ;
281      LOP:  ;
282      IF !SELECT() THEN GOTO AGAIN
283      IF ILOOP = 4 THEN GOTO LOP1      ; JFB02
284      IF ILOOP = 5 THEN GOTO LOP1      ; JFB02
285      IF ILOOP = 6 THEN GOTO LOP1      ; JFB02
286      IF ILOOP = 7 THEN GOTO LOP1      ; JFB02
287      IF ILOOP = 11 THEN GOTO LOP1     ; JFB02
288      IF ILOOP = 12 THEN GOTO LOP1     ; JFB02
289      IF ILOOP = 13 THEN GOTO LOP1     ; JFB02
290      IF ILOOP = 14 THEN GOTO LOP1     ; JFB02
291      IF ( PTYPE > 0 ) THEN             ; If some type of alarm
292      BEGIN                             ; be1
293      IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3 ; If EPW inhibited
294      IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5 ; If EPW is bad input
295      IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4 ; If EPW is disabled
296      IF ( PSTAT < 3 ) THEN             ; If not inhibited
297      BEGIN                             ; be2
298      IF PSTAT = 1 THEN                 ; be3
299      BEGIN
300      IF PTYPE I= 1 THEN
301      IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
302      IF (FTRIP4) THEN GOSUB ACK1
303      END                               ; en3
304      IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
305      IF (PSTAT) THEN
306      BEGIN                             ; be4
307      IF PDIN:DIN_VAL I= PREV THEN GOSUB CLR
308      NXT = PSTAT
309      IF (PPASS) & (PSTAT = 2) THEN NXT = 0
310      LAST = OR(LAST,NXT)
311      END                               ; en4
312      ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0 ; en2
313      END                               ; en1
314      IF ILOOP < 17 THEN STEP PDIN BY 1
315      STEP PSTAT BY 1
316      STEP PTYPE BY 1
317      STEP PREV BY 1
318      STEP PPASS BY 1
319      NEXT ILOOP
320
321      ;
322      ;      Begin PCM1 DIN (LDS216A40) -mct1+: (JFB01)
323
324      ;      ILOOP = 18
325      IF !SELECT() THEN GOTO AGAIN
326      IF ( TYPE18 > 0 ) THEN             ; If some type of alarm
327      BEGIN                             ; be1
328      IF AND('LDS1':DIN_ALMCT,16) THEN INT18 = 3 ; If EPW inhibited
329      IF (ERRNUM() = -187) THEN GOTO WHT ; Skip LDS216A40 if PCM1 micros down
330      IF AND('LDS1':DIN_ALMCT,4) THEN LET INT18 = 5 ; If EPW is bad input
331      IF !AND('LDS1':DIN_FLG,8) THEN LET INT18 = 4 ; If EPW is disabled

```

```

331      ;      IF ( INT18 < 3 ) THEN      ; If not inhibited
332      ;      BEGIN                      ; be2
333      ;      IF INT18 = 1 THEN
334      ;      BEGIN                      ; be3
335      ;      IF TYPE18 != 1 THEN
336      ;      IF AND('LDS1':DIN_ALMCT,32) THEN GOSUB ACK2
337      ;      IF (FTRIP4) THEN GOSUB ACK2
338      ;      END                      ; en3
339      ;      IF INT18 = 0 THEN IF('LDS1':DIN_VAL = FRV18) THEN GOSUB ALM2
340      ;      IF (INT18) THEN
341      ;      BEGIN                      ; be4
342      ;      IF 'LDS1':DIN_VAL != FRV18 THEN GOSUB CLR2
343      ;      NXT = INT18
344      ;      IF (FPAS18) & (INT18 = 2) THEN NXT = 0
345      ;      LAST = OR(LAST,NXT)
346      ;      END                      ; en4
347      ;      END                      ; en2
348      ;      ELSE IF !AND('LDS1':DIN_ALMCT,16)&!AND('LDS1':DIN_ALMCT,4)&AND('LDS1':DIN_FLG,8) THEN LET INT18 = 0
349      ;      END                      ; en1
350      ;
351      ;      END PCM1 DIN
352      ;
353      ;      END DINs
354      ;
355      0173  WHT:  WHITE = 0
356      0174      SSWGO = "RUN"
357      0175      MARK STEP "BELOW",0,0
358      0176      LET FTRIP4 = 0
359      0178      IF LAST > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
360      0179      CURENT = OR(LAST,WHITE)
361      0180      CURENT = OR(CURENT,ANALOG)
362      0182      IF AND(CURENT,1) THEN LMPVAR = 2
363      0184      ELSE IF AND(CURENT,2) THEN LMPVAR = 1
364      0185      ELSE LMPVAR = 0
365      0186      LAMP ADI,SKDKEY,LMPVAR
366      0187      MARK STEP "BOTTOM",0,0
367      ;      NOCRIT
368      0188      GOTO AGAIN
369      ;
370      ;      E N D   P R O G R A M   L O O P
371      ;
372      ;      BEGIN CLEAR, ALARM, ACKNOWLEDGE, MESSAGE, ERROR, AND TERMINATE SUBROUTINES:
373      ;
374      0189  CLR:  LET PSTAT = 0      ; INT set to zero
375      ;
376      ;      ; If not white then unack:
377      ;
378      0192      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
379      0193      LET FCLEAR = 1      ; Clear flag
380      0194      GOSUB ACKK
381      0195      RETURN
382      ;
383      ;      CLR2:  LET INT18 = 0      ; PCM1 DIN, -mct1+ (JFB01)
384      ;
385      ;      ; If not white, then unack:

```

```

386      ;
387      ; IF TYPE18 != 1 THEN IF AND('LDS1':DIN_ALMCT,32)THEN 'LDS1':DIN_ALMCT = XOR('LDS1':DIN_ALMCT,32)
388      LET FCLEAR = 1
389      ;
390      GOSUB ACKK
391      RETURN
392      ;
393      0196 ALM: LET PSTAT = 1 ; INT set to one
394      ;
395      ; If not white then unack:
396      0199 IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
397      ;
398      ; Turn audible alarm on if white or yellow:
399      ;
400      0201 IF AND(PTYPE,3) THEN FALMON = 1
401      0202 LET PDEV:DEV_TMP0 = 2
402      ;
403      0203 LAMP ADI,SKDKEY,2 ; Blink the LAMP
404      0204 GOSUB ACKK ; Print message
405      0205 RETURN
406      ;
407      ; ALM2: LET INT18 = 1 ; PCM1 DIN, -mct1+ (JFB01)
408      ;
409      ; If not white, then unack:
410      ;
411      ; IF TYPE18 != 1 THEN IF AND('LDS1':DIN_ALMCT,32)THEN 'LDS1':DIN_ALMCT = XOR('LDS1':DIN_ALMCT,32)
412      ;
413      ; Turn audible alarm on if white or yellow:
414      ;
415      ; FALMON = 1
416      ; LET 'WSTXFR':DEV_TMP0 = 2
417      ;
418      ; CRIT
419      ; LAMP ADI,SKDKEY,2
420      ; GOSUB ACKK
421      ; RETURN
422      0206 ACK1: LET PSTAT = 2 ; Acknowledge alarm
423      0207 LET FACK = 1 ; Set ACK flag for C program
424      ;
425      ; ACK2: IF ILOOP = 18 THEN ; PCM1 DIN, -mct1+ (JFB01)
426      ;
427      ; BEGIN
428      ; LET INT18 = 2
429      ; LET FACK = 1
430      ;
431      ; END
432      ;
433      ; ALARM TYPER STRINGS:
434      ;
435      0209 ACKK: IF ILOOP = 0 THEN STRNG0 = "151-AR DIVISION BOX LEAK"
436      0211 IF ILOOP = 1 THEN STRNG0 = "1-1-ER DIVISION BOX LEAK"
437      0213 IF ILOOP = 2 THEN STRNG0 = "151-ER LINE V229 LEAK"
438      0215 IF ILOOP = 3 THEN STRNG0 = "152-ER DIVISION BOX LEAK"
439      0217 IF ILOOP = 8 THEN STRNG0 = "153-ER DIVISION BOX LEAK"
440      0219 IF ILOOP = 9 THEN STRNG0 = "153-ER LINE V232 LEAK"
441      0221 IF ILOOP = 10 THEN STRNG0 = "153-ER LINE V234 LEAK"
442      0223 IF ILOOP = 15 THEN STRNG0 = "155-AX DIVISION BOX LEAK"

```

```

441 0235 IF LOOP = 16 THEN STRNG0 = " DRAIN DR-713 LINE LEAK"
442 0237 ;
443 IF LOOP = 17 THEN STRNG0 = " DRAIN DR-717 LINE LEAK"
444 ;
445 0239 IF LOOP = 18 THEN STRNG0 = "LD5210440 LEAK DET " ;--mct1+ (JF801)
446 0231 IF LOOP = 20 THEN STRNG0 = "NSTXFR PGM STARTING "
447 0233 IF LOOP = 21 THEN STRNG0 = "NSTXFR PROGM UNIT "
448 IF LOOP = 22 THEN STRNG0 = "NSTXFR TERMINATING "
449 ;
450 ; If acknowledging alarm, then:
451 ANMSG: IF (FACK) THEN
452 BEGIN
453 ; If white, print it is acknowledged and reset the flag
454 ;
455 IF PTYPE = 1 THEN PRINT AD1,"%p ALARM ACKNOWLEDGE",STRNG0
456 LET FACK = 0
457 GOTO NXT
458 END
459 ;
460 ; If CLEAR flag is set, print it is clear and reset the flag:
461 ;
462 0239 IF (FCLEAR) THEN
463 BEGIN
464 PRINT AD1,"%p ALARM CLEAR",STRNG0
465 LET FCLEAR = 0
466 GOTO NXT
467 END
468 ;
469 ; INO = TINSTR(SSNGO)
470 IF INO I= 8 THEN SSNGO = "TIMEFAIL"
471 INO = APPEND(STRNG1,STRNGO,SSNGO)
472 IF INO < 0 THEN STRNG1 = "APPEND FAILED!!"
473 ;
474 ; If white alarm, print:
475 ;
476 0244 IF PTYPE = 1 THEN PRINT AD1,"WHITE ALARM %p",STRNG0
477 ;
478 ; If STARTUP, then print startup message:
479 ;
480 0245 IF FSTART = 1 THEN
481 BEGIN
482 LET FSTART = 0
483 PRINT AD1,"%p",STRNG0
484 END
485 ;
486 ; If ERROR - print message, start timer, and set flag:
487 ;
488 0249 ERRSUB: IF ERRN I= ERRNUM() THEN
489 BEGIN
490 TIMERON STRNO
491 TIMERCLR STRNO
492 LET FSNO = 1
493 LET ERRN = ERRNUM()
494 LET ERRL = ERRLIN()
495 PRINT AD1,"ERROR # %i AT LINE %i",errn,errl

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

07:23:08 24-Oct-95

Page 10

```

496             END
497 0256         ERSCLR
498 0257         GOTO AGAIN
499
500             ; If FABORT = 1, then abort program:
501             ;
502             ; TERM: LET FABORT = 0
503 0258         TERM: LET STRNGO = "WSTXFR TERMINATED  "
504 0259         LET ILOOP = STAR + 2
505 0260         LET FSTART = 1
506 0261         LET SSGO = "TERM"
507 0262         GOSUB ACKK
508             END

```

```

Total Errors Detected      0      Total Warnings      0
Sequence Program Size = 5008 Bytes, 313 Pages
      28 Forward Reference(s)

```


350BATCH

```

1      : 350BATCH.SEQ                                VERSION 3
2      :
3      : Function: Sequence program for Lamp function,
4      :           message display on Graphic number 343,
5      :           and printer messages.
6      :
7      : Written By : R. M. Kersey
8      : Date      : 05-31-92
9      : Facility  : 242-A Evaporator Bldg.
10     : Machine   : VAX 4104-A
11     : Op. System : VAX/VMS 5.5-2AH
12     : D/3 Version : 6.3-2 BASILINE
13     : Compiler   : SIC
14     : Build command : @[320,20]350BATCH
15     :
16     : UNIT = 4, SKDKEY = 17
17     :
18     : Assoc. UPF: [300,1]_SHORT0.____ SHORT0.LST
19     : Assoc. RPF: [300,1]_FARM.____ FARM.LST
20     : SDB: [300,1]PCMOASDB.____ PCMOASDB.CMD
21     :
22     : POINTER SDB VAR. FUNCTION
23     : -----
24     : PSTAT INTn Current EPN alarm state is an integer tied to
25     :           graphic flashing string of value:
26     :           (n=0-100) 0 = previously not alarmed
27     :           1 = previously alarmed, not ack
28     :           2 = previously alarmed, is ack
29     :           3 = alarm is inhibited
30     :           4 = input is off scan
31     :           5 = input is bad
32     :           LEAKn Special INT variable for multiple white alarms on an EPN
33     :           'EPN' ---
34     : PTYPE TYPEn alarm type 0 = none, 1 = white, 2 = yellow,
35     :           3 = yellow and yellow, 4 = red,
36     :           6 = yellow and red
37     : PREV FRVn Reverse acting (not fail safe) flag (1=set)
38     : PSKID SKIDn Used for analog alarms
39     : PPASS FPASn Turn lamp off when alarm is ACKnowledged
40     : PALARM FALMn Analog alarm flag
41     : --- FNING Unused error flag
42     :
43     : MODIFICATION LOG:
44     :
45     : 09-JUN-93 Added comments and baselined for D/3 M. Teats
46     :           version 3.3. CCN# 93JUN01-01 and
47     :           CCN# 93JUN01-02.
48     :
49     : 04-JAN-94 Changed variable FRV2=1 to FRV2=0 per R. Persin
50     :           ECN 603359 and CCN 94JAN04-01
51     :
52     : 15-DEC-94 Revised header to indicate the new VMS J. Berger
53     :           system. Rebase line document. Removed
54     :           obsolete Select() statements.
55     :

```

```

56      ; END MOD LOG
57      ;
58      UNIT      A350
59      ;
60      0000      STAR = 13                ; Start message
61      0001      SKDKEY = 17              ; SKID Key
62      0002      ADI = 0                  ; Alarm destination index
63      0003      FSN0 = 0                 ; No alarm yet
64      0004      TIMEROFF STMRO           ; Alarm timer off
65      0005      TIMERCLR STMRO           ; Alarm timer clear
66      0006      STRNG0 = " "             ; Blank 30 character string
67      0007      ERN0 = 0                 ; Set error number to zero
68      0008      ERRL = 0                 ; Set error line to zero
69      0009      ASSIGN PDEV = '350ALARM' ; Alarm device
70      0010      FTRIP4 = 0               ; FTRIP4 = 1 when audible horn is to be turned on
71      0011      GETUNIT TRIP             ; Which sequence unit is program running in
72      0012      IF TRIP != 4 THEN        ; Abort if running in wrong seq. unit
73          BEGIN
74          FSTART = 1
75          ILOOP = STAR + 1
76          ASSIGN PTYPE = TYPE14
77          PTYPE = 0
78          GOSUB ACKK
79          GOTO TERM
80      0018      END
81      ; IF !SELECT() THEN GOTO AGAIN
82      ;
83      ; BEGIN INITIALIZING
84      ;
85      0019      SSGO = "INIT"
86      0020      INIT: FABORT = 0           ; If FABORT = 1 then abort program
87      0021      FSTART = 0                ; Start flag = 0
88      0022      LAMP ADI,SKDKEY,0         ; Turn off the annunciator key
89      0023      LSG0 = " "               ; Clear the string variables
90      0024      LSG1 = " "
91      0025      LSG2 = " "
92      0026      LSG3 = " "
93      0027      LSG4 = " "
94      0028      PDEV:DEV TMPO = 1         ; Set alarm device to non-alarm state
95      0029      IF AND(PDEV:DEV ALMCT,16) THEN ; If alarm device is inhibited, remove the inhibit
96      0030      PDEV:DEV ALMCT=XOR(PDEV:DEV ALMCT,16)
97      0031      IF AND(PDEV:DEV ALMCT,32) THEN ; If alarm device is ack, clear it
98      0032      PDEV:DEV ALMCT = XOR(PDEV:DEV ALMCT,32)
99      ;
100     ; First, set all flags and integers:
101     ;
102     0033      ASSIGN PDIN = 'LSH-350-2' ; Assign pointer to first DIN
103     0034      ASSIGN PSTAT = INT1
104     0035      ASSIGN PTYPE = TYPE1
105     0036      ASSIGN PREV = FRV1
106     0037      ASSIGN PSKID = SKID1
107     0038      ASSIGN PPASS = FPAS1
108     0039      ASSIGN PALARM = FDIS1
109     ;
110     0040      FOR ILOOP = 1 TO 10

```

```

111 0041      PPASS = 0
112 0042      PREV = 0
113 0043      PTYPE = 0
114 0044      PSTAT = 0
115 0045      PSKID = 0
116 0046      PALARM = 0
117 0047      IF ILOOP < 4 THEN                ; For DINs
118          BEGIN
119 0048      IF (AND(PDIN:DIN_DFLG,3)) THEN    ; Check for alarms (must be red or yellow)
120          BEGIN
121 0049      PTYPE = 2                        ; Default to yellow
122 0051      IF AND(PDIN:DIN_DFLG,1) THEN PREV = 1 ; Reverse acting
123 0053      IF AND(PDIN:DIN_DFLG,4) THEN PTYPE = 4 ; Set to critical alarm
124          END
125 0055      IF ILOOP < 3 THEN STEP PDIN BY 1    ; Point to next DIN
126          END
127 0056  SKP:  STEP PSTAT BY 1
128 0057          STEP PTYPE BY 1
129 0058          STEP PREV BY 1
130 0059          STEP PPASS BY 1
131 0060          STEP PSKID BY 1
132 0061          STEP PALARM BY 1
133 0062          NEXT ILOOP
134      ;
135 0063      FRV1 = 0                ; LSH-350-2
136 0064      FRV2 = 0                ; LDS-350-2
137 0065      FRV3 = 0                ; LDS-PW481
138      ;
139 0066      TYPE7 = 6                ; WFI-350TK
140      ;
141 0067      ANALOG = 0
142      ;
143      ;      END  INITIALIZING
144      ;
145      ;      SEND  PROGRAM  START  MESSAGE
146      ;
147 0068      PTYPE = 0
148 0069      FCLEAR = 0
149 0070      ILOOP = STAR
150 0071      FSTART = 1
151 0072      GOSUB ACKK
152 0073      ON ERROR GOSUB ERRSUB
153      ;
154      ;      BEGIN  PROGRAM  LOOP
155      ;
156 0074  AGAIN: MARK STEP "AGAIN",0,0
157 0075      SSGO = "WAIT"
158      ;      WAITUNTIL SELECT() ! FABORT      ; Wait until micro is selected or program aborted
159      ;      IF (FABORT) THEN GOTO TERM      ; Terminate program if FABORT flag is set
160 0076      WAIT 1
161 0077      IF (FSNO) THEN                    ; If alarm flag set
162 0078      IF STMRO > 60 THEN                ; If alarm flag active for 1 minute
163          BEGIN                            ; Reset timer and turn off flag
164 0079          TIMEROFF STMRO
165 0080          TIMERCLR STMRO

```

```

166 0081          FSN0 = 0
167 0082          ERRN = 0
168 0083          ERRL = 0
169          END
170 ;
171 ; If 350ALARM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
172 ;
173 0084          IF (PDEV:DEV_TMPO = 2 ) & AND(PDEV:DEV_ALMCT,32) THEN
174              BEGIN
175 0085              PDEV:DEV_TMPO = 1          ; Put device in ack state
176 0086              FTRIP4 = 1              ; Set for unit alarm ack, system to graphic
177 0087              ACK = 1                  ; Set for unit alarm ack, graphic to system
178 0088              CRIT
179 0089              LAMP ADI,SKDKEY,1
180              END
181 ;
182 ; BEGIN DINs:
183 ;
184 0090          ASSIGN PDIN = 'LSH-350-2'
185 0091          ASSIGN PTYPE = TYPE1
186 0092          ASSIGN PREV = FRV1
187 0093          ASSIGN PSTAT = INT1
188 0094          ASSIGN PPASS = FPAS1
189 0095          LET LAST = 0
190 0096          LET NXT = 0
191 0097          LMPVAR = 0
192 0098          BITVAL = 0
193 ;
194 0099          LOP:  FOR ILOOP = 1 TO 3
195 0100              IF ( PTYPE > 0 ) THEN          ; If some type of alarm
196                  BEGIN                          ;be1
197 0102                  IF AND(PDIN:DIN_ALMCT,16) THEN PSTAT = 3      ; If EPN inhibited
198 0104                  IF AND(PDIN:DIN_ALMCT,4) THEN PSTAT = 5      ; If EPN is bad input
199 0106                  IF AND(PDIN:DIN_FLG,8) THEN PSTAT = 4        ; If EPN is disabled
200 0107                  IF ( PSTAT < 3 ) THEN          ; If not inhibited
201                      BEGIN                          ;be2
202 0108                      IF PSTAT = 1 THEN
203                          BEGIN                      ;be3
204 0109                          IF PTYPE != 1 THEN
205 0111                          IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
206 0113                          IF (FTRIP4) THEN GOSUB ACK1
207                          END                          ;en3
208 0116                          IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
209 0117                          IF (PSTAT) THEN
210                              BEGIN                  ;be4
211 0119                              IF PDIN:DIN_VAL != PREV THEN GOSUB CLR
212 0120                              NXT = PSTAT
213 0122                              IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
214 0123                              LAST = OR(LAST,NXT)
215                              END                      ;en4
216                              END                      ;en2
217 0125                          ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN PSTAT = 0
218                              END                      ;en1
219 0126                          STEP PDIN BY 1
220 0127                          STEP PTYPE BY 1

```

```

221 0128      STEP PREV BY 1
222 0129      STEP PSTAT BY 1
223 0130      STEP PPASS BY 1
224 0131      NEXT ILOOP
225          ;
226          ;      END DINS:
227          ;
228          ;      BEGIN ANALOG RED AND YELLOWS:
229          ;
230 0132      LET ANALOG = 0
231 0133      ASSIGN PAI = '11-350-1'
232 0134      ASSIGN PSKID = SKID4
233 0135      ASSIGN PTYPE = TYPE4
234 0136      ASSIGN PREV = FRV4
235 0137      FOR JLOOP = 4 TO 10
236 0138      IF (PTYPE) THEN                                ; If some alarm exists
237          BEGIN                                           ;be1
238          BITVAL = PAI:AI_ALMCT
239          IF IAND(BITVAL,76) THEN
240              BEGIN                                           ;be2
241          0141      IF AND(BITVAL,15) THEN                    ; If alarm exists and it is current
242              BEGIN                                           ;be3
243          0143      IF AND(BITVAL,32) THEN NXT = 2            ; If alarm is ack
244              ELSE
245                  BEGIN                                           ;be4
246          0144      IF PSKID I=BITVAL THEN
247                      BEGIN                                           ;be5
248          0146      IF BITVAL < 3 THEN FALMON = 1
249          0147      PSKID = BITVAL
250          0148      PDEV:DEV_TMPO = 2
251          0149      NXT = 1
252          0150      LAMP ADI,SKDKEY,2
253          0151      CRIT
254              END                                           ;en5
255          0152      ELSE NXT = 1
256              END                                           ;en4
257          END                                           ;en3
258          ELSE
259              BEGIN                                           ; Else there is no alarm
260          0153      NXT = 0
261          0154      PSKID = 0
262              END
263          END                                           ;en2
264          ELSE
265              BEGIN                                           ; Else alarm is inhibited
266          0155      NXT = 0
267          0156      PSKID = 0
268              END
269          0157      ANALOG = OR(ANALOG,NXT)
270              END                                           ;en1
271 0158      STEP PSKID BY 1
272 0159      STEP PTYPE BY 1
273 0160      STEP PREV BY 1
274 0161      STEP PAI BY 1
275 0162      NEXT JLOOP

```

```

276      ;
277      ;
278      ;
279      0163  WHT:  WHITE = 0
280      0164      SSWG0 = "RUN"
281      0165      MARK STEP "BELOW",0,0
282      0166      FTRIP4 = 0
283      0168      IF LAST > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
284      0169      CURENT = OR(LAST,WHITE)
285      0170      CURENT = OR(CURENT,ANALOG)
286      0172      IF AND(CURENT,1)THEN LMPVAR = 2
287      0174          ELSE IF AND(CURENT,2)THEN LMPVAR = 1
288      0175          ELSE LMPVAR = 0
289      0176      LAMP ADI,SKDKEY,LMPVAR
290      0177      MARK STEP "BOTTOM",0,0
291      0178      NOCRIT
292      0179      GOTO AGAIN
293      ;
294      ;
295      ;
296      ;
297      ;
298      0180  CLR:  PSTAT = 0                      ; Int set to zero
299      ;
300      ;
301      ;
302      0183      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
303      0184      FCLEAR = 1
304      0185          ; Clear flag
305      0186      GOSUB ACKK
306      ;
307      0187  ALM:  PSTAT = 1                      ; Int set to one
308      ;
309      ;
310      ;
311      0190      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
312      ;
313      ;
314      ;
315      0192      IF AND(PTYPE,3) THEN FALMON = 1
316      0193      PDEV:DEV_TMPO = 2
317      0194      NOCRIT
318      0195      CRIT
319      0196      LAMP ADI,SKDKEY,2                      ; Blink LAMP
320      0197      GOSUB ACKK                      ; Print message
321      0198      RETURN
322      ;
323      0199  ACK1: PSTAT = 2                      ; Ack the alarm
324      0200      FACK = 1                      ; Set ACK flag for C program
325      ;
326      ;
327      ;
328      0202  ACKK: IF ILOOP = 1 THEN STRNG0 = "A-350 TANK LEVEL HIGH"
329      0204      IF ILOOP = 2 THEN STRNG0 = "A-350 LIFT STATION LEAK"
330      0206      IF ILOOP = 3 THEN STRNG0 = "PW-481 WASTE LINE LEAK"

```

```

331 0208 IF LOOP = 13 THEN STRING = "350BATCH PGM STARTING "
332 0210 IF LOOP = 14 THEN STRING = "350BATCH WORKING UNIT "
333 0212 IF LOOP = 15 THEN STRING = "350BATCH TERMINATING "
334
335 ; If acknowledging the alarm, then:
336
337 MSG: IF PTYPE = 1 THEN
338 BEGIN
339 IF (PACK) THEN
340 BEGIN
341 ;
342 ; If white, print it is ack and reset the flag:
343 ;
344 PRINT AD1,"%p ALARM ACKNOWLEDGE",STRING
345 PACK = 0
346 END
347 ELSE PRINT AD1,"WHITE ALARM %p",STRING
348 END
349
350 ; If CLEAR flag is set, print it is clear and reset the flag:
351 ;
352 IF (FCLEAR) THEN
353 BEGIN
354 PRINT AD1,"%p ALARM CLEAR",STRING
355 FCLEAR = 0
356 END
357
358 ; If STARTUP, then print startup message:
359 ;
360 IF (FSTART) THEN
361 BEGIN
362 FSTART = 0
363 PRINT AD1,"%p",STRING
364 END
365 RETURN
366
367 ; If ERROR - print message, start timer, and set flag:
368 ;
369 ERSUB: IF(ERNUM()) = -187) THEN ERSRET
370 ELSE
371 BEGIN
372 IF ERN I= ERNUM() THEN
373 BEGIN
374 TIMERON STMO
375 TIMERCLR STMO
376 FSHO = 1
377 ERN = ERNUM()
378 ERRL = ERRLIN()
379 PRINT AD1,"ERROR # %i AT LINE %i",errm,errl
380 ERSCLR
381 END
382
383 GOTO AGAIN
384
385 ; If FABORT = 1, then abort program:

```



```

386      ;
387      ; TERM: FABORT = 0
388      ; TERM: STRING = "SSOBATCH TERMINATED"
389      ; LOOP = STAR + 2
390      ; START = "FILE"
391      ; START = "TERM"
392      ; COSUB ACK
393      END

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 4416 Bytes, 276 Pages
10 Forward Reference(s)

FEEDBACK

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

06:46:13 16-May-95

Page 1

```

1      ;      F E E D B A T C H . S E Q      VERSION 3
2      ;
3      ;      Function:      Sequence program for Lamp function,
4      ;                      message display on Graphic number 301,
5      ;                      and printer messages.
6      ;
7      ;      Written By      : R. N. Kersey
8      ;      Date           : 05-31-92
9      ;      Facility       : 242-A Evaporator Bldg.
10     ;      Machine        : VAX 4104-A
11     ;      Op. System     : VAX/VMS 5.5-2AH
12     ;      D/3 Version    : 6.3-2 BASELINE
13     ;      Compiler       : SIC
14     ;      Build command  : @([320,20]FEEDBATC
15     ;
16     ;      NOTE: lines commented out with 'OLD' are temporary changes to be
17     ;            reimplemented when operational testing is done. Related
18     ;            changes are in PCBATCH.
19     ;
20     ;      UNIT = 5, SKDKEY = 31
21     ;
22     ;      Assoc. UPF:      [300,1]_PROCESS.____ PROCESS.LST
23     ;      Assoc. RPF:      [300,1]_242A.____ 242A.LST
24     ;      SDB:           [300,1]PCMOA.SDB PCMOASDB.COM
25     ;
26     ;      POINTER      SDB VAR.      FUNCTION
27     ;      -----
28     ;      PSTAT      INTn      Current EPN alarm state is an integer tied to
29     ;                      graphic flashing string of value:
30     ;                      0 = previously not alarmed
31     ;                      1 = previously alarmed, not ack
32     ;                      2 = previously alarmed, is ack
33     ;                      3 = alarm is inhibited
34     ;                      4 = input is off scan
35     ;                      5 = input is bad
36     ;      LEAKn      Special INT variable for multiple white alarms on an EPN
37     ;      'EPN'
38     ;      PDIN      ---
39     ;      PTYP      TYPEn      alarm type 0 = none, 1 = white, 2 = yellow,
40     ;                      3 = yellow and yellow, 4 = red,
41     ;                      6 = yellow and red
42     ;      PREV      FRVn      Reverse acting (not fail safe) flag (1=set)
43     ;      PSKID      SKIDn      Used for analog alarms
44     ;      PPASS      FPASn      Turn lamp off when alarm is ACKnowledged
45     ;      PALARM      FALMn      Analog alarm flag
46     ;      ---      FNING      Unused error flag
47     ;
48     ;      MODIFICATION LOG:
49     ;
50     ;      16-NOV-92      Added O/C display bar feature.      M. Teats
51     ;
52     ;      23-NOV-92      Added 102AW pressure signals, analog red M. Teats
53     ;                      and yellow section.
54     ;
55     ;      30-DEC-92      Fixed white alarm print message.      M. Teats

```

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

06:46:13 16-May-95

Page 2

```

56      ;      13-MAY-93      Removed I1-AW-102 hi white alarm per CCN#      M. Teats
57      ;      93MAY13-01.
58
59      ;--mct1+ 25-MAY-93      In ALM routine, changed FACK=1 to FALMON=1      M. Teats
60      ;      to prevent "alarm ack" message printout
61      ;      when alarm occurs. CCN# 93MAY25-01.
62
63      ;--mct2+ 01-JUN-93      Add code to put FIC-CA1-1 in manual and set      M. Teats
64      ;      its output to zero when HV-CA1-1 closes.
65      ;      CCN# 93JUN01-01.
66
67      ;      07-JUN-93      Added comments. CCN# 93JUN01-01.      M. Teats
68      ;
69      ;      20-JUL-93      Commented and baselined for D/3 version 3.3.      M. Teats
70      ;
71      ;--mct3+ 12-JAN-94      Delete LO white alarm on I1-P-C100 per      M. Teats
72      ;      CCN# 94JAN12-01.
73      ;
74      ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
75      ;      system. Rebase line document. Removed
76      ;      obsolete Select() statements.
77
78      ;      27-APR-95      Revised YS-AW102 to alarm on yellow alarm      J. Berger
79      ;      See WP EE-95-00006 and ECN 622499. (JFB01)
80      ;
81      ;
82      ;      END OF MOD LOG
83      ;
84      UNIT      FEED
85      ;
86      COMMON WHOLE0,WHOLE1,CA1      ; CA1 for synch PID
87      FLAGBITS == P$PARO
88      HV_TMP0 == 'VCA1-INLK':DEV_TMP0
89      HV_TMP1 == 'HV-CA1-1':DEV_TMP1
90      PAW_TMP1 == 'P-AW-102':DEV_TMP1
91      PAW_STAT == 'P-AW-102':DEV_STAT
92      FSN0 == F$PARO
93      FIC1SP == 'FIC-CA1-1':PID_SPVL:3 ; for synch PID
94
95      0000      NOP
96      0001      STAR = 24      ; Start message
97      0002      LET SKDKEY = 31      ; SKID Key
98      0003      LET UNITT = 5      ; SEQ Unit number
99      0004      ASSIGN PDEV = 'PAW102ALM'      ; Alarm device
100     0005      ASSIGN PTYPE = TYPE24
101     0006      LET ADI = 0      ; Alarm destination index
102     0007      LET FSN0 = 0      ; No alarm yet
103     0008      TIMEROFF STIME0      ; Alarm timer off
104     0009      TIMERCLR STIME0      ; Alarm timer clear
105     0010      LET STRNG0 = " "      ; Blank 30 character string
106     0011      LET ERRN = 0      ; Set error number to zero
107     0012      LET ERRL = 0      ; Set error line to zero
108     0013      LET FTRIP4 = 0      ; FTRIP4 = 1 when audible horn is to be turned on
109     0014      GETUNITT TRIP      ; Which sequence unit is program running in
110     0015      IF TRIP != UNITT THEN      ; Abort if running in wrong seq. unit

```

```

111          BEGIN
112 0016      FSTART = 1
113 0017      ILOOP = STAR + 1
114 0018      PTYPE = 0
115 0019      GOSUB ACKK
116 0020      GOTO TERM
117          END
118          ; IF !SELECT() THEN GOTO AGAIN
119          ;
120          ; BEGIN INITIALIZING
121          ;
122          ; ILOOP 0-6 are DINS, 7-16 are AINs
123          ;
124 0021      LET STRNG0 = "INIT"
125 0022      INIT: LET FABORT = 0 ; If FABORT = 1 then abort program
126 0023      LET FSTART = 0 ; Program start flag = 0
127 0024      LAMP AD1,SKDKEY,0 ; Turn off the annunciator key
128 0025      LET LSG0 = " " ; Clear the string variables
129 0026      LET LSG1 = " "
130 0027      LET LSG2 = " "
131 0028      LET LSG3 = " "
132 0029      LET LSG4 = " "
133 0030      PDEV:DEV_TMPO = 1 ; Set alarm device to non-alarm state
134 0031      IF AND(PDEV:DEV_ALMCT,16) THEN ; If alarm device is inhibited, remove the inhibit
135 0032      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
136 0033      IF AND(PDEV:DEV_ALMCT,32) THEN ; If alarm device is ack, clear it
137 0034      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
138          ;
139          ; First, set all flags and integers:
140          ;
141 0035      ASSIGN PDIN = 'YS-AW102' ; Assign pointer to first DIN
142 0036      ASSIGN PPASS = FPAS0
143 0037      ASSIGN PREV = FRV0
144 0038      ASSIGN PSTAT = INTO
145 0039      ASSIGN PTYPE = TYPE0
146 0040      ASSIGN PSKID = SKID0
147          ;
148 0041      FOR ILOOP = 0 TO 16
149 0042      PPASS = 0
150 0043      PREV = 0
151 0045      IF PSTAT > 5 THEN PSTAT = 0 ; Reset current status if undefined
152 0046      PSKID = 0 ; Default previous status to no alarms
153 0047      PTYPE = 2 ; Default to Yellow alarm
154 0048      IF ILOOP <= 6 THEN ; For DINS
155          BEGIN
156 0049      IF AND(PDIN:DIN_DFLG,3) THEN ; Check for alarms if defined (must be red or yellow)
157          BEGIN
158 0050      PTYPE = 2 ; Default to Yellow alarm
159 0052      IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1 ; Reverse acting
160 0054      IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4 ; Set to red critical alarm
161          END
162          END
163 0055      ELSE PTYPE = 1 ; Set to white alarm
164 0057      IF ILOOP < 6 THEN STEP PDIN BY 1 ; Point to next DIN
165 0058      STEP PPASS BY 1

```

```

166 0059      STEP PREV BY 1
167 0060      STEP PSTAT BY 1
168 0061      STEP PSKID BY 1
169 0062      STEP PTYPE BY 1
170 0063      NEXT ILOOP
171
172 0064      ;
173      ;
174 0065      LET FPASO = 1      ; YS-102AW, turn lamp off when alarm is ACK
175 0066      LET TYPE0 = 2      ; YS-AW102 added yellow alam (JFB01)
176 0067      LET TYPE5 = 0      ; ZS-CA1-1C, no alarm
177      ;
178 0068      LET FRV0 = 0      ; YS-AW102 alarm on change from 1 to 0
179 0069      LET FRV1 = 1      ; YS-CA1-1 alarm on change from 0 to 1
180 0070      LET FRV2 = 0      ; LDS-SMPL1 alarm on change from 1 to 0
181 0071      LET FRV7 = 1      ; II-AW-102 is a white alarm
182      ;
183 0072      LET A10 = 12.0      ; II-AW-102 Lo alarm
184      ;
185 0073      FLG13 = 0      ; PID casc (1) flag, for synch FIC-CA1-1 > LIC
186 0074      FLG14 = 0      ; PID auto (1) flag, for synch FIC-CA1-1 > LIC
187 0075      FLG15 = 0      ; PID manual (1) flag, for synch FIC-CA1-1 > LIC
188 0076      A120 = 0.0      ; For PID O/C display bar, FIC-CA1-1 manual (-mct2+)
189      ;
190      ;
191      ;
192      ;
193      ;
194 0077      ANALOG = 0
195 0078      ILOOP = STAR
196 0079      FPASS = 1
197 0080      FSTART = 1
198 0081      GOSUB ACKK
199 0082      FPASS = 0
200 0083      ON ERROR GOSUB ERRSUB
201 0084      ESTPGM "SHUTDOWN"
202      ;
203      ;
204      ;
205 0085  AGAIN:  MARK STEP "WTUNTL",0,0
206      ;
207      ;
208      ;
209      ;
210      ;
211      ;
212      ;
213 0087      IF PWRFAIL() THEN ESTOP
214 0088      MARK STEP "RNNING",0,0
215 0089      WAIT 1
216 0090      IF (FSMO) THEN      ; If alarm flag set
217 0091      IF STIME0 > 60 THEN ; If alarm flag active for 1 minute
218      BEGIN                  ; Reset timer and turn off flag
219 0092      TIMEROFF STIME0
220 0093      TIMERCLR STIME0

```

```

221 0094          LET FSN0 = 0
222 0095          LET ERRN = 0
223 0096          LET ERRL = 0
224          END
225          ;
226          ;      IF PAW10ZALM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
227          ;
228 0097          IF (PDEV:DEV_TMPO = 2) & AND(PDEV:DEV_ALMCT,32) THEN
229              BEGIN
230 0098          LET PDEV:DEV_TMPO = 1 ; Put PAW10ZALM in ack state
231 0099          LET FTRIP4 = 1          ; Set for Unit alarm ack, system to graphic
232 0100          LET ACK = 1            ; Set for Unit alarm ack, graphic to system
233 0101          CRIT
234 0102          LAMP ADI,SKDKEY,1      ; Turn lamp on
235          END
236          ;
237          ;      BEGIN DINs:
238          ;
239 0103          ASSIGN PDIN = 'YS-AW102'
240 0104          ASSIGN PTYPE = TYPE0
241 0105          ASSIGN PREV = FRV0
242 0106          ASSIGN PSTAT = INT0
243 0107          ASSIGN PPASS = FPAS0
244 0108          LET CURENT = 0          ; Used to set LAMP status
245 0109          LET LAST = 0
246 0110          LET ANALOG = 0
247 0111          LET NXT = 0
248 0112          LMPVAR = 0
249 0113          BITVAL = 0
250          ;
251 0114          LOP:  FOR ILOOP = 0 TO 6
252 0115          IF PTYPE THEN          ; If some type of alarm
253              BEGIN
254 0117          IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3      ; If EPN inhibited
255 0119          IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5      ; If EPN is bad input
256 0121          IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4      ; If EPN is disabled
257 0122          IF PSTAT < 3 THEN                                  ; If not inhibited
258              BEGIN
259 0123          IF PSTAT = 1 THEN
260                  BEGIN
261 0124          IF PTYPE != 1 THEN
262 0126          IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
263 0128          IF (FTRIP4) THEN GOSUB ACK1
264                  END
265 0131          IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
266 0132          IF PSTAT THEN
267                  BEGIN
268 0134          IF PDIN:DIN_VAL != PREV THEN GOSUB CLR
269 0135          NXT = PSTAT
270 0137          IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
271 0138          LAST = OR(LAST,NXT)
272                  END
273          END
274 0140          ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8)THEN LET PSTAT = 0
275          END

```

```

276 0141      STEP PDIN BY 1
277 0142      STEP PTYPE BY 1
278 0143      STEP PREV BY 1
279 0144      STEP PSTAT BY 1
280 0145      STEP PPASS BY 1
281          ; IF ISELECT() THEN GOTO AGAIN
282 0146      REMAIN = LAST
283 0147      NEXT ILOOP
284          ;
285          ; END DINS
286          ;
287          ; BEGIN ANALOG RED AND YELLOWS:
288          ;
289 0148      ASSIGN PAI = 'II-AW-102'
290 0149      ASSIGN PSKID = SKID7
291 0150      ASSIGN PTYPE = TYPE7
292 0151      FOR JLOOP = 7 TO 16
293 0152      IF (PTYPE) THEN                                ; If some alarm exists
294          BEGIN                                           ;be1
295 0153          BITVAL = PAI:AI_ALMCT
296 0154          IF IAND(BITVAL,16) THEN                    ; If not inhibited
297              BEGIN                                       ;be2
298 0155          IF AND(BITVAL,15) THEN                      ; If alarm exists and is current
299              BEGIN                                       ;be3
300 0157          IF AND(BITVAL,32) THEN NXT = 2              ; If alarm is ack
301              ELSE
302              BEGIN                                       ;be4
303 0158          IF PSKID I=BITVAL THEN
304              BEGIN                                       ;be5
305 0159          LET PSKID = BITVAL
306 0161          IF BITVAL < 3 THEN FALMON = 1
307 0162          PDEV:DEV_TMPO = 2
308 0163          NXT = 1 -
309              END                                         ;en5
310 0164          ELSE NXT = 1
311              END                                         ;en4
312          END                                             ;en3
313          ELSE
314              BEGIN
315 0165          NXT = 0
316 0166          LET PSKID = 0
317              END
318          END                                             ;en2
319          ELSE
320              BEGIN
321 0167          NXT = 0
322 0168          LET PSKID = 0
323              END
324 0169          ANALOG = OR(ANALOG,NXT)
325              END                                         ;en1
326 0170      STEP PSKID BY 1
327 0171      STEP PTYPE BY 1
328 0173      IF JLOOP < 16 THEN STEP PAI BY 1
329 0174      NEXT JLOOP
330          ;

```



```

331      ;      END ANALOG RED AND YELLOWS
332      ;
333      ;      BEGIN ANALOG WHITE ALARMS:
334      ;
335      0175      WHITE = 0
336      0176      ASSIGN PA1 = '11-AW-102'
337      0177      ASSIGN PALARM = A10
338      0178      ASSIGN PSTAT = INT7
339      0179      ASSIGN PTYPE = TYPE7
340      0180      ILOOP = 7
341      0181      GOSUB ANLG
342      ;
343      ;      11-AW-102 Lo, use with String packet 78:
344      ;
345      0183      IF (INT7 & INT7 < 3) THEN LEAK0 = INT7 + 2
346      0184      ELSE LEAK0 = 0
347      ;
348      ;      END ANALOG WHITE ALARMS:
349      ;
350      0185      MARK STEP "BELOW",0,0
351      0186      FTRIP4 = 0
352      0188      IF REMAIN > 3 THEN PRINT AD1," LAMP VARIABLE IS GREATER THAN 3!"
353      0189      CURENT = OR(REMAIN,WHITE)
354      0190      CURENT = OR(CURENT,ANALOG)
355      0192      IF AND(CURENT,1) THEN LMPVAR = 2
356      0194      ELSE IF AND(CURENT,2) THEN LMPVAR = 1
357      0195      ELSE LMPVAR = 0
358      0196      LAMP AD1,SKDKEY,LMPVAR
359      0197      MARK STEP "BOTTOM",0,0
360      0198      NOCRIT
361      ;
362      ;      BEGIN HV-CA1-1 LOGIC (for graphic only):
363      ;
364      0199      ASSIGN FLAGBITS = ATO
365      0200      BVAL = 1
366      0201      FOR IN1 = 0 TO 12
367      0202      FLAGBITS = AND(BVAL,HV_TMP0)
368      0203      BVAL=BVAL/1
369      0204      STEP FLAGBITS BY 1
370      0205      NEXT IN1
371      0206      SA1 = 'YS-CA1-1':DIN_VAL
372      0207      AT25 = AND(HV_TMP1,512)
373      0208      AT26 = AND(HV_TMP1,1024)
374      0209      AT27 = AND(HV_TMP1,2048)
375      0210      AT30 = AND(HV_TMP1,16384)
376      0211      AT31 = AND(HV_TMP1,32768)
377      0212      SA2 = AND('RSH-VVB/G':DIN_VAL,'PDSH-FC51':DIN_VAL)
378      0213      AT14 = AND('WFSH-CA11':DIN_VAL,'WFSH-CA12':DIN_VAL)
379      0214      AT15 = AND('RSH-CA1-1':DIN_VAL,'PDSH-CA1':DIN_VAL)
380      0215      AT14 = AND(AT14,SA2)
381      0216      SA3 = AND(AT14,AT15)
382      0217      SA4 = OR('YS-CA1-1':DIN_VAL,SA3)
383      0218      AT14 = AND(HV_TMP1,32)
384      0219      AT15 = AND('HV-CA1-1':DEV_STAT,2)
385      0220      SA5 = AND('HV-CA1-1':DEV_CMD,2)

```

```

386 0221      SA6 = AND(SA5,SA4)
387 0222      SB0 = 'ZS-CA1-1C':DIN_VAL
388 0223      SB1 = 'ZS-CA1-1N':DIN_VAL
389 0224      AT20 = AT12
390 0225      AT21 = AND('VCA1-INLK':DEV_CMD,2)
391 0226      AT23 = AND('VCA1-INLK':DEV_TMP1,1)
392 0227      AT22 = XOR(AT21,AT23)
393 0228      AT28 = AT31
394
395      ;
396      ;
397      ;
398      ;
399 0229      END HV-CA1-1 LOGIC
400
401 0230      BEGIN P-AW-102 LOGIC (for graphic only):
402
403 0231      ASSIGN FLAGBITS = BT23
404 0232      BVAL = 128
405 0233      FOR IN1 = 23 TO 29
406 0234          FLAGBITS = AND(BVAL,PAW_TMP1)
407 0235          BVAL=BVAL/1
408 0236          STEP FLAGBITS BY 1
409 0237      NEXT IN1
410 0238      BT31 = AND(PAW_TMP1,32768)
411 0239      BT0 = AND('HS-ARM-AW':DEV_TMP0,128)
412 0240      BT1 = OR('RSH-AW-12':DIN_VAL,BT0)
413 0241      AT15 = AND('LDS-AW-PP':DIN_VAL,! 'YS-AW-MSD':DIN_VAL)
414 0242      BT2 = AND(AT15,BT1)
415 0243      AT14 = AND('RSH-SP-AW':DIN_VAL,'LDS-SN269':DIN_VAL)
416 0244      AT15 = AND('LDS-SN270':DIN_VAL,'PSH-AWSN2':DIN_VAL)
417 0245      AT14 = AND(AT14,AT15)
418 0246      BT3 = AND(BT2,AT14)
419 0247      AT14 = AND('WFSH-CA11':DIN_VAL,'WFSH-CA12':DIN_VAL)
420 0248      AT15 = AND('RSH-VVB/G':DIN_VAL,'RSH-CA1-1':DIN_VAL)
421 0249      AT16 = AND('PDSH-CA11':DIN_VAL,'RSH-CA1-1':DIN_VAL)
422 0250      AT17 = AND(BT3,'LSH-SC-1':DIN_VAL)
423 0251      AT18 = AND(AT14,AT15)
424 0252      AT19 = AND(AT16,AT17)
425 0253      BT4 = AND(AT18,AT19)
426 0254      BT5 = AND('P-AW-102':DEV_TMP0,2)
427 0255      BT6 = 'YS-AW102':DIN_VAL
428 0256      BT14 = AND(BT31,PAW_STAT)
429 0257      SB3 = AND(PAW_STAT,4)
430 0258      SB4 = AND(PAW_STAT,16)
431 0259      SB5 = AND('HV-CA1-1':DEV_STAT,4) ; RESET
432 0260      SB6 = AND('HV-CA1-1':DEV_STAT,16) ; CHGSTATE
433 0261      SB7 = AND('HV-CA1-1':DEV_STAT,32) ; INTERLOCK
434 0262      SC0 = AND(PAW_STAT,32)
435 0263      BT20 = AND(HV_TMP1,16)
436 0264      BT7 = AND(BT20,BT14)
437 0265      BT8 = AND(BT7,BT4)
438 0266      IF AND(PAW102ALM:DEV_FLG,64)THEN PUTMANL PAW102ALM
439 0267      IF AND('P-AW-102':DEV_FLG,64)THEN PUTMANL 'P-AW-102'
440 0268      IF AND('HV-CA1-1':DEV_FLG,64)THEN PUTMANL 'HV-CA1-1'
441 0269      IF AND('VCA1-INLK':DEV_FLG,64)THEN PUTMANL 'VCA1-INLK'
442
443      ;
444      ;
445      ;
446      ;
447      ;
448      ;
449      ;
450      ;
451      ;
452      ;
453      ;
454      ;
455      ;
456      ;
457      ;
458      ;
459      ;
460      ;
461      ;
462      ;
463      ;
464      ;
465      ;
466      ;
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473      ;
474      ;
475      ;
476      ;
477      ;
478      ;
479      ;
480      ;
481      ;
482      ;
483      ;
484      ;
485      ;
486      ;
487      ;
488      ;
489      ;
490      ;
491      ;
492      ;
493      ;
494      ;
495      ;
496      ;
497      ;
498      ;
499      ;
500      ;
501      ;
502      ;
503      ;
504      ;
505      ;
506      ;
507      ;
508      ;
509      ;
510      ;
511      ;
512      ;
513      ;
514      ;
515      ;
516      ;
517      ;
518      ;
519      ;
520      ;
521      ;
522      ;
523      ;
524      ;
525      ;
526      ;
527      ;
528      ;
529      ;
530      ;
531      ;
532      ;
533      ;
534      ;
535      ;
536      ;
537      ;
538      ;
539      ;
540      ;
541      ;
542      ;
543      ;
544      ;
545      ;
546      ;
547      ;
548      ;
549      ;
550      ;
551      ;
552      ;
553      ;
554      ;
555      ;
556      ;
557      ;
558      ;
559      ;
560      ;
561      ;
562      ;
563      ;
564      ;
565      ;
566      ;
567      ;
568      ;
569      ;
570      ;
571      ;
572      ;
573      ;
574      ;
575      ;
576      ;
577      ;
578      ;
579      ;
580      ;
581      ;
582      ;
583      ;
584      ;
585      ;
586      ;
587      ;
588      ;
589      ;
590      ;
591      ;
592      ;
593      ;
594      ;
595      ;
596      ;
597      ;
598      ;
599      ;
600      ;
601      ;
602      ;
603      ;
604      ;
605      ;
606      ;
607      ;
608      ;
609      ;
610      ;
611      ;
612      ;
613      ;
614      ;
615      ;
616      ;
617      ;
618      ;
619      ;
620      ;
621      ;
622      ;
623      ;
624      ;
625      ;
626      ;
627      ;
628      ;
629      ;
630      ;
631      ;
632      ;
633      ;
634      ;
635      ;
636      ;
637      ;
638      ;
639      ;
640      ;
641      ;
642      ;
643      ;
644      ;
645      ;
646      ;
647      ;
648      ;
649      ;
650      ;
651      ;
652      ;
653      ;
654      ;
655      ;
656      ;
657      ;
658      ;
659      ;
660      ;
661      ;
662      ;
663      ;
664      ;
665      ;
666      ;
667      ;
668      ;
669      ;
670      ;
671      ;
672      ;
673      ;
674      ;
675      ;
676      ;
677      ;
678      ;
679      ;
680      ;
681      ;
682      ;
683      ;
684      ;
685      ;
686      ;
687      ;
688      ;
689      ;
690      ;
691      ;
692      ;
693      ;
694      ;
695      ;
696      ;
697      ;
698      ;
699      ;
700      ;
701      ;
702      ;
703      ;
704      ;
705      ;
706      ;
707      ;
708      ;
709      ;
710      ;
711      ;
712      ;
713      ;
714      ;
715      ;
716      ;
717      ;
718      ;
719      ;
720      ;
721      ;
722      ;
723      ;
724      ;
725      ;
726      ;
727      ;
728      ;
729      ;
730      ;
731      ;
732      ;
733      ;
734      ;
735      ;
736      ;
737      ;
738      ;
739      ;
740      ;
741      ;
742      ;
743      ;
744      ;
745      ;
746      ;
747      ;
748      ;
749      ;
750      ;
751      ;
752      ;
753      ;
754      ;
755      ;
756      ;
757      ;
758      ;
759      ;
760      ;
761      ;
762      ;
763      ;
764      ;
765      ;
766      ;
767      ;
768      ;
769      ;
770      ;
771      ;
772      ;
773      ;
774      ;
775      ;
776      ;
777      ;
778      ;
779      ;
780      ;
781      ;
782      ;
783      ;
784      ;
785      ;
786      ;
787      ;
788      ;
789      ;
790      ;
791      ;
792      ;
793      ;
794      ;
795      ;
796      ;
797      ;
798      ;
799      ;
800      ;
801      ;
802      ;
803      ;
804      ;
805      ;
806      ;
807      ;
808      ;
809      ;
810      ;
811      ;
812      ;
813      ;
814      ;
815      ;
816      ;
817      ;
818      ;
819      ;
820      ;
821      ;
822      ;
823      ;
824      ;
825      ;
826      ;
827      ;
828      ;
829      ;
830      ;
831      ;
832      ;
833      ;
834      ;
835      ;
836      ;
837      ;
838      ;
839      ;
840      ;
841      ;
842      ;
843      ;
844      ;
845      ;
846      ;
847      ;
848      ;
849      ;
850      ;
851      ;
852      ;
853      ;
854      ;
855      ;
856      ;
857      ;
858      ;
859      ;
860      ;
861      ;
862      ;
863      ;
864      ;
865      ;
866      ;
867      ;
868      ;
869      ;
870      ;
871      ;
872      ;
873      ;
874      ;
875      ;
876      ;
877      ;
878      ;
879      ;
880      ;
881      ;
882      ;
883      ;
884      ;
885      ;
886      ;
887      ;
888      ;
889      ;
890      ;
891      ;
892      ;
893      ;
894      ;
895      ;
896      ;
897      ;
898      ;
899      ;
900      ;
901      ;
902      ;
903      ;
904      ;
905      ;
906      ;
907      ;
908      ;
909      ;
910      ;
911      ;
912      ;
913      ;
914      ;
915      ;
916      ;
917      ;
918      ;
919      ;
920      ;
921      ;
922      ;
923      ;
924      ;
925      ;
926      ;
927      ;
928      ;
929      ;
930      ;
931      ;
932      ;
933      ;
934      ;
935      ;
936      ;
937      ;
938      ;
939      ;
940      ;
941      ;
942      ;
943      ;
944      ;
945      ;
946      ;
947      ;
948      ;
949      ;
950      ;
951      ;
952      ;
953      ;
954      ;
955      ;
956      ;
957      ;
958      ;
959      ;
960      ;
961      ;
962      ;
963      ;
964      ;
965      ;
966      ;
967      ;
968      ;
969      ;
970      ;
971      ;
972      ;
973      ;
974      ;
975      ;
976      ;
977      ;
978      ;
979      ;
980      ;
981      ;
982      ;
983      ;
984      ;
985      ;
986      ;
987      ;
988      ;
989      ;
990      ;
991      ;
992      ;
993      ;
994      ;
995      ;
996      ;
997      ;
998      ;
999      ;
1000      ;

```

```

441      ; BEGIN SYNCH PID CASCADED SETPT
442      ;
443      ;
444      ; Fuction: Closes slave controller's PID (SPID) cascade,
445      ; sets cascade controller's PID (CPID) to auto,
446      ; sets the CPID output equal to the SPID setpoint
447      ; only on a manual to auto transition in the SPID.
448      ; Puts LIC's in manual if FIC was previously auto
449      ; and cascade open.
450 0272 IF AND('FIC-CA1-1':CB_FLG:3,64) THEN ; If in auto
451      BEGIN ;be1
452 0273 IF (FLG14=0) & (FLG15=0) THEN ; If initializing
453      BEGIN ;be2
454 0275 IF AND('FIC-CA1-1':PID_SPVS:3,64) THEN LET FLG13 = 1 ; If casc closed
455 0276 LET FLG14 = 1
456 0277 LET FLG15 = 0
457      END ;en2
458 0278 IF AND('FIC-CA1-1':PID_SPVS:3,64) & (FLG13=0) THEN ; If clos casc + prev auto-open
459      BEGIN ;be3
460 0279 IF IAND('LIC-CA1-1':CB_FLG:1,64) THEN ; If LIC-CA1-1 in manual
461      BEGIN ;be3.1
462 0280 PUTOUT 'LIC-CA1-1',FIC1SP
463 0281 PUTAUTO 'LIC-CA1-1'
464      END ;en3.1
465 0282 IF IAND('LIC-CA1-2':CB_FLG:1,64) THEN ; If LIC-CA1-2 in manual
466      BEGIN ;be3.2
467 0283 PUTOUT 'LIC-CA1-2',FIC1SP
468 0284 PUTAUTO 'LIC-CA1-2'
469      END ;en3.2
470 0285 LET FLG13 = 1
471      END ;en3
472 0286 IF IAND('FIC-CA1-1':PID_SPVS:3,64) & (FLG13=1) THEN ; If open casc + prev auto-clos
473      BEGIN ;be4
474 0288 IF AND('LIC-CA1-1':CB_FLG:1,64) THEN PUTMANL 'LIC-CA1-1'
475 0290 IF AND('LIC-CA1-2':CB_FLG:1,64) THEN PUTMANL 'LIC-CA1-2'
476 0291 LET FLG13 = 0
477      END ;en4
478 0292 IF (FLG14=0) & (FLG15=1) THEN ; If FIC previously manual
479      BEGIN ;be5
480 0293 CRIT
481 0294 PUTOUT 'LIC-CA1-1',FIC1SP
482 0295 PUTOUT 'LIC-CA1-2',FIC1SP
483 0296 CLOSCASC 'FIC-CA1-1'
484 0298 IF IAND('LIC-CA1-1':CB_FLG:1,64) THEN PUTAUTO 'LIC-CA1-1'
485 0300 IF IAND('LIC-CA1-2':CB_FLG:1,64) THEN PUTAUTO 'LIC-CA1-2'
486 0301 LET FLG13 = 1
487 0302 LET FLG14 = 1
488 0303 LET FLG15 = 0
489 0304 NOCRIT
490      END ;en5
491      END ;en1
492 ELSE ; If FIC in manual
493      BEGIN ;be7
494 0305 IF (FLG14=1) & (FLG15=0) THEN ; If FIC previously auto
495      BEGIN ;be8

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

06:46:13 16-May-95

Page 10

```

496 0307      IF AND('LIC-CA1-1':CB_FLG:1,64) THEN PUTMANL 'LIC-CA1-1'
497 0309      IF AND('LIC-CA1-2':CB_FLG:1,64) THEN PUTMANL 'LIC-CA1-2'
498 0310      LET FLG13 = 0
499          END
500 0311      IF ((FLG14=1)&(FLG15=0)) ; ((FLG14=0)&(FLG15=0)) THEN ; If FIC prev auto or init
501          BEGIN ;en8
502 0312      LET FLG14 = 0 ;be9
503 0313      LET FLG15 = 1
504          END
505          ;en9
506          ;en7
507      ;
508      ;+mct2-
509      ;
510      ;
511      ;
512      ;
513      ;
514 0315      IF AND('HV-CA1-1':DEV_STAT,2) THEN DLEAK2 = 1 ; If Valve is open
515 0316      IF IAND('HV-CA1-1':DEV_STAT,2) & AND(DLEAK2,1) THEN ; If not open and 1st pass
516          BEGIN
517 0317      PUTMANL 'FIC-CA1-1'
518 0318      PRINT ADI," HV-CA1-1 IS NOT OPEN >>> FIC-CA1-1 PUT IN MANUAL"
519 0319      PUTOUT 'FIC-CA1-1',A120
520 0320      PRINT ADI," HV-CA1-1 IS NOT OPEN >>> FIC-CA1-1 OUTPUT SET TO ZERO"
521 0321      DLEAK2 = 0
522          END
523      ;
524      ;
525      ;+mct2+
526      ;
527      ;
528      ;
529      ;
530      ;
531 0323      IF AND('PB1-BYPAS':DEV_TMP0,256) THEN DLEAK3 = 1 ; Permissive $TB must be Hi (1)
532 0324      IF IAND('PB1-BYPAS':DEV_TMP0,256) & AND(DLEAK3,1) THEN ; Permissive lost and 1st pass
533          BEGIN
534 0325      PUTMANL 'FIC-CA1-1'
535 0326      PRINT ADI," PB1-BYPAS PERMISSIVE LOST >>> FIC-CA1-1 PUT IN MANUAL"
536 0327      PUTOUT 'FIC-CA1-1',A120
537 0328      PRINT ADI," PB1-BYPAS PERMISSIVE LOST >>> FIC-CA1-1 OUTPUT SET TO ZERO"
538 0329      DLEAK3 = 0
539          END
540      ;
541      ;
542      ;
543 0330      GOTO AGAIN
544      ;
545      ;
546      ;
547      ;
548      ;
549      ;
550 0331      ANLG: IF ISELECT() THEN GOTO AGAIN
551      ANLG: IF (PTYPE = 1) THEN ; If some alarm

```

```

551      BEGIN                                     ;be1
552      0332 BITVAL = PA1:A1_ALMCT
553      0333 IF I(AND(BITVAL,16)) THEN           ; If not inhibited
554      BEGIN                                     ;be2
555      0335 IF PSTAT = 3 THEN LET PSTAT = 0
556      0336 IF (PREV) THEN                       ; If rev acting, LOL0, or LO
557      BEGIN                                     ;be3
558      0337 IF (PSTAT = 0) THEN                   ; If not previously alarmed
559      BEGIN                                     ;be4
560      0339 IF (PA1:A1_MEAS <= PALARM) THEN GOSUB ALM
561      END                                       ;en4
562      ELSE
563      BEGIN                                     ;be5
564      0340 IF (PSTAT = 1) THEN                   ; If previously alarmed
565      BEGIN                                     ;be6
566      0342 IF (FTRIP4) THEN GOSUB ACK1           ; If ACK flag set, then ack it
567      0345 IF PTYPE != 1 THEN IF AND(PA1:A1_ALMCT,32) THEN GOSUB ACK1
568      END                                       ;en6
569      0347 IF (PA1:A1_MEAS >= PALARM + PA1:A1_DEAD) THEN GOSUB CLR
570      END                                       ;en5
571      ELSE                                     ;en3
572      BEGIN                                     ;be10
573      0348 IF (PSTAT = 0) THEN
574      BEGIN                                     ;be11
575      0350 IF (PA1:A1_MEAS >= PALARM) THEN GOSUB ALM
576      END                                       ;en11
577      ELSE
578      BEGIN                                     ;be12
579      0351 IF (PSTAT = 1) THEN
580      BEGIN                                     ;be13
581      0353 IF (FTRIP4) THEN GOSUB ACK1
582      0356 IF PTYPE != 1 THEN IF AND(PA1:A1_ALMCT,32) THEN GOSUB ACK1
583      END                                       ;en13
584      0358 IF (PA1:A1_MEAS <= PALARM - PA1:A1_DEAD) THEN GOSUB CLR
585      END                                       ;en12
586      END                                       ;en10
587      0359 NXT = PSTAT
588      0361 IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
589      END                                       ;en2
590      ELSE
591      BEGIN
592      0362 LET PSTAT = 3
593      0363 NXT = 0
594      END
595      0364 WHITE = OR(WHITE,NXT)
596      END                                       ;en1
597      0365 ELSE PSTAT = 0
598      0366 STEP PALARM BY 1
599      0367 STEP PSTAT BY 1
600      0368 STEP PTYPE BY 1
601      0369 STEP PREV BY 1
602      0370 STEP PPASS BY 1
603      0371 RETURN
604
605      ;

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

06:46:13 16-May-95

Page 12

```

606      ;      END ANALOG WHITE SUBROUTINE
607      ;
608      ;      BEGIN CLEAR, ALARM, ACKNOWLEDGE, MESSAGE, ERROR, AND TERMINATE SUBROUTINES:
609      ;
610 0372 CLR:  LET PSTAT = 0                      ; Int set to zero
611      ;
612      ;      If not white then unack:
613      ;
614 0375      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
615 0376      LET FCLEAR = 1                      ; Clear flag
616 0377      GOSUB ACKK                          ; Print message
617 0378      RETURN
618      ;
619 0379 ALM:  LET PSTAT = 1                      ; Int set to one
620      ;
621      ;      If not white then unack:
622      ;
623 0382      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
624      ;
625      ;      Turn audible alarm on if white or yellow:
626      ;
627 0384      IF AND(PTYPE,3) THEN FALMOW = 1      ; -mct1+
628 0385      CRIT
629 0386      LET PDEV:DEV_TMPO = 2
630 0387      GOSUB ACKK                          ; Print message
631 0388      RETURN
632      ;
633 0389 ACK1: LET PSTAT = 2                      ; Ack the alarm
634 0390      LET FACK = 1                      ; Set ACK flag for C program
635      ;
636      ;      ALARM TYPER STRINGS:
637      ;
638 0392 ACKK: IF ILOOP = 0 THEN STRNGO = "FEED PUMP YS-AW102 SHUTDOWN "
639 0394      IF ILOOP = 1 THEN STRNGO = "FEED VALVE INTERLOCK BYPASS ON"
640 0396      IF ILOOP = 2 THEN STRNGO = "FEED SAMPLER ENCLOSURE LEAK "
641 0398      IF ILOOP = 3 THEN STRNGO = "TK-102-AW PRESSURE HI "
642 0400      IF ILOOP = 4 THEN STRNGO = "FEED PUMP INTERLOCK BYPASS ON "
643 0402      IF ILOOP = 7 THEN STRNGO = "FEED PUMP II-AW-102 AMPS LO "
644 0404      IF ILOOP = 24 THEN STRNGO = "START FEEDBATCH PGM "
645 0406      IF ILOOP = 25 THEN STRNGO = "FEEDBATCH WRONG UNIT"
646 0408      IF ILOOP = 26 THEN STRNGO = "TERMINATE FEEDBATCH "
647      ;
648      ;      If acknowledging alarm, then:
649      ;
650 0409 AMSG: IF PTYPE = 1 THEN
651      BEGIN
652      ;
653      ;      If white alarm, print:
654      ;
655 0411      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT AD1,"WHITE ALARM %p",STRNGO
656 0412      IF (FACK) THEN
657      BEGIN
658      ;
659      ;      If white, print it is ack and reset the flag:
660      ;

```

```

661 0413      PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
662 0414      LET FACK = 0
663          END
664          END
665          ;
666          ; If CLEAR flag is set, print it is clear and reset the flag:
667          ;
668 0415      IF (FCLEAR) THEN
669          BEGIN
670 0416      PRINT ADI,"%p ALARM CLEAR",STRNGO
671 0417      LET FCLEAR = 0
672          END
673          ;
674          ; If STARTUP, then print startup message:
675          ;
676 0418      IF (FSTART) THEN
677          BEGIN
678 0419      LET FSTART = 0
679 0420      PRINT ADI,"%p",STRNGO
680          END
681 0421      RETURN
682          ;
683          ; If ERROR - print message, start timer, and set flag:
684          ;
685 0422      ERRSUB: IF ERRN != ERRNUM() THEN
686          BEGIN
687 0423      TIMEROW STIMEO
688 0424      TIMERCLR STIMEO
689 0425      LET FSN0 = 1
690 0426      LET ERRN = ERRNUM()
691 0427      LET ERRL = ERRLIN()
692 0428      PRINT ADI,"ERROR # %i AT LINE %i",errn,errl
693          END
694 0429      ERSCLR
695 0430      GOTO AGAIN
696          ;
697          ; If FABORT = 1, then abort program:
698          ;
699          ; TERM: LET FABORT = 0
700 0431      TERM: LET STRNGO = "FEEDBATCH TERMINATED"
701 0432      LET ILOOP = STAR + 2
702 0433      LET FSTART = 1
703 0434      LET STRNGO = "TERM"
704 0435      GOSUB ACKK
705          END

```

Total Errors Detected 0   Total Warnings 0
 Sequence Program Size = 8736 Bytes, 546 Pages
 19 Forward Reference(s)

CA1BATCH

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:34:11 24-Feb-95

Page 1

```

1      ;      C A 1 B A T C H . S E Q      VERSION 2
2      ;
3      ;      Function:      Sequence program for Lamp function,
4      ;                      message display on Graphic number 010,
5      ;                      and printer messages.
6      ;
7      ;      Written By      : R. M. Kersey
8      ;      Date           : 05-31-92
9      ;      Facility       : 242-A Evaporator Bldg.
10     ;      Machine        : VAX 4104-A
11     ;      Op. System     : VAX/VMS 5.5-2AH
12     ;      D/3 Version    : 6.3-2 BASILINE
13     ;      Compiler       : SIC
14     ;      Build command  : @[320,20]CA1BATCH
15     ;
16     ;      UNIT = 6, SKDKEY = 34
17     ;
18     ;      Assoc. UPF:      [300,1]_PROCESS.____ PROCESS.LST
19     ;      Assoc. RPF:      [300,1]_242A.____ 242A.LST
20     ;      SDB:            [300,1]PCMOA.SDB PCMOASDB.CMD
21     ;
22     ;      POINTER      SDB VAR.      FUNCTION
23     ;      -----
24     ;      PSTAT      INTn      Current EPN alarm state is an integer tied to
25     ;                      graphic flashing string of value:
26     ;                      (n=0-100)
27     ;                      0 = previously not alarmed
28     ;                      1 = previously alarmed, not ack
29     ;                      2 = previously alarmed, is ack
30     ;                      3 = alarm is inhibited
31     ;                      4 = input is off scan
32     ;                      5 = input is bad
33     ;      LEAKn      Special INT variable for multiple white alarms on an EPN
34     ;      PDIN      'EPN'
35     ;      PTYPE      TYPEn      alarm type 0 = none, 1 = white, 2 = yellow,
36     ;                      3 = yellow and yellow, 4 = red,
37     ;                      6 = yellow and red
38     ;      PREV      FRVn      Reverse acting (not fail safe) flag (1=set)
39     ;      PSKID      SKIDn      Used for analog alarms
40     ;      PPASS      FPASn      Turn lamp off when alarm is Acknowledged
41     ;      PALARM      FALMn      Analog alarm flag
42     ;      ---      FWING      Unused error flag
43     ;
44     ;      MODIFICATION LOG:
45     ;
46     ;      16-SEP-92      Delete white alarms on DDI-CA1-1      M. Teats
47     ;
48     ;      30-DEC-92      Fixed white alarm print message.      M. Teats
49     ;
50     ;      07-MAR-93      Added comments and baselined for      M. Teats
51     ;                      D/3 version 3.3. CCN# 93JUN01-01.
52     ;
53     ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
54     ;                      system. Rebase line document. Removed
55     ;                      obsolete Select() statements.

```

```

56 ; END OF MOD LOG
57 ;
58 UNIT CA1
59 ;
60 COMMON PB2
61 ;
62 0000 NOP
63 0001 STAR = 17 ; Start message
64 0002 LET SKDKEY = 34 ; SKID Key
65 0003 LET UNIT = 6 ; SEQ Unit number
66 0004 ASSIGN PDEV = 'CA1-ALARM' ; Alarm device
67 0005 ASSIGN PTYPE = TYPE17
68 0006 LET ADI = 0 ; Alarm Destination Index
69 0007 LET FSN0 = 0 ; No alarm yet
70 0008 TIMEROFF STIME0 ; Alarm timer off
71 0009 TIMERCLR STIME0 ; Alarm timer clear
72 0010 STRNG0 = " " ; Blank 30 character string
73 0011 LET ERRN = 0 ; Set error number to zero
74 0012 LET FERRL = 0 ; Set error line to zero
75 0013 LET FTRIP4 = 0 ; FTRIP4 = 1 when audible horn is to be turned on
76 0014 GETUNIT TRIP ; Which sequence unit is program running in
77 0015 IF TRIP != UNIT THEN ; Abort if running in wrong seq. unit
78 BEGIN
79 0016 FSTART = 1
80 0017 ILOOP = STAR + 1
81 0018 PTYPE = 0
82 0019 GOSUB ACKK
83 0020 GOTO TERM
84 END
85 ; IF ISELECT() THEN GOTO AGAIN
86 ;
87 ; BEGIN INITIALIZING
88 ;
89 0021 STRNG0 = "INIT"
90 0022 INIT: LET FABORT = 0 ; If FABORT = 1 then abort program
91 0023 LET FSTART = 0 ; Program start flag = 0
92 0024 LAMP ADI,SKDKEY,0 ; Turn off annunciator key
93 0025 LET LSG0 = " " ; Clear the string variables
94 0026 LET LSG1 = " "
95 0027 LET LSG2 = " "
96 0028 LET LSG3 = " "
97 0029 LET LSG4 = " "
98 0030 LET PDEV:DEV_TMPO = 1 ; Set alarm device to non-alarm state
99 0031 IF AND(PDEV:DEV_ALMCT,16) THEN ; If alarm device is inhibited, remove the inhibit
100 0032 PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
101 0033 IF AND(PDEV:DEV_ALMCT,32) THEN ; If alarm device is ack, clear it
102 0034 PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
103 ;
104 ; First, set all flags and integers:
105 ;
106 0035 ASSIGN PDIN = 'PSH-CA111' ; Assign pointer to first DIN
107 0036 ASSIGN PSTAT = INTO
108 0037 ASSIGN PTYPE = TYPE0
109 0038 ASSIGN PREV = FRVO
110 0039 ASSIGN PSKID = SKID0

```

```

111 0040      ASSIGN PPASS = FPASO
112      ;
113 0041      FOR ILOOP = 0 TO 39
114 0042      LET PPASS = 0
115 0043      LET PREV = 0
116 0045      IF PSTAT > 5 THEN PSTAT = 0      ; Reset current status if undefined
117 0046      LET PSKID = 0      ; Default previous status to no alarms
118 0047      LET PTYPE = 0      ; Default to no alarm
119 0048      IF ILOOP < 4 THEN      ; For DINs
120      BEGIN
121 0049      LET PTYPE = 0      ; Default to no alarm
122 0050      IF AND(PDIN:DIN_DFLG,3) THEN      ; Check for alarms if defined (must be red or yellow)
123      BEGIN
124 0051      LET PTYPE = 2      ; Default to yellow
125 0053      IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1      ; Reverse acting
126 0055      IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4      ; Set to red critical alarm
127      END
128 0056      STEP PDIN BY 1      ; Point to next DIN
129      END
130 0058      IF ILOOP > 34 THEN PTYPE = 1      ; Set to white alarm
131 0059      STEP PPASS BY 1
132 0060      STEP PREV BY 1
133 0061      STEP PSTAT BY 1
134 0062      STEP PSKID BY 1
135 0063      STEP PTYPE BY 1
136 0064      NEXT ILOOP
137      ;
138 0065      LET TYPE4 = 2
139 0066      LET TYPE6 = 4
140 0067      LET TYPE7 = 4
141 0068      LET TYPE10 = 6
142 0069      LET TYPE12 = 6
143 0070      LET TYPE14 = 6
144 0071      LET TYPE16 = 6
145 0072      LET TYPE22 = 4
146 0073      LET TYPE27 = 2
147 0074      LET TYPE29 = 6
148 0075      LET TYPE32 = 6
149      ;
150      ;
151 0076      BLET TYPE35 = 0      ; Delete White alarms on DDI-CA1-1
152 0077      BLET TYPE36 = 0      ; Delete White alarms on DDI-CA1-1
153 0078      LET TYPE37 = 1
154 0079      LET TYPE38 = 1
155 0080      LET TYPE39 = 1
156 0081      LET FRV0 = 0
157 0082      LET FRV1 = 0
158      ;
159      ;
160      ;
161 0083      LET FRV2 = 0
162 0084      LET FRV3 = 0      ; Delete White alarms on DDI-CA1-1
163 0085      BLET A10 = -0.08      ; Delete White alarms on DDI-CA1-1
164      ;
165      ;
166 0086      BLET A11 = 0.08      ; Delete White alarms on DDI-CA1-1
167      ;
168 0087      LET A12 = 15.0
169 0088      LET A13 = 15.0
170 0089      LET A14 = 5.0
171      ;
172      ;
173      ;
174      ;
175      ;
176      ;
177      ;
178      ;
179      ;
180      ;
181      ;
182      ;
183      ;
184      ;
185      ;
186      ;
187      ;
188      ;
189      ;
190      ;
191      ;
192      ;
193      ;
194      ;
195      ;
196      ;
197      ;
198      ;
199      ;
200      ;
201      ;
202      ;
203      ;
204      ;
205      ;
206      ;
207      ;
208      ;
209      ;
210      ;
211      ;
212      ;
213      ;
214      ;
215      ;
216      ;
217      ;
218      ;
219      ;
220      ;
221      ;
222      ;
223      ;
224      ;
225      ;
226      ;
227      ;
228      ;
229      ;
230      ;
231      ;
232      ;
233      ;
234      ;
235      ;
236      ;
237      ;
238      ;
239      ;
240      ;
241      ;
242      ;
243      ;
244      ;
245      ;
246      ;
247      ;
248      ;
249      ;
250      ;
251      ;
252      ;
253      ;
254      ;
255      ;
256      ;
257      ;
258      ;
259      ;
260      ;
261      ;
262      ;
263      ;
264      ;
265      ;
266      ;
267      ;
268      ;
269      ;
270      ;
271      ;
272      ;
273      ;
274      ;
275      ;
276      ;
277      ;
278      ;
279      ;
280      ;
281      ;
282      ;
283      ;
284      ;
285      ;
286      ;
287      ;
288      ;
289      ;
290      ;
291      ;
292      ;
293      ;
294      ;
295      ;
296      ;
297      ;
298      ;
299      ;
300      ;
301      ;
302      ;
303      ;
304      ;
305      ;
306      ;
307      ;
308      ;
309      ;
310      ;
311      ;
312      ;
313      ;
314      ;
315      ;
316      ;
317      ;
318      ;
319      ;
320      ;
321      ;
322      ;
323      ;
324      ;
325      ;
326      ;
327      ;
328      ;
329      ;
330      ;
331      ;
332      ;
333      ;
334      ;
335      ;
336      ;
337      ;
338      ;
339      ;
340      ;
341      ;
342      ;
343      ;
344      ;
345      ;
346      ;
347      ;
348      ;
349      ;
350      ;
351      ;
352      ;
353      ;
354      ;
355      ;
356      ;
357      ;
358      ;
359      ;
360      ;
361      ;
362      ;
363      ;
364      ;
365      ;
366      ;
367      ;
368      ;
369      ;
370      ;
371      ;
372      ;
373      ;
374      ;
375      ;
376      ;
377      ;
378      ;
379      ;
380      ;
381      ;
382      ;
383      ;
384      ;
385      ;
386      ;
387      ;
388      ;
389      ;
390      ;
391      ;
392      ;
393      ;
394      ;
395      ;
396      ;
397      ;
398      ;
399      ;
400      ;
401      ;
402      ;
403      ;
404      ;
405      ;
406      ;
407      ;
408      ;
409      ;
410      ;
411      ;
412      ;
413      ;
414      ;
415      ;
416      ;
417      ;
418      ;
419      ;
420      ;
421      ;
422      ;
423      ;
424      ;
425      ;
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432      ;
433      ;
434      ;
435      ;
436      ;
437      ;
438      ;
439      ;
440      ;
441      ;
442      ;
443      ;
444      ;
445      ;
446      ;
447      ;
448      ;
449      ;
450      ;
451      ;
452      ;
453      ;
454      ;
455      ;
456      ;
457      ;
458      ;
459      ;
460      ;
461      ;
462      ;
463      ;
464      ;
465      ;
466      ;
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473      ;
474      ;
475      ;
476      ;
477      ;
478      ;
479      ;
480      ;
481      ;
482      ;
483      ;
484      ;
485      ;
486      ;
487      ;
488      ;
489      ;
490      ;
491      ;
492      ;
493      ;
494      ;
495      ;
496      ;
497      ;
498      ;
499      ;
500      ;
501      ;
502      ;
503      ;
504      ;
505      ;
506      ;
507      ;
508      ;
509      ;
510      ;
511      ;
512      ;
513      ;
514      ;
515      ;
516      ;
517      ;
518      ;
519      ;
520      ;
521      ;
522      ;
523      ;
524      ;
525      ;
526      ;
527      ;
528      ;
529      ;
530      ;
531      ;
532      ;
533      ;
534      ;
535      ;
536      ;
537      ;
538      ;
539      ;
540      ;
541      ;
542      ;
543      ;
544      ;
545      ;
546      ;
547      ;
548      ;
549      ;
550      ;
551      ;
552      ;
553      ;
554      ;
555      ;
556      ;
557      ;
558      ;
559      ;
560      ;
561      ;
562      ;
563      ;
564      ;
565      ;
566      ;
567      ;
568      ;
569      ;
570      ;
571      ;
572      ;
573      ;
574      ;
575      ;
576      ;
577      ;
578      ;
579      ;
580      ;
581      ;
582      ;
583      ;
584      ;
585      ;
586      ;
587      ;
588      ;
589      ;
590      ;
591      ;
592      ;
593      ;
594      ;
595      ;
596      ;
597      ;
598      ;
599      ;
600      ;
601      ;
602      ;
603      ;
604      ;
605      ;
606      ;
607      ;
608      ;
609      ;
610      ;
611      ;
612      ;
613      ;
614      ;
615      ;
616      ;
617      ;
618      ;
619      ;
620      ;
621      ;
622      ;
623      ;
624      ;
625      ;
626      ;
627      ;
628      ;
629      ;
630      ;
631      ;
632      ;
633      ;
634      ;
635      ;
636      ;
637      ;
638      ;
639      ;
640      ;
641      ;
642      ;
643      ;
644      ;
645      ;
646      ;
647      ;
648      ;
649      ;
650      ;
651      ;
652      ;
653      ;
654      ;
655      ;
656      ;
657      ;
658      ;
659      ;
660      ;
661      ;
662      ;
663      ;
664      ;
665      ;
666      ;
667      ;
668      ;
669      ;
670      ;
671      ;
672      ;
673      ;
674      ;
675      ;
676      ;
677      ;
678      ;
679      ;
680      ;
681      ;
682      ;
683      ;
684      ;
685      ;
686      ;
687      ;
688      ;
689      ;
690      ;
691      ;
692      ;
693      ;
694      ;
695      ;
696      ;
697      ;
698      ;
699      ;
700      ;
701      ;
702      ;
703      ;
704      ;
705      ;
706      ;
707      ;
708      ;
709      ;
710      ;
711      ;
712      ;
713      ;
714      ;
715      ;
716      ;
717      ;
718      ;
719      ;
720      ;
721      ;
722      ;
723      ;
724      ;
725      ;
726      ;
727      ;
728      ;
729      ;
730      ;
731      ;
732      ;
733      ;
734      ;
735      ;
736      ;
737      ;
738      ;
739      ;
740      ;
741      ;
742      ;
743      ;
744      ;
745      ;
746      ;
747      ;
748      ;
749      ;
750      ;
751      ;
752      ;
753      ;
754      ;
755      ;
756      ;
757      ;
758      ;
759      ;
760      ;
761      ;
762      ;
763      ;
764      ;
765      ;
766      ;
767      ;
768      ;
769      ;
770      ;
771      ;
772      ;
773      ;
774      ;
775      ;
776      ;
777      ;
778      ;
779      ;
780      ;
781      ;
782      ;
783      ;
784      ;
785      ;
786      ;
787      ;
788      ;
789      ;
790      ;
791      ;
792      ;
793      ;
794      ;
795      ;
796      ;
797      ;
798      ;
799      ;
800      ;
801      ;
802      ;
803      ;
804      ;
805      ;
806      ;
807      ;
808      ;
809      ;
810      ;
811      ;
812      ;
813      ;
814      ;
815      ;
816      ;
817      ;
818      ;
819      ;
820      ;
821      ;
822      ;
823      ;
824      ;
825      ;
826      ;
827      ;
828      ;
829      ;
830      ;
831      ;
832      ;
833      ;
834      ;
835      ;
836      ;
837      ;
838      ;
839      ;
840      ;
841      ;
842      ;
843      ;
844      ;
845      ;
846      ;
847      ;
848      ;
849      ;
850      ;
851      ;
852      ;
853      ;
854      ;
855      ;
856      ;
857      ;
858      ;
859      ;
860      ;
861      ;
862      ;
863      ;
864      ;
865      ;
866      ;
867      ;
868      ;
869      ;
870      ;
871      ;
872      ;
873      ;
874      ;
875      ;
876      ;
877      ;
878      ;
879      ;
880      ;
881      ;
882      ;
883      ;
884      ;
885      ;
886      ;
887      ;
888      ;
889      ;
890      ;
891      ;
892      ;
893      ;
894      ;
895      ;
896      ;
897      ;
898      ;
899      ;
900      ;
901      ;
902      ;
903      ;
904      ;
905      ;
906      ;
907      ;
908      ;
909      ;
910      ;
911      ;
912      ;
913      ;
914      ;
915      ;
916      ;
917      ;
918      ;
919      ;
920      ;
921      ;
922      ;
923      ;
924      ;
925      ;
926      ;
927      ;
928      ;
929      ;
930      ;
931      ;
932      ;
933      ;
934      ;
935      ;
936      ;
937      ;
938      ;
939      ;
940      ;
941      ;
942      ;
943      ;
944      ;
945      ;
946      ;
947      ;
948      ;
949      ;
950      ;
951      ;
952      ;
953      ;
954      ;
955      ;
956      ;
957      ;
958      ;
959      ;
960      ;
961      ;
962      ;
963      ;
964      ;
965      ;
966      ;
967      ;
968      ;
969      ;
970      ;
971      ;
972      ;
973      ;
974      ;
975      ;
976      ;
977      ;
978      ;
979      ;
980      ;
981      ;
982      ;
983      ;
984      ;
985      ;
986      ;
987      ;
988      ;
989      ;
990      ;
991      ;
992      ;
993      ;
994      ;
995      ;
996      ;
997      ;
998      ;
999      ;
1000      ;

```

```

166      ;
167      ;
168      ;
169      0086      ANALOG = 0
170      0087      ILOOP = STAR
171      0088      FSTART = 1
172      0089      PTYPE = 0
173      0090      GOSUB ACK
174      0091      ON ERROR GOSUB ERRSUB
175      ;
176      ;
177      ;
178      0092      AGAIN: MARK STEP "WTUNTL",0,0
179      0093      STRNGO = "WAIT"
180      ;
181      ;
182      0094      IF (FABORT) THEN GOTO TERM      ; Wait until micro is selected or program aborted
183      0095      MARK STEP "RNNING",0,0      ; Terminate program if FABORT flag is set
184      0096      WAIT 1
185      0097      IF (FSNO) THEN      ; If alarm flag set
186      0098      IF STIMEO > 60 THEN      ; If alarm flag active for 1 minute
187      0099      BEGIN      ; Reset timer and turn off flag
188      0100      TIMEROFF STIMEO
189      0101      LET FSNO = 0
190      0102      LET ERRN = 0
191      0103      LET ERRL = 0
192      0104      END
193      ;
194      ;
195      ;
196      0105      IF CA1-ALARM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
197      0106      IF (PDEV:DEV_TMPO = 2 ) & AND(PDEV:DEV_ALMCT,32) THEN
198      0107      BEGIN
199      0108      LET PDEV:DEV_TMPO = 1 ; Put CA1-ALARM in ack state
200      0109      LET FRIP4 = 1      ; Set for unit alarm ack, system to graphic
201      0110      LET ACK = 1      ; Set for unit alarm ack, graphic to system
202      0111      CRIT
203      0112      LAMP ADI,SKDKEY,1      ; Turn lamp on
204      0113      END
205      ;
206      ;
207      ;
208      0114      BEGIN DINS:
209      0115      ASSIGN PDIN = 'PSH-CA111'
210      0116      ASSIGN PTYPE = TYPEO
211      0117      ASSIGN PREV = FRVO
212      0118      ASSIGN PSTAT = INTO
213      0119      ASSIGN PPASS = FPASO
214      0120      LET CURENT = 0      ; Used to set LAMP status
215      0121      LET LAST = 0
216      0122      LET ANALOG = 0
217      0123      LET NXT = 0
218      0124      LMPVAR = 0
219      0125      BITVAL = 0
220      0126      ;
221      0127      LOP: FOR ILOOP = 0 TO 3
222      0128      IF !SELECT() THEN GOTO AGAIN

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:34:11 24-Feb-95

Page 5

```

221 0121      IF (PTYPE > 0) THEN                                ; If some type of alarm
222      BEGIN                                                    ;be1
223      IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3          ; If EPN inhibited
224      IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5          ; If EPN is bad input
225      IF AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4             ; If EPN is disabled
226      IF (PSTAT < 3) THEN                                      ; If not inhibited
227      BEGIN                                                    ;be2
228      IF (PSTAT = 1) THEN
229      BEGIN                                                    ;be3
230      IF (PTYPE != 1) THEN
231      IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
232      IF (FTRIP4) THEN GOSUB ACK1
233      END                                                        ;en3
234      IF PSTAT = 0 THEN IF (PDIN:DIN_VAL = PREV) THEN GOSUB ALM
235      IF (PSTAT) THEN
236      BEGIN                                                    ;be4
237      IF (PDIN:DIN_VAL != PREV) THEN GOSUB CLR
238      NXT = PSTAT
239      IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
240      LAST = OR(LAST,NXT)
241      END                                                        ;en4
242      ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0
243      ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0
244      END                                                        ;en1
245      STEP PDIN BY 1
246      STEP PTYPE BY 1
247      STEP PREV BY 1
248      STEP PSTAT BY 1
249      STEP PPASS BY 1
250      NEXT ILOOP
251      ;
252      ;
253      ;
254      ;
255      ;
256      0153      LET STRNGO = "RUN"
257      0154      ASSIGN PAI = 'PI-CA1-11'
258      0155      ASSIGN PSKID = SKID4
259      0156      ASSIGN PTYPE = TYPE4
260      0157      FOR JLOOP = 4 TO 34
261      ;
262      0158      IF !SELECT() THEN GOTO AGAIN
263      IF (PTYPE) THEN                                          ; If some alarm exists
264      BEGIN                                                    ;be1
265      BITVAL = PAI:AI_ALMCT
266      IF !AND(BITVAL,16) THEN                                  ; If not inhibited
267      BEGIN                                                    ;be2
268      IF AND(BITVAL,15) THEN                                    ; If alarm exists and is current
269      BEGIN                                                    ;be3
270      IF AND(BITVAL,32) THEN NXT = 2                            ; If alarm is ack
271      ELSE
272      BEGIN                                                    ;be4
273      IF PSKID != BITVAL THEN
274      BEGIN                                                    ;be5
275      IF BITVAL < 3 THEN FALMON = 1
276      LET PSKID = BITVAL

```

```

276 0168          LET PDEV:DEV_TMPO = 2
277 0169          NXT = 1
278          END                                ;en5
279 0170          ELSE NXT = 1
280          END                                ;en4
281          END                                ;en3
282          ELSE                                ; Else there is no alarm
283          BEGIN
284 0171          NXT = 0
285 0172          LET PSKID = 0
286          END
287          END                                ;en2
288          ELSE                                ; Else alarm is inhibited
289          BEGIN
290 0173          NXT = 0
291 0174          LET PSKID = 0
292          END
293 0175          ANALOG = OR(ANALOG,NXT)
294          END                                ;en1
295 0176          STEP PSKID BY 1
296 0177          STEP PTYPE BY 1
297 0178          STEP PAI BY 1
298 0179          NEXT JLOOP
299          ;
300          ;
301          ;
302          ;
303          ;
304 0180  WHT:    WHITE = 0
305 0181          ASSIGN PALARM = A12
306 0182          ASSIGN PREV = FRV37
307 0183          ASSIGN PSTAT = INT37
308 0184          ASSIGN PTYPE = TYPE37
309 0185          ASSIGN PPASS = FPAS37
310          ;
311          ;
312          ;
313          ;
314          ;
315 0186          ILOOP = 35                                ; Delete White alarms on DD1-CA1-1
316 0187          ASSIGN PAI = 'DD1-CA1-1'
317          ;
318          ;
319          ;
320          ;
321          ;
322          ;
323 0194          GOSUB ANLG                                ; DD1-CA1-2 LOLO 35
324          ;
325          ;
326          ;
327          ;
328          ;
329          ;
330          ;
331          ;
332          ;
333          ;
334          ;
335          ;
336          ;
337          ;
338          ;
339          ;
340          ;
341          ;
342          ;
343          ;
344          ;
345          ;
346          ;
347          ;
348          ;
349          ;
350          ;
351          ;
352          ;
353          ;
354          ;
355          ;
356          ;
357          ;
358          ;
359          ;
360          ;
361          ;
362          ;
363          ;
364          ;
365          ;
366          ;
367          ;
368          ;
369          ;
370          ;
371          ;
372          ;
373          ;
374          ;
375          ;
376          ;
377          ;
378          ;
379          ;
380          ;
381          ;
382          ;
383          ;
384          ;
385          ;
386          ;
387          ;
388          ;
389          ;
390          ;
391          ;
392          ;
393          ;
394          ;
395          ;
396          ;
397          ;
398          ;
399          ;
400          ;
401          ;
402          ;
403          ;
404          ;
405          ;
406          ;
407          ;
408          ;
409          ;
410          ;
411          ;
412          ;
413          ;
414          ;
415          ;
416          ;
417          ;
418          ;
419          ;
420          ;
421          ;
422          ;
423          ;
424          ;
425          ;
426          ;
427          ;
428          ;
429          ;
430          ;
431          ;
432          ;
433          ;
434          ;
435          ;
436          ;
437          ;
438          ;
439          ;
440          ;
441          ;
442          ;
443          ;
444          ;
445          ;
446          ;
447          ;
448          ;
449          ;
450          ;
451          ;
452          ;
453          ;
454          ;
455          ;
456          ;
457          ;
458          ;
459          ;
460          ;
461          ;
462          ;
463          ;
464          ;
465          ;
466          ;
467          ;
468          ;
469          ;
470          ;
471          ;
472          ;
473          ;
474          ;
475          ;
476          ;
477          ;
478          ;
479          ;
480          ;
481          ;
482          ;
483          ;
484          ;
485          ;
486          ;
487          ;
488          ;
489          ;
490          ;
491          ;
492          ;
493          ;
494          ;
495          ;
496          ;
497          ;
498          ;
499          ;
500          ;
501          ;
502          ;
503          ;
504          ;
505          ;
506          ;
507          ;
508          ;
509          ;
510          ;
511          ;
512          ;
513          ;
514          ;
515          ;
516          ;
517          ;
518          ;
519          ;
520          ;
521          ;
522          ;
523          ;
524          ;
525          ;
526          ;
527          ;
528          ;
529          ;
530          ;
531          ;
532          ;
533          ;
534          ;
535          ;
536          ;
537          ;
538          ;
539          ;
540          ;
541          ;
542          ;
543          ;
544          ;
545          ;
546          ;
547          ;
548          ;
549          ;
550          ;
551          ;
552          ;
553          ;
554          ;
555          ;
556          ;
557          ;
558          ;
559          ;
560          ;
561          ;
562          ;
563          ;
564          ;
565          ;
566          ;
567          ;
568          ;
569          ;
570          ;
571          ;
572          ;
573          ;
574          ;
575          ;
576          ;
577          ;
578          ;
579          ;
580          ;
581          ;
582          ;
583          ;
584          ;
585          ;
586          ;
587          ;
588          ;
589          ;
590          ;
591          ;
592          ;
593          ;
594          ;
595          ;
596          ;
597          ;
598          ;
599          ;
600          ;
601          ;
602          ;
603          ;
604          ;
605          ;
606          ;
607          ;
608          ;
609          ;
610          ;
611          ;
612          ;
613          ;
614          ;
615          ;
616          ;
617          ;
618          ;
619          ;
620          ;
621          ;
622          ;
623          ;
624          ;
625          ;
626          ;
627          ;
628          ;
629          ;
630          ;
631          ;
632          ;
633          ;
634          ;
635          ;
636          ;
637          ;
638          ;
639          ;
640          ;
641          ;
642          ;
643          ;
644          ;
645          ;
646          ;
647          ;
648          ;
649          ;
650          ;
651          ;
652          ;
653          ;
654          ;
655          ;
656          ;
657          ;
658          ;
659          ;
660          ;
661          ;
662          ;
663          ;
664          ;
665          ;
666          ;
667          ;
668          ;
669          ;
670          ;
671          ;
672          ;
673          ;
674          ;
675          ;
676          ;
677          ;
678          ;
679          ;
680          ;
681          ;
682          ;
683          ;
684          ;
685          ;
686          ;
687          ;
688          ;
689          ;
690          ;
691          ;
692          ;
693          ;
694          ;
695          ;
696          ;
697          ;
698          ;
699          ;
700          ;
701          ;
702          ;
703          ;
704          ;
705          ;
706          ;
707          ;
708          ;
709          ;
710          ;
711          ;
712          ;
713          ;
714          ;
715          ;
716          ;
717          ;
718          ;
719          ;
720          ;
721          ;
722          ;
723          ;
724          ;
725          ;
726          ;
727          ;
728          ;
729          ;
730          ;
731          ;
732          ;
733          ;
734          ;
735          ;
736          ;
737          ;
738          ;
739          ;
740          ;
741          ;
742          ;
743          ;
744          ;
745          ;
746          ;
747          ;
748          ;
749          ;
750          ;
751          ;
752          ;
753          ;
754          ;
755          ;
756          ;
757          ;
758          ;
759          ;
760          ;
761          ;
762          ;
763          ;
764          ;
765          ;
766          ;
767          ;
768          ;
769          ;
770          ;
771          ;
772          ;
773          ;
774          ;
775          ;
776          ;
777          ;
778          ;
779          ;
780          ;
781          ;
782          ;
783          ;
784          ;
785          ;
786          ;
787          ;
788          ;
789          ;
790          ;
791          ;
792          ;
793          ;
794          ;
795          ;
796          ;
797          ;
798          ;
799          ;
800          ;
801          ;
802          ;
803          ;
804          ;
805          ;
806          ;
807          ;
808          ;
809          ;
810          ;
811          ;
812          ;
813          ;
814          ;
815          ;
816          ;
817          ;
818          ;
819          ;
820          ;
821          ;
822          ;
823          ;
824          ;
825          ;
826          ;
827          ;
828          ;
829          ;
830          ;
831          ;
832          ;
833          ;
834          ;
835          ;
836          ;
837          ;
838          ;
839          ;
840          ;
841          ;
842          ;
843          ;
844          ;
845          ;
846          ;
847          ;
848          ;
849          ;
850          ;
851          ;
852          ;
853          ;
854          ;
855          ;
856          ;
857          ;
858          ;
859          ;
860          ;
861          ;
862          ;
863          ;
864          ;
865          ;
866          ;
867          ;
868          ;
869          ;
870          ;
871          ;
872          ;
873          ;
874          ;
875          ;
876          ;
877          ;
878          ;
879          ;
880          ;
881          ;
882          ;
883          ;
884          ;
885          ;
886          ;
887          ;
888          ;
889          ;
890          ;
891          ;
892          ;
893          ;
894          ;
895          ;
896          ;
897          ;
898          ;
899          ;
900          ;
901          ;
902          ;
903          ;
904          ;
905          ;
906          ;
907          ;
908          ;
909          ;
910          ;
911          ;
912          ;
913          ;
914          ;
915          ;
916          ;
917          ;
918          ;
919          ;
920          ;
921          ;
922          ;
923          ;
924          ;
925          ;
926          ;
927          ;
928          ;
929          ;
930          ;
931          ;
932          ;
933          ;
934          ;
935          ;
936          ;
937          ;
938          ;
939          ;
940          ;
941          ;
942          ;
943          ;
944          ;
945          ;
946          ;
947          ;
948          ;
949          ;
950          ;
951          ;
952          ;
953          ;
954          ;
955          ;
956          ;
957          ;
958          ;
959          ;
960          ;
961          ;
962          ;
963          ;
964          ;
965          ;
966          ;
967          ;
968          ;
969          ;
970          ;
971          ;
972          ;
973          ;
974          ;
975          ;
976          ;
977          ;
978          ;
979          ;
980          ;
981          ;
982          ;
983          ;
984          ;
985          ;
986          ;
987          ;
988          ;
989          ;
990          ;
991          ;
992          ;
993          ;
994          ;
995          ;
996          ;
997          ;
998          ;
999          ;
1000         ;

```

```

331      ;
332      ;
333      ;
334      0195      LET FTRIP4 = 0
335      0196      MARK STEP "BELOW",0,0
336      0198      IF CURENT > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
337      0199      CURENT = OR(LAST,WHITE)
338      0200      CURENT = OR(CURENT,ANALOG)
339      0202      IF AND(CURENT,1)THEN LMPVAR = 2
340      0204      ELSE IF AND(CURENT,2)THEN LMPVAR = 1
341      0205      ELSE LMPVAR = 0
342      0206      LAMP ADI,SKDKEY,LMPVAR
343      0207      MARK STEP "BOTTOM",0,0
344      0208      NOCRIT
345      ;
346      ;
347      ;
348      0209      WF1:  VL1 = 'LI-CA1-1G':AI_MEAS
349      0210      VL2 = 'LI-CA1-2G':AI_MEAS
350      0211      ASSIGN PSPARO = VL1
351      0213      IF AND('SELECT-WF':DEV_STAT,4)THEN STEP PSPARO BY 1
352      0214      IF PSPARO > 19860 THEN
353      BEGIN
354      0216      IF PSPARO < 20472 THEN WF1 = ((PSPARO-19860)/612) * 7
355      0218      ELSE IF PSPARO < 21780 THEN WF1=((PSPARO-20472)/1308)*13 + 7
356      0220      ELSE IF PSPARO < 23088 THEN WF1=((PSPARO-21780)/2308)*10+ 20
357      0222      ELSE IF PSPARO <= 30000 THEN WF1=((PSPARO-23088)/6912)*70 + 30
358      END
359      0223      ELSE WF1 = 0
360      ;
361      0224      WF3:  VL3 = 'LI-CA1-3G':AI_MEAS
362      0225      IF VL3 > 760 THEN
363      BEGIN
364      0227      IF VL3 < 8013 THEN WF3 = ((VL3-760)/7253) * 19
365      0229      ELSE IF VL3 < 11377 THEN WF3 = ((VL3- 8013)/3364)*29 + 19
366      0231      ELSE IF VL3 < 19815 THEN WF3 = ((VL3-11377)/8438)*18 + 48
367      0233      ELSE IF VL3 < 20472 THEN WF3 = ((VL3-19860)/612)*2 + 66
368      0235      ELSE IF VL3 < 21780 THEN WF3 = ((VL3-20472)/1308)*3 + 68
369      0237      ELSE IF VL3 < 23088 THEN WF3 = ((VL3-21780)/1308)*3 + 71
370      0239      ELSE IF VL3 <= 30560 THEN WF3 = ((VL3-23088)/7472)*26 + 74
371      END
372      ;
373      ;
374      ;
375      ;
376      ;
377      0241      IF AND('CA1-ALARM':DEV_FLG,64)THEN PUTMANL 'CA1-ALARM'
378      0243      IF AND('SELECT-WF':DEV_FLG,64)THEN PUTMANL 'SELECT-WF'
379      0245      IF AND('SELECT-DI':DEV_FLG,64)THEN PUTMANL 'SELECT-DI'
380      0247      IF AND('HV-PDSPRY':DEV_FLG,64)THEN PUTMANL 'HV-PDSPRY'
381      ;
382      0248      GOTO AGAIN
383      ;
384      ;
385      ;
      END PROGRAM LOOP

```

```

386      ;      BEGIN ANALOG WHITE ALARM SUBROUTINE:
387      ;
388      ; ANLG: IF !SELECT() THEN GOTO AGAIN
389      ANLG: IF (PTYPE = 1) THEN      ; If some alarm
390      BEGIN
391      0250      BITVAL = PAI:A1_ALMCT      ;be1
392      0251      IF !(AND(BITVAL,16)) THEN      ; If not inhibited
393      BEGIN
394      0253      IF PSTAT = 3 THEN LET PSTAT = 0      ;be2
395      0254      IF (PREV) THEN      ; If rev acting, LOLO, or LO
396      BEGIN
397      0255      IF (PSTAT = 0) THEN      ; If not previously alarmed
398      BEGIN
399      0257      IF (PAI:A1_MEAS <= PALARM) THEN GOSUB ALM      ;be4
400      END      ;en4
401      ELSE
402      BEGIN      ;be5
403      0258      IF (PSTAT = 1) THEN      ; If previously alarmed
404      BEGIN      ;be6
405      0260      IF (FTRIP4) THEN GOSUB ACK1      ; If ACK flag set, then ack it
406      0263      IF PTYPE != 1 THEN IF AND(PAI:A1_ALMCT,32) THEN GOSUB ACK1
407      END      ;en6
408      0265      IF (PAI:A1_MEAS >= PALARM + PAI:A1_DEAD) THEN GOSUB CLR
409      END      ;en5
410      END      ;en3
411      ELSE      ; If fail-safe, HIHI, or HI
412      BEGIN      ;be10
413      0266      IF (PSTAT = 0) THEN
414      BEGIN      ;be11
415      0268      IF (PAI:A1_MEAS >= PALARM) THEN GOSUB ALM
416      END      ;en11
417      ELSE
418      BEGIN      ;be12
419      0269      IF (PSTAT = 1) THEN
420      BEGIN      ;be13
421      0271      IF (FTRIP4) THEN GOSUB ACK1
422      0274      IF PTYPE != 1 THEN IF AND(PAI:A1_ALMCT,32) THEN GOSUB ACK1
423      END      ;en13
424      0276      IF (PAI:A1_MEAS <= PALARM - PAI:A1_DEAD) THEN GOSUB CLR
425      END      ;en12
426      END      ;en10
427      0277      NXT = PSTAT
428      0279      IF (PPASS) & (PSTAT = 2) THEN NXT = 0
429      END
430      ELSE
431      BEGIN
432      0280      LET PSTAT = 3
433      0281      NXT = 0
434      END
435      0282      WHITE = OR(WHITE,NXT)
436      END
437      0283      ELSE PSTAT = 0
438      0284      STEP PALARM BY 1
439      0285      STEP PSTAT BY 1
440      0286      STEP PTYPE BY 1

```



```

441 0287          STEP PREV BY 1
442 0288          STEP PPASS BY 1
443 0289          RETURN
444          ;
445          ;      END ANALOG WHITE ALARM SUBROUTINE
446          ;
447          ;      BEGIN CLEAR, ALARM, ACKNOWLEDGE, MESSAGE, ERROR, AND TERMINATE SUBROUTINES:
448          ;
449 0290 CLR:      LET PSTAT = 0                      ; Int set to zero
450          ;
451          ;      If not white, then unack:
452          ;
453 0293          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
454 0294          LET FCLEAR = 1                      ; Clear flag
455 0295          GOSUB ACKK                          ; Print message
456 0296          RETURN
457          ;
458 0297 ALM:      LET PSTAT = 1                      ; Int set to one
459          ;
460          ;      If not white, then unack:
461          ;
462 0300          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
463 0301          LET PDEV:DEV_TMPO = 2
464          ;
465          ;      Turn audible alarm on if white or yellow:
466          ;
467 0303          IF AND(PTYPE,3) THEN FALMON = 1
468 0304          CRIT
469 0305          LAMP ADI,SKDKEY,2                      ; Blink the LAMP
470 0306          GOSUB ACKK                          ; Print message
471 0307          RETURN
472          ;
473 0308 ACK1:     LET PSTAT = 2                      ; Ack the alarm
474 0309          LET FACK = 1                        ; Set ACK flag for C program
475          ;
476          ;      ALARM TYPER STRINGS:
477          ;
478 0311 ACKK:     IF ILOOP = 0 THEN STRNGO = "CA1 VESSEL PRESSURE HI"
479 0313          IF ILOOP = 1 THEN STRNGO = "CA1-1 WEIGHT FACTOR HI"
480 0315          IF ILOOP = 2 THEN STRNGO = "CA1-2 WEIGHT FACTOR HI"
481 0317          IF ILOOP = 3 THEN STRNGO = "LOWER DE-ENTRNMNT PAD DP HI"
482          ;      IF ILOOP = 35 THEN STRNGO = "(CA1-1)-(CA1-2) SPG DIFF LOLO"      ; Delete White on DDI-CA1-1
483          ;      IF ILOOP = 36 THEN STRNGO = "(CA1-1)-(CA1-2) SPG DIFF HIHI"      ; Delete White on DDI-CA1-1
484          ;
485 0319          IF ILOOP = 37 THEN STRNGO = "F-H-1 RW FILTER DELTA P HIGH"
486 0321          IF ILOOP = 38 THEN STRNGO = "F-H-2 RW FILTER DELTA P HIGH"
487 0323          IF ILOOP = 39 THEN STRNGO = "F-H-3 RW FILTER DELTA P HIGH"
488 0325          IF ILOOP = 17 THEN STRNGO = "START CA1BATCH PGM "
489 0327          IF ILOOP = 18 THEN STRNGO = "CA1BATCH WRONG UNIT "
490 0329          IF ILOOP = 19 THEN STRNGO = "TERMINATE CA1BATCH "
491          ;
492          ;      If acking the alarm, then:
493          ;
494 0330 AMSG:     IF PTYPE = 1 THEN
495          BEGIN

```

```

496           ; If white alarm, print:
497           ;
498 0332       IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNGO
499 0333       IF (FACK) THEN
500           BEGIN
501           ;
502           ; If white, print it is ack and reset the flag:
503           ;
504 0334       PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
505 0335       LET FACK = 0
506           END
507           END
508           ;
509           ; If Clear, print it is clear and reset the flag:
510           ;
511 0336       IF (FCLEAR) THEN
512           BEGIN
513 0337       PRINT ADI,"%p ALARM CLEAR",STRNGO
514 0338       LET FCLEAR = 0
515           END
516           ;
517           ; If Startup, then print startup message:
518           ;
519 0339       IF (FSTART) THEN
520           BEGIN
521 0340       LET FSTART = 0
522 0341       PRINT ADI,"%p",STRNGO
523           END
524 0342       RETURN
525           ;
526           ; If Error - print message, start timer, and set flag:
527           ;
528 0343       ERRSUB: IF ERRN != ERRNUM() THEN
529           BEGIN
530 0344       TIMERON STIMEO
531 0345       TIMERCLR STIMEO
532 0346       LET FSN0 = 1
533 0347       LET ERRN = ERRNUM()
534 0348       LET ERRL = ERRLIN()
535 0349       PRINT ADI,"ERROR # %i AT LINE %i",errn,errl
536           END
537 0350       ERSCLR
538 0351       GOTO AGAIN
539           ;
540           ; If FABORT = 1, then abort program:
541           ;
542           ; TERM: LET FABORT = 0
543 0352       TERM: LET STRNGO = "CA1BATCH TERMINATED  "
544 0353       LET ILOOP = STAR + 2
545 0354       LET FSTART = 1
546 0355       LET STRNGO = "TERM"
547 0356       GOSUB ACKK
548           END

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 6960 Bytes, 435 Pages

21 Forward Reference(s)

PB1BATCH

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

06:47:01 16-May-95

Page 1

```

1      ; P B 1 B A T C H . S E Q                                VERSION 4
2      ;
3      ; Function:      Sequence program for Lamp function,
4      ;                message display on Graphic number 012,
5      ;                and printer messages.
6      ;
7      ; Written By      : R. N. Kersey
8      ; Date            : 05-31-92
9      ; Facility        : 242-A Evaporator Bldg.
10     ; Machine         : VAX 4104-A
11     ; Op. System      : VAX/VMS 5.5-2AH
12     ; D/3 Version     : 6.3-2 BASELINE
13     ; Compiler        : SIC
14     ; Build command   : @I320,20PB1BATCH
15     ;
16     ; UNIT = 7, SKDKEY = 30
17     ;
18     ; Assoc. UPF:      [300,1]_PROCESS.____ PROCESS.LST
19     ; Assoc. RPF:      [300,1]_242A.____ 242A.LST
20     ; SDB:            [300,1]PCMOA.SDB PCMOASDB.CMD
21     ;
22     ; POINTER SDB VAR. FUNCTION
23     ; -----
24     ; PSTAT INTn Current EPN alarm state is an integer tied to
25     ;                graphic flashing string of value:
26     ;                (n=0-100) 0 = previously not alarmed
27     ;                1 = previously alarmed, not ack
28     ;                2 = previously alarmed, is ack
29     ;                3 = alarm is inhibited
30     ;                4 = input is off scan
31     ;                5 = input is bad
32     ; LEAKn Special INT variable for multiple white alarms on an EPN
33     ; 'EPN' ---
34     ; PTYPE TYPEn alarm type 0 = none, 1 = white, 2 = yellow,
35     ;                3 = yellow and yellow, 4 = red,
36     ;                6 = yellow and red
37     ; PREV FRVn Reverse acting (not fail safe) flag (1=set)
38     ; PSKID SKIDn Used for analog alarms
39     ; PPASS FPASn Turn lamp off when alarm is ACKnowledged
40     ; PALARM FALMn Analog alarm flag
41     ; --- FWING Unused error flag
42     ;
43     ; MODIFICATION LOG:
44     ;
45     ; 30-DEC-92 Fixed white alarm print message. M. Teats
46     ;
47     ; 07-JUN-93 Added comments and baselined for M. Teats
48     ; D/3 version 3.3. CCN# 93JUN01-01.
49     ; 28-JUL-94 Added white alarm for PDI-CA1-3 J. Berger
50     ; per EE-94-00019.
51     ;
52     ; 15-DEC-94 Revised header to indicate the new VMS J. Berger
53     ; system. Rebaselined document. Removed
54     ; obsolete Select() statement.
55     ;

```

```

56      ;      26-MAR-95      Revised program to add the effected EPNs J. Berger
57      ;
58      ;      for the Condensate Recycle Loop per ECN
59      ;      620390 and WP EE-95-00036. (JFB01)
60      ;
61      ;      27-APR-95      Revised program to change YS-PB1-1 to a J. Berger
62      ;      yellow alarm. See WP EE-95-00006 and
63      ;      ECN 622499.
64      ;      END MOD LOG
65      ;
66      UNIT      PB1
67      ;
68      COMMON PB2
69      0000      NOP
70      0001      STAR = 24      ; Start message
71      0002      LET SKDKEY = 30      ; SKID Key
72      0003      LET UNIT1 = 7      ; SEQ Unit number
73      0004      ASSIGN PDEV = 'PB1-ALARM'      ; Alarm device
74      0005      ASSIGN PTYPE = TYPE24
75      0006      LET ADI = 0      ; Alarm destination index
76      0007      LET FSN0 = 0      ; No alarm yet
77      0008      TIMEROFF STIME0      ; Alarm timer off
78      0009      TIMERCLR STIME0      ; Alarm timer clear
79      0010      STRNG0 = " "      ; Blank 30 character string
80      0011      LET ERRN = 0      ; Set error number to zero
81      0012      LET ERRL = 0      ; Set error line to zero
82      0013      LET FTRIP4 = 0      ; FTRIP4 = 1 when audible horn is to be turned on
83      0014      GETUNIT TRIP      ; Which sequence unit is program running in
84      0015      IF TRIP != UNIT1 THEN      ; Abort if running in wrong seq. unit
85          BEGIN
86      0016          FSTART = 1
87      0017          ILOOP = STAR + 1
88      0018          PTYPE = 0
89      0019          GOSUB ACKK
90      0020          GOTO TERM
91      0021          END
92      ;      IF !SELECT() THEN GOTO AGAIN
93      ;
94      ;      BEGIN INITIALIZING
95      ;
96      0021      LET STRNG0 = "INIT"
97      0022      INIT:      LET FABORT = 0      ; If FABORT = 1 then abort program
98      0023      LET FSTART = 0      ; Program start flag = 0
99      0024      LAMP ADI,SKDKEY,0      ; Turn off the annunciator key
100     0025      LET LSG0 = " "      ; Clear the string variables
101     0026      LET LSG1 = " "
102     0027      LET LSG2 = " "
103     0028      LET LSG3 = " "
104     0029      LET LSG4 = " "
105     0030      LET PDEV:DEV_TMP0 = 1      ; Set alarm device to non-alarm state
106     0031      IF AND(PDEV:DEV_ALMCT,16) THEN      ; If the alarm device is inhibited, remove the inhibit
107     0032      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
108     0033      IF AND(PDEV:DEV_ALMCT,32) THEN      ; If the alarm device is ack, clear it
109     0034      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
110     ;

```

```

111      ;      First, set all flags and integers:
112      ;
113      0035      ASSIGN PDIN = 'YS-PC-105'
114      0036      ASSIGN PPASS = FPASO
115      0037      ASSIGN PREV = FRVO
116      0038      ASSIGN PSTAT = INTO
117      0039      ASSIGN PSKID = SKIDO
118      0040      ASSIGN PTYPE = TYPEO
119      ;
120      0041      FOR ILOOP = 0 TO 29                      ; JFB01
121      0042      LET PPASS = 0
122      0043      LET PREV = 0
123      0045      IF PSTAT > 5 THEN LET PSTAT = 0          ; Reset current status if undefined
124      0046      LET PSKID = 0                          ; Default previous status to no alarms
125      0047      LET PTYPE = 0                          ; Default to no alarm
126      0048      IF ILOOP < 12 THEN                      ; For DINs JFB01
127      BEGIN
128      0049      LET PTYPE = 0
129      0050      IF AND(PDIN:DIN_DFLG,3) THEN              ; Check for alarms if defined (must be red or yellow)
130      BEGIN
131      0051      LET PTYPE = 2                          ; Default to yellow
132      0053      IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1 ; Reverse acting
133      0055      IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4 ; Set to red critical alarm
134      END
135      END
136      0056      ELSE LET PTYPE = 4                      ; For AIs, default to red
137      0058      IF ILOOP < 11 THEN STEP PDIN BY 1        ; Point to next DIN JFB01
138      0059      STEP PPASS BY 1
139      0060      STEP PREV BY 1
140      0061      STEP PSTAT BY 1
141      0062      STEP PSKID BY 1
142      0063      STEP PTYPE BY 1
143      0064      NEXT ILOOP
144      ;
145      0065      LET FRV2 = 1
146      0066      LET FRV4 = 1
147      0067      LET FRV5 = 0
148      0068      LET FRV27 = 0
149      ;
150      0069      LET TYPE3 = 2                          ; JFB02
151      0070      LET TYPE10 = 0
152      0071      LET TYPE11 = 0
153      0072      LET TYPE18 = 0
154      0073      LET TYPE22 = 0
155      0074      LET TYPE24 = 0
156      0075      LET TYPE30 = 2                          ; JFB01
157      0076      LET TYPE27 = 1                          ; JFB01
158      0077      LET TYPE29 = 6                          ; JFB01
159      0078      LET INT30 = 0                          ; JFB01
160      0079      LET AI15 = 15                          ; JFB01
161      ;
162      ;      END INITIALIZING
163      ;
164      ;      SEND PROGRAM START MESSAGE
165      ;

```

```

166 0080      ILOOP = STAR
167 0081      FSTART = 1
168 0082      PTYPE = 0
169 0083      CURENT = 0
170 0084      LAST = 0
171 0085      ANALOG = 0
172 0086      WHITE = 0
173 0087      GOSUB ACKK
174 0088      ON ERROR GOSUB ERRSUB
175      ;
176      ;
177      ;
178 0089  AGAIN:  MARK STEP "WTUNTIL",0,0
179      ;          WAITUNTIL SELECT() ; FABORT      ; Wait until micro is selected or program aborted
180      ;          IF (FABORT) THEN GOTO TERM      ; Terminate program if abort flag is set
181 0090      ;          MARK STEP "RNNING",0,0
182 0091      ;          WAIT 1
183 0092      ;          IF (FSNO) THEN              ; If alarm flag set
184 0093      ;              IF STIME0 > 60 THEN      ; If alarm flag active for 1 minute
185      ;                  BEGIN                    ; Reset timer and turn off flag
186 0094      ;                      TIMEROFF STIME0
187 0095      ;                      TIMERCLR STIME0
188 0096      ;                      LET FSNO = 0
189 0097      ;                      LET ERRN = 0
190 0098      ;                      LET ERRL = 0
191      ;                      END
192      ;
193      ;
194      ;
195 0099      IF (PDEV:DEV_TMP0 = 2) & AND(PDEV:DEV_ALMCT,32) THEN
196      BEGIN
197 0100      LET PDEV:DEV_TMP0 = 1 ; Put PB1-ALARM in ack state
198 0101      LET FTRIP4 = 1      ; Set for Unit alarm ack, system to graphic
199 0102      LET ACK = 1          ; Set for Unit alarm ack, graphic to system
200 0103      CRIT
201 0104      LAMP ADI,SKDKEY,1    ; Turn LAMP on
202      END
203      ;
204      ;
205      ;
206 0105      BEGIN DINS:
207 0106      ASSIGN PDIN = 'YS-PC-105'      ; Assign pointer to first DIN
208 0107      ASSIGN PTYPE = TYPE0
209 0108      ASSIGN PREV = FRV0
210 0109      ASSIGN PSTAT = INTO
211 0110      ASSIGN PPASS = FPAS0
212 0111      LAST = 0
213 0112      NXT = 0
214 0113      LMPVAR = 0
215 0114      BITVAL = 0
216      ;
217      ;
218 0115      LOP:  FOR ILOOP = 0 TO 11
219      ;          IF !SELECT() THEN GOTO AGAIN
220 0117      ;          IF ( PTYPE > 0 ) THEN      ; If some type of alarm
221      ;              BEGIN                      ;be1
222      ;              IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3 ; If EPN inhibited

```



```

221 0119      IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5          ; If EPN is bad input
222 0121      IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4          ; If EPN is disabled
223 0122      IF ( PSTAT < 3 ) THEN                                ; If not inhibited
224          BEGIN                                                ;be2
225 0123          IF (PSTAT = 1) THEN
226              BEGIN                                            ;be3
227 0124              IF (PTYPE != 1) THEN
228 0126                  IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
229 0128                  IF (FTRIP4) THEN GOSUB ACK1
230              END                                              ;en3
231 0131      IF PSTAT = 0 THEN IF (PDIN:DIN_VAL = PREV) THEN GOSUB ALM
232 0132      IF (PSTAT) THEN
233          BEGIN                                                ;be4
234 0134          IF (PDIN:DIN_VAL != PREV) THEN GOSUB CLR
235 0135          NXT = PSTAT
236 0137          IF (PPASS) & (PSTAT = 2) THEN NXT = 0
237 0138          LAST = OR(LAST,NXT)
238          END                                                  ;en4
239          END                                                  ;en2
240 0140      ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0
241          END                                                  ;en1
242 0142      IF ILOOP < 11 THEN STEP PDIN BY 1
243 0143      STEP PTYPE BY 1
244 0144      STEP PREV BY 1
245 0145      STEP PSTAT BY 1
246 0146      STEP PPASS BY 1
247 0147      NEXT ILOOP
248      ;
249      ;
250      ;
251      ;
252      ;
253 0148      ANALOG = 0
254 0149      ASSIGN PAI = 'KY-PB1-1'
255 0150      ASSIGN PSKID = SKID0
256 0151      FOR JLOOP = 0 TO 17
257          IF ISELECT() THEN GOTO AGAIN
258 0152      IF (PTYPE) THEN                                ; If some alarm exists
259          BEGIN                                                ;be1
260 0153          BITVAL = PAI:AI_ALMCT
261 0154          IF !AND(BITVAL,16) THEN                                ; If not inhibited
262              BEGIN                                            ;be2
263 0155          IF AND(BITVAL,15) THEN                                ; If alarm exists and is current
264              BEGIN                                            ;be3
265 0157          IF AND(BITVAL,32) THEN NXT = 2                                ; If alarm is ack
266              ELSE                                            ; If alarm is unack
267              BEGIN                                            ;be4
268 0158          IF PSKID !=BITVAL THEN
269              BEGIN                                            ;be5
270 0160              IF BITVAL < 3 THEN FALMON = 1
271 0161              LET PSKID = BITVAL
272 0162              LET PDEV:DEV_TMPO = 2
273 0163              NXT = 1
274              END
275 0164          ELSE NXT = 1

```

```

276                END                                ;en4
277                END                                ;en3
278            ELSE                                     ; Else there is no alarm
279                BEGIN
280                0165        NXT = 0
281                0166        LET PSKID = 0
282                END
283            END                                     ;en2
284        ELSE                                     ; Else alarm is inhibited
285            BEGIN
286            0167        NXT = 0
287            0168        LET PSKID = 0
288            END
289            0169        ANALOG = OR(ANALOG,NXT)
290            END                                     ;en1
291            0170        STEP PSKID BY 1
292            0171        STEP PTYPE BY 1
293            0172        STEP PAI BY 1
294            0173        STEP PSTAT BY 1
295            0174        NEXT JLOOP
296        ;
297        ;
298        ;
299        BEGIN DEVICE ALARMS:
300
301        0175    JGV:    ILOOP = 30                    ; JGV-SUMP
302        0176        ASSIGN PSTAT = INT30
303        0177        ASSIGN PTYPE = TYPE30
304        ;
305        0178        IF ISELECT() THEN GOTO AGAIN        ; If some type of alarm
306                BEGIN                                ;be1
307                0180        IF AND('JGV-SUMP':DEV_ALMCT,16) THEN PSTAT = 3 ; If EPW inhibited
308                0181        IF (PSTAT < 3) THEN        ; If not inhibited
309                    BEGIN                                ;be2
310                    0182        IF (PSTAT = 1) THEN
311                        BEGIN                                ;be3
312                        0183        IF (PTYPE != 1) THEN
313                        0185        IF AND('JGV-SUMP':DEV_ALMCT,32) THEN GOSUB ACK1
314                        0187        IF (FTRIP4) THEN GOSUB ACK1
315                        END                                ;en3
316                    ;
317                    ; If alarm (ON 8+ MIN) is active and is previously not alarmed:
318                    ;
319                    0189        IF PSTAT = 0 THEN IF AND('JGV-SUMP':DEV_STAT,8) THEN
320                        BEGIN                                ;be4
321                        0190        LET PSTAT = 1
322                        0193        IF PTYPE != 1 THEN IF AND('JGV-SUMP':DEV_ALMCT,32) THEN 'JGV-SUMP':DEV_ALMCT = XOR('JGV-SUMP':DEV_ALMCT,32)
323                        0194        LET PDEV:DEV_TMP0 = 2
324                        0196        IF AND(PTYPE,3) THEN FALMON = 1
325                        0197        CRIT
326                        0198        LAMP AD1,SKDKEY,2
327                        END                                ;en4
328                        0199        IF (PSTAT) THEN        ; If previously alarmed
329                            BEGIN                                ;be5
330                            0200        IF IAND('JGV-SUMP':DEV_STAT,8) THEN        ; If not alarming now, then clear

```

```

331          BEGIN                                ;be6
332 0201      PSTAT = 0
333 0202      GOSUB ACKK
334          END                                ;en6
335 0203      NXT = PSTAT
336 0205      IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
337 0206      LAST = OR(LAST,NXT)
338          END                                ;en5
339          END                                ;en2
340 0208      ELSE IF IAND('JGV-SUMP':DEV_ALMCT,16) THEN PSTAT = 0
341          END                                ;en1
342      ;
343      ;
344      ;
345      ;
346      ;
347      ;
348 0209  WHT:  WHITE = 0
349 0210      ASSIGN PALARM = A115
350 0211      ASSIGN PREV = FRV27
351 0212      ASSIGN PSTAT = INT27
352 0213      ASSIGN PTYPE = TYPE27
353 0214      ASSIGN PPASS = FPAS27
354 0215      ILOOP = 31
355 0216      ASSIGN PA1 = 'PDI-CA1-3'
356          GOSUB ANLG                                ; PDICA1-3    30
357      ;
358      ;
359      ;
360      ;
361      ;
362      ;
363      ;
364      ;
365 0217  ANLG: IF ISELECT() THEN GOTO AGAIN
366      ;
367      ;
368 0218      BEGIN                                ;BE1
369 0219      BITVAL = PA1:A1_ALMCT
370          IF I(AND(BITVAL,16)) THEN                ; IF NOT INHIBITED
371      BEGIN                                ;BE2
372          IF PSTAT = 3 THEN LET PSTAT = 0
373          IF (PREV) THEN                            ; IF REV ACTING, LOLO, OR LO
374      BEGIN                                ;BE3
375          IF (PSTAT = 0) THEN                        ; IF NOT PREVIOUSLY ALARMED
376      BEGIN                                ;BE4
377          IF (PA1:A1_MEAS <= PALARM) THEN GOSUB ALM
378          END
379      ELSE
380      BEGIN                                ;BE5
381          IF (PSTAT = 1) THEN                        ; IF PREVIOUSLY ALARMED
382      BEGIN                                ;BE6
383          IF (FTRIP4) THEN GOSUB ACK1 ;IF ACK FLAG SET, THEN ACK IT
384          IF PTYPE !=1 THEN IF AND(PA1:A1_ALMCT,32) THEN GOSUB ACK1
385          END                                ;EN5
386          IF (PA1:A1_MEAS >= PALARM + PA1:A1_DEAD) THEN GOSUB CLR
387          END                                ;EN3

```

```

386      END
387      ELSE
388          BEGIN
389              IF (PSTAT = 0) THEN
390                  BEGIN
391                      IF (PA1:AI_MEAS >= PALARM) THEN GOSUB ALM
392                      END
393                  ELSE
394                      BEGIN
395                          IF (PSTAT = 1) THEN
396                              BEGIN
397                                  IF (FTRIP4) THEN GOSUB ACK1
398                                  IF PTYP4 = 1 THEN IF AND(PA1:AI_ALMCT,32) THEN GOSUB ACK1
399                                  END
400                                  IF (PA1:AI_MEAS <= PALARM - PA1:AI_DEAD) THEN GOSUB CLR
401                                  END
402                                  END
403                                  NXT = PSTAT
404                                  IF (PPASS) & (PSTAT = 2) THEN NXT = 0
405                                  END
406                                  ELSE
407                                      BEGIN
408                                          LET PSTAT = 3
409                                          NXT = 0
410                                          END
411                                          WHITE = OR(WHITE,NXT)
412                                          END
413                                          ELSE PSTAT = 0
414                                          STEP PALARM BY 1
415                                          STEP PSTAT BY 1
416                                          STEP PTYP4 BY 1
417                                          STEP PREV BY 1
418                                          STEP PPASS BY 1
419                                          RETURN
420
421      ;
422      ;
423      ;
424      ;
425      ;
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432      ;
433      ;
434      ;
435      ;
436      ;
437      ;
438      ;
439      ;
440      ;

```

END ANALOG WHITE ALARM SUBROUTINE

```

FTRIP4 = 0
IF LAST > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
CURRENT = OR(LAST,ANALOG)
CURRENT = OR(CURRENT,WHITE)
IF AND(CURRENT,1) THEN LMPVAR = 2
ELSE IF AND(CURRENT,2) THEN LMPVAR = 1
ELSE LMPVAR = 0
LAMP ADI,SKOKEY,LMPVAR
MARK STEP "BOTTOM",0,0
NOCRT

```

BEGIN LOGIC FOR GRAPHICS:

```

SA0 = AND('PB-1':DEV_STAT,1) ; CF-OFF
SA1 = AND('PB-1':DEV_STAT,2) ; CF-ON
SA2 = AND('PB-1':DEV_STAT,4) ; SD-RESET
SA3 = AND('PB-1':DEV_STAT,8) ; CF-JOGNG
SA4 = AND('PB-1':DEV_STAT,16) ; SHUTDOWN

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

06:47:01 16-May-95

Page 9

```

441 0272      SA5 = AND('PB-1':DEV_STAT,32)      ; INTERLOCK
442      ; SA6 = AND('PB-1':DEV_STAT,64)      ; STOPPING
443      ; SA7 = AND('PB-1':DEV_STAT,128)     ; STARTING
444      ;
445 0274      IF AND('PB1-ALARM':DEV_FLG,64) THEN PUTMANL 'PB1-ALARM'
446 0276      IF AND('PB-1':DEV_FLG,64) THEN PUTMANL 'PB-1'
447 0278      IF AND('PB1WFINKL':DEV_FLG,64) THEN PUTMANL 'PB1WFINKL'
448 0280      IF AND('PB1-BYPAS':DEV_FLG,64) THEN PUTMANL 'PB1-BYPAS'
449 0282      IF AND('BOT-DUMP':DEV_FLG,64) THEN PUTMANL 'BOT-DUMP'
450 0284      IF AND('HV-SUMP-1':DEV_FLG,64) THEN PUTMANL 'HV-SUMP-1'
451 0286      IF AND('JGV-SUMP':DEV_FLG,64) THEN PUTMANL 'JGV-SUMP'
452      ;
453 0287      AT0 = AND('PB1WFINKL':DEV_TMP0,1)      ; LI-CA1-3
454 0288      AT1 = AND('PB1WFINKL':DEV_TMP0,2)      ; LI-CA1-1
455 0289      AT2 = AND('PB1WFINKL':DEV_TMP0,4)      ; LI-CA1-2
456 0290      AT3 = AND('PB1WFINKL':DEV_TMP0,8)      ; PB1WFINKL INTERLOCK
457 0291      AT4 = AND('PB1WFINKL':DEV_TMP0,16)     ; BYPAS OR NO INTERLOCK
458      ; AT12= AND(AT0,AT4)                      ; LI-CA1-3 CONTROLS PB1WFINKL
459 0292      AT5 = AND('PB1-BYPAS':DEV_TMP0,32)     ; timer running
460 0293      AT6 = AND('PB1-BYPAS':DEV_TMP0,64)     ; time out
461 0294      AT8 = AND('PB1-BYPAS':DEV_TMP0,256)    ; PERMISSIVE ON
462 0295      AT23 = AND('PB-1':DEV_TMP1,128)
463 0296      AT24 = AND('PB-1':DEV_TMP1,256)
464 0297      AT25 = AND('PB-1':DEV_TMP1,512)
465 0298      AT27 = AND('HV-CA1-10':DEV_TMP1,2048)   ; JFB01 $T27
466 0299      AT28 = AND('HV-CA1-10':DEV_TMP1,4096)   ; JFB01 $T28
467 0300      AT29 = AND('HV-CA1-10':DEV_TMP1,8192)   ; JFB01 $T29
468 0301      AT31= AND('PB-1':DEV_TMP1,32768)      ; PB-1 INTERLOCKS
469      ;
470 0302      BT26 = AND('P-C-105':DEV_TMP1,1024)
471 0303      BT27 = AND('P-C-105':DEV_TMP1,2048)
472 0304      BT28 = AND('P-C-105':DEV_TMP1,4096)
473 0305      BT29 = AND('P-C-105':DEV_TMP1,8192)
474 0306      BT30 = AND('P-C-105':DEV_TMP1,16384)
475      ;
476 0307      CT0 = 'YS-PB1-1':DIN_VAL
477 0308      CT1 = AND('PB-1':DEV_TMP0,2)
478      ;
479 0309      GT8 = AND(PB2GLOBAL:DEV_TMP0,256)
480 0310      LEAK0 = GT8
481      ;
482 0311      SB4 = AND('P-C-105':DEV_STAT,16)
483 0312      SB5 = AND('P-C-105A':DEV_STAT,16)
484      ;
485      ; LEAK0 CONTROLS THE COLOR OF LINE BETWEEN LI-CA1-1/2 AND LI-CA1-3:
486      ; 0 = GREEN, 1 = RED, 2 - 7 = BLACK. IF AT4 THEN LINE = GREEN
487      ; ELSE IF !AT12 THEN LINE = RED, ELSE LINE = BLACK.
488      ;
489 0314      IF AT12 THEN LEAK0 = 0      ; Wfs are OK or BYPASS ON and WF-3 not LO (GREEN)
490 0316      ELSE IF !AT0 THEN LEAK0 = 2 ; WF-3 LO (BLACK)
491 0317      ELSE LEAK0 = 1      ; INTERLOCK and WF-3 not LO (RED)
492      ;
493      ; END LOGIC FOR GRAPHICS
494      ;
495 0318      GOTO AGAIN

```

```

496      ;
497      ;      E N D   P R O G R A M   L O O P
498      ;
499      ;      BEGIN CLEAR, ALARM, ACKNOWLEDGE, MESSAGE, ERROR, AND TERMINATE SUBROUTINES:
500      ;
501 0319 CLR:  LET PSTAT = 0                                ; Int set to zero
502      ;
503      ;      If not white, then unack:
504      ;
505 0322      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
506 0323      LET FCLEAR = 1                                ; Clear flag
507 0324      GOSUB ACKK                                    ; Print message
508 0325      RETURN
509      ;
510 0326 ALM:  LET PSTAT = 1                                ; Int set to one
511      ;
512      ;      If not white, then unack:
513      ;
514 0329      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
515      ;
516      ;      Turn audible alarm on if white or yellow:
517      ;
518 0330      PDEV:DEV_TMPO = 2
519 0332      IF AND(PTYPE,3) THEN FALMON = 1
520 0333      CRIT
521 0334      * LAMP ADI,SKOKEY,2
522 0335      GOSUB ACKK                                    ; Print message
523 0336      RETURN
524      ;
525 0337 ACK1: LET PSTAT = 2                                ; Ack the alarm
526 0338      LET FACK = 1                                  ; Set ACK flag for C program
527      ;
528      ;      ALARM TYPER STRINGS:
529      ;
530 0340 ACKK: IF ILOOP = 0 THEN STRNGO = "P-C-105 SEAL WATER PUMP ON"
531 0342      IF ILOOP = 1 THEN STRNGO = "P-C-105A SEAL WATER PUMP ON"
532 0344      IF ILOOP = 2 THEN STRNGO = "CA1-7 POT DUMP VALVE OPEN"
533 0346      IF ILOOP = 3 THEN STRNGO = "P-B-1 PUMP MOTOR CONFIRMED"
534 0348      IF ILOOP = 4 THEN STRNGO = "CA1-7 POT DUMP VALVE CLOSED"
535 0350      IF ILOOP = 5 THEN STRNGO = "SLURRY SAMPLER ENCLOSURE LEAK"
536 0352      IF ILOOP = 6 THEN STRNGO = "HV-CA1-8 RAW WATER VALVE OPEN"
537 0354      IF ILOOP = 7 THEN STRNGO = "HV-CA1-8 RAW WATER VALVE CLOS"
538 0356      IF ILOOP = 8 THEN STRNGO = "CA1-9 POT DUMP VALVE CLOSED"
539 0358      IF ILOOP = 9 THEN STRNGO = "CA1-9 POT DUMP VALVE OPEN"
540 0360      IF ILOOP = 30 THEN STRNGO = "JGV-SUMP ON 8 + MINUTES "
541 0362      IF ILOOP = 24 THEN STRNGO = "START PB1BATCH PGM "
542 0364      IF ILOOP = 25 THEN STRNGO = "PB1BATCH WRONG UNIT "
543 0366      IF ILOOP = 26 THEN STRNGO = "TERMINATE PB1BATCH "
544 0368      IF ILOOP = 31 THEN STRNGO = "F-CA1-L/R DELTA PRESS HI "
545      ;
546 0369 AMSG: IF PTYPE = 1 THEN
547      BEGIN
548      ;
549      ;      If white alarm, print:
550      ;

```

```

551 0371      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNGO
552 0372      IF (FACK) THEN
553          BEGIN
554          ;
555          ; If white, print it is ack are reset the flag:
556          ;
557 0373      PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
558 0374      LET FACK = 0
559          END
560          END
561      ;
562      ; If CLEAR, print it is clear and reset the flag:
563      ;
564 0375      IF (FCLEAR) THEN
565          BEGIN
566 0376      PRINT ADI,"%p ALARM CLEAR",STRNGO
567 0377      LET FCLEAR = 0
568          END
569      ;
570      ; If STARTUP, then print startup message:
571      ;
572 0378      IF (FSTART) THEN
573          BEGIN
574 0379      LET FSTART = 0
575 0380      PRINT ADI,"%p",STRNGO
576          END
577 0381      RETURN
578      ;
579      ; If ERROR - print message, start timer, and set flag:
580      ;
581 0382      ERRSUB: IF ERRN != ERRNUM() THEN
582          BEGIN
583 0383      TIMERON STIMEO
584 0384      TIMERCLR STIMEO
585 0385      LET FSWO = 1
586 0386      LET ERRN = ERRNUM()
587 0387      LET ERRL = ERRLIN()
588 0388      PRINT ADI,"ERROR # %i AT LINE %i",errn,errl
589          END
590 0389      ERSCLR
591 0390      GOTO AGAIN
592      ;
593      ; If FABORT = 1, then abort program:
594      ;
595      ; TERM: LET FABORT = 0
596 0391      TERM: LET STRNGO = "PB1BATCH TERMINATED  "
597 0392      LET ILOOP = STAR + 2
598 0393      LET FSTART = 1
599 0394      GOSUB ACKK
600          END

```

```

Total Errors Detected      0      Total Warnings      0
Sequence Program Size = 7760 Bytes, 485 Pages
21 Forward Reference(s)

```

PB2BATCH


```

1      ; P B 2 B A T C H . S E Q                                VERSION 3
2      ;
3      ; Function:      Sequence program for Lamp function,
4      ;                message display on Graphic number 015,
5      ;                and printer messages.
6      ;
7      ; Written By    : R. N. Kersey
8      ; Date         : 05-31-92
9      ; Facility     : 242-A Evaporator Bldg.
10     ; Machine      : VAX 4104-A
11     ; Op. System   : VAX/VMS 5.5-2AH
12     ; D/3 Version  : 6.3-2 BASELINE
13     ; Compiler     : SIC
14     ; Build command : @([320,1]PB2BATC)
15     ;
16     ; UNIT = 8, SKDKEY = 35
17     ;
18     ; Assoc. UPF:    [300,1] PROCESS.____ PROCESS.LST
19     ; Assoc. RPF:    [300,1] 242A.____ 242A.LST
20     ; SDB:          [300,1]PCMOA.SDB PCMOASDB.CMD
21     ;
22     ; POINTN      SDB VAR.      FUNCTION
23     ; -----
24     ; PSTAT      INTn           Current EPN alarm state is an integer tied to
25     ;                graphic flashing string of value:
26     ;                (n=0-100)   0 = previously not alarmed
27     ;                .           1 = previously alarmed, not ack
28     ;                .           2 = previously alarmed, is ack
29     ;                .           3 = alarm is inhibited
30     ;                .           4 = input is off scan
31     ;                .           5 = input is bad
32     ;                LEAKn       Special INT variable for multiple white alarms on an EPN
33     ;                PDIN 'EPN'  ---
34     ;                PTYPE      TYPEn alarm type 0 = none, 1 = white, 2 = yellow,
35     ;                .           3 = yellow and yellow, 4 = red,
36     ;                .           6 = yellow and red
37     ;                PREV      FRVn Reverse acting (not fail safe) flag (1=set)
38     ;                PSKID      SKIDn Used for analog alarms
39     ;                PPASS      FPASn Turn lamp off when alarm is ACKnowledged
40     ;                PALARM      FALMn Analog alarm flag
41     ;                ---        FNING Unused error flag
42     ;
43     ; MODIFICATION LOG:
44     ;
45     ; 20-AUG-92      Set frv5=0.                                M. Teats
46     ;
47     ; 08-SEP-92      Comment-out auto-initializing of DIC-CA1-1    M. Teats
48     ;                (search ";NU") and deleted CA1 Unit in
49     ;                COMMON for synch PID.
50     ;
51     ; 30-DEC-92      Fixed white alarm print message.            M. Teats
52     ;
53     ; 07-MAY-93      Added bldg raw water alarm, FQA-RW-1.        M. Teats
54     ;
55     ; 07-JUN-93      Added comments and baselined for            M. Teats

```

```

56      :      D/3 version 3.3.  CCN# 93JUN01-01.
57      :
58      :      15-DEC-94      Revised header to indicate the new VMS      J. Berger
59      :      system. Rebase line document. Remove
60      :      obsolete Select() statement.
61      :
62      :      31-JUL-95      Revised to delete FQA-RW-1 DIN and redefine      J. Berger
63      :      it as a DEV. Added code to program to properly
64      :      let DEVs alarm. See ECN 622808 and EE-95-00006.(JFB01)
65      :
66      :
67      :
68      :
69      :
70      :
71      :
72      :
73      :
74      :
75      :
76      :
77      :
78      :
79      :
80      :
81      :
82      :
83      :
84      :
85      :
86      :
87      :
88      :
89      :
90      :
91      :
92      :
93      :
94      :
95      :
96      :
97      :
98      :
99      :
100     :
101     :
102     :
103     :
104     :
105     :
106     :
107     :
108     :
109     :
110     :

```

END OF MOD LOG

DEVS:	EPN	VAR	LOOP#	COLOR	
PB2-ALARM					
PB-2					
HV-CA1-2					
HV-CA1-2S					
HV-CA1-5					
HV-CA1-6					
JGV-STNFL					
CONTROL					
PB2SLFILK					
PB2WFILK					
PB2GLOBAL					
FQA-RW-1	24	SKID24	Y		JFB01
SPARE					
SPARE					
SPARE					
DINs:	YS-PB2-1	INT0	0	Y	
	YS-PB2-1L	INT1	1	Y	
	TSH-PB2-1	INT2	2	Y	
	PSH-CA1-3	INT3	3	R	
	PSH-PB2-1	INT4	4	R	
	SXS-PB2-1	INT5	5	Y	
	YS-VFD-1	INT6	6	Y	
	SSL-PB2-1	INT7	7	Y	
	SPARE	-	-	-	
	ZS-CA1-21	-	-	-	
	ZS-CA1-22	-	-	-	
	ZS-CA12A1	-	-	-	
	ZS-CA12A2	-	-	-	
	ZS-CA1-6N	-	-	-	
	ZS-CA1-6C	-	-	-	
	ZS-CA1-54	-	-	-	
	ZS-CA1-55	-	-	-	
	SPARE				
	SPARE				
	SPARE				
AINs:	HI-WVR-1	SKID8	8	-	
	II-PB2-1	SKID9	9	Y	

```

111      ;          SIC-PB2-1          SKID10 10   Y          SKID23 23   95H
112      ;          VI-PB2-1A         SKID11 11   Y
113      ;          PI-CA1-8           SKID12 12   Y
114      ;          FQI-PB2-W          SKID13 13   -
115      ;          FI-CA1-2           SKID14 14   RY
116      ;          PI-CA1-10          SKID15 15   -
117      ;          KY-CA1-2           SKID16 16   -
118      ;          KY-CA1-2F          SKID17 17   -
119      ;          FIC-CA1-4          SKID18 18   Y
120      ;          FQI-CA1-4          SKID19 19   -
121      ;          PI-CA1-5           SKID20 20   -
122      ;          FQI-RW-1           SKID21 21   -
123      ;          FQI-RW-1B          SKID22 22   -
124      ;          SPARE
125      ;          SPARE
126      ;          SPARE
127      ;          FI-RW2-1/2         -         -         -
128      ;          YS-BFP1/2         -         -         -
129      ;
130      ;
131      UNIT      PB2
132      ;
133      ;          COMMON WHOLE0,WHOLE1
134      ;          PB2_TMP0 == 'PB-2':DEV_TMP0
135      ;          PB2_TMP1 == 'PB-2':DEV_TMP1
136      ;          SLURRY == 'PB2SLFILK':DEV_TMP0
137      ;          WFACTOR == 'PB2WFILK':DEV_TMP0
138      ;          GBITS == 'PB2GLOBAL':DEV_TMP0
139      ;          FLAGBITS == PSPARO
140      ;          HV_TMP0 == 'HV-CA1-2':DEV_TMP0
141      ;          HV_CMD == 'HV-CA1-2':DEV_CMD
142      ;
143      0000      NOP
144      0001      LET STAR = 25          ; Start message
145      0002      LET SKDKEY = 35        ; SKID Key
146      0003      LET UNITT = 8          ; SEQ Unit number
147      0004      ASSIGN PDEV = 'PB2-ALARM' ; Alarm device
148      0005      ASSIGN PTYPE = TYPE25
149      0006      PTYPE = 0              ; No start message yet
150      0007      LET ADI = 0            ; Alarm destination index
151      0008      LET FSN0 = 0           ; No alarm yet
152      0009      TIMEROFF STIME0        ; Alarm timer off
153      0010      TIMERCLR STIME0        ; Alarm timer clear
154      0011      STRNG0 = " "           ; Blank 30 character string
155      0012      LET ERRN = 0           ; Set error number to zero
156      0013      LET ERRL = 0           ; Set error line to zero
157      0014      FTRIP4 = 0             ; FTRIP = 1 when audible horn is to be turned on
158      0015      GETUNIT TRIP           ; Which sequence unit is program running in
159      0016      IF TRIP != UNITT THEN ; Abort if running in wrong seq. unit
160      ;
161      0017      BEGIN
162      0018      FSTART = 1
163      0019      ILOOP = STAR + 1
164      0020      PTYPE = 0
165      0021      GOSUB ACKK
166      0021      GOTO TERM

```

```

166      END
167      ; IF !SELECT() THEN GOTO AGAIN
168      ;
169      ;
170      ; BEGIN INITIALIZING
171 0022 INIT: LET FABORT = 0 ; If FABORT = 1 then abort program
172 0023 LET FSTART = 0 ; Program start flag = 0
173 0024 LAMP ADI,SKOKEY,0 ; Turn off the annunciator key
174 0025 LET LSG0 = " " ; Clear the string variables
175 0026 LET LSG1 = " "
176 0027 LET LSG2 = " "
177 0028 LET LSG3 = " "
178 0029 LET LSG4 = " "
179 0030 PDEV:DEV TMO = 1 ; Set alarm device to non-alarm state
180 0031 IF AND(PDEV:DEV_ALMCT,16) THEN ; If alarm device is inhibited, remove the inhibit
181 0032 PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
182 0033 IF AND(PDEV:DEV_ALMCT,32) THEN ; If alarm device is ack, clear it
183 0034 PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
184 ;
185 ;
186 ; First, set all flags and integers:
187 ;
188 ;
189 0035 ASSIGN PDIN = 'YS-PB2-1' ; Assign pointer to first DIN
190 0036 ASSIGN PSTAT = INTO
191 0037 ASSIGN PTYPE = TYPE0
192 0038 ASSIGN PREV = FRVO
193 0039 ASSIGN PSKID = SKID0
194 0040 ASSIGN PPASS = FPAS0
195 ;
196 0041 FOR ILOOP = 0 TO 22 ;JFB01
197 0042 LET PPASS = 0
198 0043 LET PREV = 0
199 0044 IF PSTAT > 5 THEN PSTAT = 0 ; Reset current status if undefined
200 0045 LET PSKID = 0 ; Default previous status to no alarms
201 0046 LET PTYPE = 0 ; Default to no alarms
202 0047 IF ILOOP < 8 THEN ; For DINs
203 0048 BEGIN
204 0049 LET PTYPE = 0 ; Default is no alarm
205 0050 IF AND(PDIN:DIN_DFLG,3) THEN ; Check for alarms if defined (must be red or yellow)
206 0051 BEGIN
207 0052 LET PTYPE = 2 ; Default to yellow alarm
208 0053 IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1 ; Reverse acting
209 0054 IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4 ; Set to red critical alarm
210 0055 END
211 0056 END
212 0057 IF ILOOP < 7 THEN STEP PDIN BY 1 ; Point to next DIN
213 0058 STEP PPASS BY 1
214 0059 STEP PREV BY 1
215 0060 STEP PSTAT BY 1
216 0061 STEP PSKID BY 1
217 0062 STEP PTYPE BY 1
218 0063 NEXT ILOOP
219 ;
220 0064 LET TYPE9 = 2 ;JFB01

```

```

221 0065      LET TYPE11 = 2
222 0066      LET TYPE12 = 4
223 0067      LET TYPE14 = 2
224 0068      LET TYPE15 = 2
225 0069      LET TYPE18 = 2
226 0070      LET TYPE23 = 1
227 0071      LET TYPE24 = 2
228          ;
229 0072      LET FRV0 = 0
230 0073      LET FRV1 = 0
231 0074      LET FRV2 = 0
232 0075      LET FRV3 = 0
233 0076      LET FRV4 = 0
234 0077      LET FRV5 = 0
235 0078      LET FRV24 = 1
236 0079      LET FPAS24 = 0
237          ;
238 0080      LET A10 = 95.0
239          ;
240 0081      FLG14 = 0          ; PID auto (1) flag, for synch PID
241 0082      FLG15 = 0          ; PID manual (1) flag, for synch PID
242 0083      A120 = 0.0        ; For PID O/C display bar
243          ;
244          END INITIALIZING
245          ;
246          SEND PROGRAM START MESSAGE
247          ;
248 0084      ANALOG = 0
249 0085      ILOOP = STAR
250 0086      FSTART = 1
251 0087      ASSIGN PTYPE = TYPE34
252 0088      GOSUB ACKK
253 0089      ON ERROR GOSUB ERRSUB
254          ;
255          ;
256          ;
257 0090      AGAIN: MARK STEP "WTUNTL",0,0
258          ; WAITUNTIL SELECT() | FABORT          ; Wait until micro is selected or program aborted
259          ; IF (FABORT) THEN GOTO TERM          ; Terminate program if abort flag set
260 0091      MARK STEP "RNNING",0,0
261 0092      WAIT 1
262 0093      IF (FSN0) THEN          ; If alarm flag is set
263 0094      IF STIME0 > 60 THEN          ; If alarm flag active for 1 minute
264          BEGIN          ; Reset timer and turn off flag
265          TIMEROFF STIME0
266          TIMERCLR STIME0
267 0097      LET FSN0 = 0
268 0098      LET ERRN = 0
269 0099      LET ERRL = 0
270          END
271          ;
272          ; If PB2-ALARM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
273          ;
274 0100      IF (PDEV:DEV_TMO0 = 2) & AND(PDEV:DEV_ALMCT,32) THEN
275          BEGIN

```

```

276 0101      PDEV:DEV.TMPO = 1          ; Put PB2-ALARM in ack state
277 0102      LET FTRIP4 = 1            ; Set for unit alarm ack, system to graphic
278 0103      LET ACK = 1              ; Set for unit alarm ack, graphic to system
279 0104      CRIT
280 0105      LAMP ADI,SKDKEY,1        ; Turn lamp on
281      END
282      ;
283      ;
284      ;
285 0106      ASSIGN PDIN = 'YS-PB2-1'
286 0107      ASSIGN PTYPE = TYPE0
287 0108      ASSIGN PREV = FRV0
288 0109      ASSIGN PSTAT = INTO
289 0110      ASSIGN PPASS = FPAS0
290 0111      LET CURENT = 0            ; Used to set lamp status
291 0112      LET LAST = 0
292 0113      LET ANALOG = 0
293 0114      LET NXT = 0
294 0115      LMPVAR = 0
295 0116      BITVAL = 0
296      ;
297 0117      LOP: FOR ILOOP = 0 TO 7
298      ;
299 0118      IF !SELECT() THEN GOTO AGAIN
300      IF ( PTYPE > 0 ) THEN
301      BEGIN
302      IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3 ;be1 ; If some type of alarm
303      IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5 ; If EPN inhibited
304      IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4 ; If EPN is bad input
305      IF ( PSTAT < 3 ) THEN ; If EPN is disabled
306      BEGIN ; If not inhibited
307      IF (PSTAT = 1) THEN ;be2
308      BEGIN ;be3
309      IF (PTYPE != 1) THEN
310      IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
311      IF (FTRIP4) THEN GOSUB ACK1
312      END ;en3
313      IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
314      IF (PSTAT) THEN ;be4
315      BEGIN
316      IF (PDIN:DIN_VAL != PREV) THEN GOSUB CLR
317      NXT = PSTAT
318      IF (PPASS) & (PSTAT = 2) THEN NXT = 0
319      LAST = OR(LAST,NXT)
320      END ;en4
321      ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0 ;en2
322      END ;en1
323      STEP PDIN BY 1
324      STEP PTYPE BY 1
325      STEP PREV BY 1
326      STEP PSTAT BY 1
327      STEP PPASS BY 1
328      NEXT ILOOP
329      ;
330      ;
331      END DINs

```

```

331      ;
332      ;
333      ;
334      0150      ASSIGN PAI = 'HI-WVR-1'
335      0151      ASSIGN PSKID = SKID8
336      0152      FOR JLOOP = 8 TO 22
337      ;
338      0153      IF ISELECT() THEN GOTO AGAIN
339      ;
340      0154      IF (PTYPE) THEN
341      ;
342      0155      BEGIN
343      0156      BITVAL = PAI:AI ALMCT
344      0157      IF !AND(BITVAL,16) THEN
345      ;
346      0158      BEGIN
347      0159      IF AND(BITVAL,15) THEN
348      ;
349      0160      BEGIN
350      0161      PSKID = BITVAL
351      0162      IF BITVAL < 3 THEN FALMON = 1
352      0163      LET PDEV:DEV_TMP0 = 2
353      0164      NXT = 1
354      ;
355      0165      END
356      ;
357      0166      ELSE NXT = 1
358      ;
359      0167      END
360      ;
361      0168      ELSE
362      ;
363      0169      BEGIN
364      0170      NXT = 0
365      0171      LET PSKID = 0
366      0172      END
367      ;
368      0173      END
369      ;
370      0174      ELSE
371      ;
372      0175      BEGIN
373      0176      NXT = 0
374      0177      LET PSKID = 0
375      0178      END
376      ;
377      0179      END
378      ;
379      ;
380      ;
381      ;
382      ;
383      0177      RC1:  ASSIGN PTYPE = TYPE24
384      0178      ASSIGN PREV = FRV24
385      0179      ASSIGN PSTAT = INT24

```

; If some alarm exists

; If not inhibited

; If alarm exists and is current

; If alarm is ack

; If alarm is unack

;be1

;be2

;be3

;be4

;be5

;en5

;en4

;en3

; Else there is no alarm

;en2

; Else alarm is inhibited

;en1

JFB01

```

386 0180      ASSIGN PPASS = FPAS24
387 0181      IF ( PTYPE > 0 ) THEN
388          BEGIN
389 0183          IF AND('FQA-RW-1':DEV_ALMCT,16) THEN LET PSTAT = 3      ; EPN is inha
390          IF AND('FQA-RW-1':DEV_ALMCT,4) THEN BLET PSTAT = 5          ; No IBAD on devices
391 0185          IF !AND('FQA-RW-1':DEV_FLG,8) THEN LET PSTAT = 4      ; EPN is dis
392 0186          IF ( PSTAT < 3 ) THEN
393              BEGIN
394 0187              IF PSTAT = 1 THEN
395                  BEGIN
396 0188                  IF PTYPE != 1 THEN
397 0190                      IF AND('FQA-RW-1':DEV_ALMCT,32) THEN GOSUB ACK1
398 0192                      IF (FTRIP4) THEN GOSUB ACK1
399                      END
400 0194                      IF PSTAT = 0 THEN IF AND('FQA-RW-1':DEV_STAT, 1) THEN
401                          BEGIN
402 0195                          CRIT
403 0196                          LET PSTAT = 1
404 0197                          'PB2-ALARM':DEV_TMPO = 2
405 0198                          FALMON = 1
406 0199                          LAMP ADI,SKDKEY,2
407 0200                          GOSUB ACKK
408 0201                          NOCRIT
409                          END
410 0202                          IF (PSTAT) THEN
411                              BEGIN
412 0203                              IF !AND('FQA-RW-1':DEV_STAT, 1) THEN
413                                  BEGIN
414 0204                                  LET PSTAT = 0
415 0205                                  LET FCLEAR = 1
416 0206                                  GOSUB ACKK
417                                  END
418 0207                                  NXT = PSTAT
419 0209                                  IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
420 0210                                  LAST = OR(LAST,NXT)
421                                  END
422                                  END
423 0212                      ELSE IF !AND('FQA-RW-1':DEV_ALMCT,16)&AND('FQA-RW-1':DEV_FLG,8)THEN LET PSTAT = 0
424 0214                      IF PSTAT = 5 THEN PSTAT = 0
425                      END
426              ;
427              ;
428              ;
429              ;
430              ;
431 0215      WHT:      WHITE = 0
432 0216      ASSIGN PAI = 'SIC-PB2-1'
433 0217      ASSIGN PALARM = AIO
434 0218      ASSIGN PREV = FRV23
435 0219      ASSIGN PSTAT = INT23
436 0220      ASSIGN PTYPE = TYPE23
437 0221      ASSIGN PPASS = FPAS23
438 0222      ILOOP = 23
439 0223      GOSUB ANLG
440      ;

```



```

441      ;      END ANALOG WHITE ALARMS:
442      ;
443      0224      MARK STEP "BELOW",0,0
444      0226      IF LAST > 3 THEN PRINT AD1," LAMP VARIABLE IS GREATER THAN 3!"
445      0227      CURENT = OR(LAST,WHITE)
446      0228      CURENT = OR(CURENT,ANALOG)
447      0230      IF AND(CURENT,1)THEN LMPVAR = 2
448      0232      ELSE IF AND(CURENT,2)THEN LMPVAR = 1
449      0233      ELSE LMPVAR = 0
450      0234      LAMP AD1,SKDKEY,LMPVAR
451      0235      MARK STEP "BOTTOM",0,0
452      0236      NOCRIT
453      0237      LET FTRIP4 = 0
454
455      ;
456      ;      BEGIN LOGIC FOR GRAPHICS:
457      ;
458      0238      at21 = and('PSH-CA1-3':DIN_VAL,'TSH-PB2-1':DIN_VAL)
459      0239      at22 = and('YS-PB2-1':DIN_VAL,'PSH-PB2-1':DIN_VAL)
460      0240      at23 = and(pb2_tmp1,128)
461      0241      at24 = and(pb2_tmp1,256)
462      0242      at25 = and(pb2_tmp1,512)
463      0243      at26 = and(at21,at22)
464      0244      at27 = and(at23,at24)
465      0245      at28 = and(at25,at26)
466      0246      at29 = and(at27,at28)
467      ;
468      0247      AT31 = AND(PB2_TMP1,32768)
469      0248      AT0 = AND(WFACTOR,1)
470      0249      AT1 = AND(WFACTOR,2)
471      0250      AT2 = AND(WFACTOR,4)
472      0251      AT3 = AND(WFACTOR,8)
473      0252      GT3 = AND(GBITS,8)
474      0253      GT4 = AND(GBITS,16)
475      0254      GT9 = AND(GBITS,512)
476      0255      GT11 = AND(GBITS,2048)
477      0256      BT0 = AND('LDS-SL502':DIN_VAL,'LDS-SL503':DIN_VAL)
478      0257      BT1 = AND('LDS-SL504':DIN_VAL,'LDS-SL505':DIN_VAL)
479      0258      BT2 = AND('LDS-DR700':DIN_VAL,'LDS-SL500':DIN_VAL)
480      0259      BT3 = AND(BT2,'LDS-SL501':DIN_VAL)
481      0260      BT4 = AND(BT0,BT1)
482      0261      AT9 = AND(BT3,BT4)
483      0262      BT5 = AND('YS-AW-MSD':DIN_VAL,'LDS-COBAN':DIN_VAL)
484      0263      BT6 = AND('LDS-ANSL1':DIN_VAL,'LDS-DR368':DIN_VAL)
485      0264      BT7 = AND('LDS-SP-AW':DIN_VAL,'LDS-AP-A':DIN_VAL)
486      0265      BT8 = AND('YS-AP-MSD':DIN_VAL,'LDS-AP-VP':DIN_VAL)
487      0266      BT9 = AND(BT5,BT6)
488      0267      BT10 = AND(BT7,BT8)
489      0268      AT10 = AND(BT9,BT10)
490      0269      BT11 = AND('PSH-CA1-3':DIN_VAL,'TSH-PB2-1':DIN_VAL)
491      0270      BT12 = AND('PSH-PB2-1':DIN_VAL,'PSH-SL113':DIN_VAL)
492      0271      BT13 = AND('PSH-SL114':DIN_VAL,'PSH-SL101':DIN_VAL)
493      0272      BT14 = AND(BT13,'PSH-SL100':DIN_VAL)
494      0273      BT15 = AND(BT14,BT12)
495      0274      AT11 = AND(BT14,BT11)
496      0275      BT16 = AND('LDS-AP-SL':DIN_VAL,'YS-AW-MSD':DIN_VAL)

```

```

496 0276      BT17= AND('LDS-AW-SL':DIN_VAL,'RSH-SP-AW':DIN_VAL)
497 0277      BT18= AND('LDS-AW-DR':DIN_VAL,'LDS-COBAAX':DIN_VAL)
498 0278      BT19= OR('RSH-AXZY1':DIN_VAL,'YS-AAXARM':DIN_VAL)
499 0279      BT20= AND(BT19,'LDSPPCOB1':DIN_VAL)
500 0280      BT21= AND(BT16,BT17)
501 0281      BT22= AND(BT18,BT20)
502 0282      AT12= AND(BT21,BT22)
503 0283      BT23= AND(AT9,AT10)
504 0284      BT24= AND(AT11,AT12)
505 0285      AT13= AND(BT23,BT24)
506 0286      BT25= AND(PB2,TMP0,1)
507 0287      BT26= AND(BT25,AT31)
508 0288      SA1 = AND(AT20,BT26)
509 0289      SA3 = AND(SA1,AT13)
510 0290      SA2 = AND('PB-2':DEV_STAT,4)      ; PB-2 RESET
511 0291      SA4 = AND('PB-2':DEV_STAT,16)     ; PB-2 SHUTDOWN
512 0292      SB4 = AND(PB2,TMP0,2)             ; PB-2 TEMPORARY BIT $T1
513 0293      SB5 = 'YS-PB2-1':DIN_VAL
514 0294      SB6 = AND('PB-2':DEV_STAT,32)   ; PB-2 INTERLOCK
515 0295      CT0 = AND(1,HV_TMP0)              ; FIC-CA1-4 LOW FLOW $T0 HV-CA1-2
516 0296      CT2 = AND(4,HV_CMD)              ; EVAPORATOR FLUSH, $C2 HV-CA1-2
517 0297      CT3 = AND(8,HV_TMP0)             ; EVAP FLUSH TIMEOUT, $T3 HV-CA1-2
518 0298      CT4 = AND(4,HV_TMP0)             ; EVAP BTIMER ON $T2 HV-CA1-2
519 0299      CT5 = AND(2048,HV_TMP0)          ; FARM BTIMER ON $T11 HV-CA1-2
520 0300      CT6 = 'ZS-CA1-21':DIN_VAL        ; HV-CA1-2 POSN-1
521 0301      CT7 = 'ZS-CA1-22':DIN_VAL        ; HV-CA1-2 POSN-2
522 0302      CT8 = 'ZS-CA12A1':DIN_VAL        ; HV-CA1-2A POSN-1
523 0303      CT9 = 'ZS-CA12A2':DIN_VAL        ; HV-CA1-2A POSN-2
524 0304      CT10= XOR(CT5,CT12)
525 0305      CT11= AND(8,HV_CMD)              ; FARM FLUSH $C3 HV-CA1-2
526 0306      CT12= AND(4096,HV_TMP0)         ; FARM FLUSH TIMEOUT $T12 HV-CA1-2
527 0307      CT13= AND(CT4,CT3)
528 0309      IF AND('PB2-ALARM':DEV_FLG,64) THEN PUTMANL 'PB2-ALARM'
529 0311      IF AND('PB-2':DEV_FLG,64) THEN PUTMANL 'PB-2'
530 0313      IF AND('HV-CA1-5':DEV_FLG,64) THEN PUTMANL 'HV-CA1-5'
531 0315      IF AND('HV-CA1-6':DEV_FLG,64) THEN PUTMANL 'HV-CA1-6'
532 0317      IF AND('JGV-STMF1':DEV_FLG,64) THEN PUTMANL 'JGV-STMF1'
533 0319      IF AND('CONTROL':DEV_FLG,64) THEN PUTMANL 'CONTROL'
534 0321      IF AND('PB2SLFILK':DEV_FLG,64) THEN PUTMANL 'PB2SLFILK'
535 0323      IF AND('PB2WFINLK':DEV_FLG,64) THEN PUTMANL 'PB2WFINLK'
536
537      ;
538      ;
539      ;
540      ;
541      ;
542      ;
543      ;
544      ;
545      ;
546      ;
547 0324      IF AND('SIC-PB2-1':CB_FLG,1,64) THEN      ; If SIC in auto
548      BEGIN                                           ;be1
549 0325      IF (FLG16=0) & (FLG15=1) THEN              ; If SIC previously manual
550      BEGIN                                           ;be2

```

```

551 0326          CLOSCASC 'SIC-PB2-1'
552 0328          IF IAND('FIC-CA1-4':CB_FLG:4,64) THEN PUTAUTO 'FIC-CA1-4'
553 0329          LET FLG14 = 1
554 0330          LET FLG15 = 0
555          END
556          ;en2
557          ELSE
558          ; If SIC in manual
559          BEGIN
560          ;be3
561          IF ((FLG14=1)&(FLG15=0)) ; ((FLG14=0)&(FLG15=0)) THEN
562          ; If prev auto or init
563          BEGIN
564          LET FLG14 = 0
565          LET FLG15 = 1
566          END
567          ;en4
568          ;en3
569          END SYNCH PID CASCADED SETPT
570          BEGIN O/C DISPLAY BAR:
571          Function: Upon loss of permissive, put the control loop in manual
572          and set the PID block output to zero.
573 0335          IF ('SIC-PB2-1':CB_LOUT:3 >= 2) THEN DLEAK3 = 1
574 0336          IF ('SIC-PB2-1':CB_LOUT:3 = 0) & AND(DLEAK3,1) THEN
575          ; Permissive must be Hi (1)
576          ; Permissive lost and 1st pass
577          BEGIN
578          PUTMANL 'SIC-PB2-1'
579          PRINT ADI," PERMISSIVE LOST >>> SIC-PB2-1 PUT IN MANUAL"
580          PUTOUT 'SIC-PB2-1',AI20
581          PRINT ADI," PERMISSIVE LOST >>> SIC-PB2-1 OUTPUT SET TO ZERO"
582          DLEAK3 = 0
583          END
584          END O/C DISPLAY BAR
585 0342          GOTO AGAIN
586          END PROGRAM LOOP
587          BEGIN ANALOG WHITE SUBROUTINE:
588          ;
589          ;
590          ;
591          ; ANLG: IF ISELECT() THEN GOTO AGAIN
592          0343          ANLG: IF (PTYPE = 1) THEN
593          ; If some alarm
594          BEGIN
595          BITVAL = PAI:AI_ALMCT
596          IF I(AND(BITVAL,16)) THEN
597          ; If not inhibited
598          BEGIN
599          IF PSTAT = 3 THEN LET PSTAT = 0
600          ; If rev acting, LOLO, or LO
601          BEGIN
602          IF (PSTAT = 0) THEN
603          ; If not previously alarmed
604          BEGIN
605          IF PAI:AI_MEAS <= PALARM THEN GOSUB ALM
606          END
607          ELSE
608          BEGIN
609          ;be5

```

```

606 0352          IF (PSTAT = 1) THEN                ; If previously alarmed
607              BEGIN                                ;be6
608          0354          IF (FTRIP4) THEN GOSUB ACK1 ; If ACK flag set, then ack it
609          0357          IF PTYPE != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
610              END                                  ;en6
611          0359          IF (PAI:AI_MEAS >= PALARM + PAI:AI_DEAD) THEN GOSUB CLR
612              END                                  ;en5
613              END                                  ;en3
614          ELSE
615              BEGIN                                ;be10
616          0360          IF (PSTAT = 0) THEN
617              BEGIN                                ;be11
618          0362          IF (PAI:AI_MEAS >= PALARM) THEN GOSUB ALM
619              END                                  ;en11
620          ELSE
621              BEGIN                                ;be12
622          0363          IF (PSTAT = 1) THEN
623              BEGIN                                ;be13
624          0365          IF (FTRIP4) THEN GOSUB ACK1
625          0368          IF PTYPE != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
626              END                                  ;en13
627          0370          IF (PAI:AI_MEAS <= PALARM - PAI:AI_DEAD) THEN GOSUB CLR
628              END                                  ;en12
629              END                                  ;en10
630          0371          NXT = PSTAT
631          0373          IF (PPASS) & (PSTAT = 2) THEN NXT = 0
632              END                                  ;en2
633          ELSE
634              BEGIN
635          0374          LET PSTAT = 3
636          0375          NXT = 0
637              END
638          0376          WHITE = OR(WHITE,NXT)
639              END                                  ;en1
640          0377          ELSE PSTAT = 0
641          0378          STEP PALARM BY 1
642          0379          STEP PSTAT BY 1
643          0380          STEP PTYPE BY 1
644          0381          STEP PREV BY 1
645          0382          STEP PPASS BY 1
646          0383          ILOOP = ILOOP + 1
647          0384          RETURN
648          ;
649          ;
650          ;
651          ;
652          ;
653          0385          CLR: PSTAT = 0                ; Int set to zero
654              ;
655              ; If not white, then unack:
656              ;
657          0388          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
658          0389          LET FCLEAR = 1                ; Clear flag
659          0390          GOSUB ACKK                    ; Print message
660          0391          RETURN

```

```

661      ;
662 0392  ALM:    LET PSTAT = 1                                ; Int set to one
663      ;
664      ; If not white, then unack:
665      ;
666 0395      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
667      ;
668      ; Turn audible alarm on if white or yellow:
669      ;
670 0396      CRIT
671 0398      IF AND(PTYPE,3) THEN FALMON = 1
672 0399      LET PDEV:DEV_TMPD = 2
673 0400      LAMP ADI,SKDKEY,2
674 0401      GOSUB ACKK                                         ; Print message
675 0402      RETURN
676      ;
677 0403  ACK1:   LET PSTAT = 2                                ; Ack the alarm
678 0404      LET FACK = 1                                       ; Set ACK flag for C program
679 0405      LAMP ADI,SKDKEY,1                                   ; Lamp on
680      ;
681      ;
682      ; ALARM TYPER STRINGS:
683      ;
684 0407  ACKK:   IF ILOOP = 1 THEN STRNGO = "PUMP PB-2 HARDWARE INTERLOCK"
685      IF ILOOP = 2 THEN STRNGO = "PUMP PB-2 TEMPERATURE HIGH "
686 0411      IF ILOOP = 3 THEN STRNGO = "PUMP PB-2 DISCHARGE PRESS HI"
687 0413      IF ILOOP = 4 THEN STRNGO = "PUMP PB-2 RUPTURE DISK BROKN"
688 0415      IF ILOOP = 5 THEN STRNGO = "PUMP PB-2 ENCODER LOSS "
689 0417      IF ILOOP = 6 THEN STRNGO = "PB-2 VARIABLE FREQ DRIVE LOSS"
690 0419      IF ILOOP = 7 THEN STRNGO = "PUMP PB-2 ZERO SPEED "
691 0421      IF ILOOP = 24 THEN STRNGO = "BUILDING FLUSH WATER ON "
692 0423      IF ILOOP = 23 THEN STRNGO = "PUMP PB-2 CURRENT HIGH "
693 0425      IF ILOOP = 25 THEN STRNGO = "START PB2BATCH PGM "
694 0427      IF ILOOP = 26 THEN STRNGO = "PB2BATCH WRONG UNIT"
695 0429      IF ILOOP = 27 THEN STRNGO = "TERMINATE PB2BATCH "
696      ;
697      ; If acknowledging alarm, then:
698      ;
699 0430  AMSG:   IF PTYPE = 1 THEN
700      BEGIN
701      ;
702      ; If white alarm, print:
703      ;
704 0432      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNGO
705 0433      IF (FACK) THEN
706      BEGIN
707      ;
708      ; If white, print it is ack and reset the flag:
709      ;
710 0434      PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
711 0435      LET FACK = 0
712      END
713      END
714      ;
715      ; If CLEAR, print it is clear and reset flag:

```

```

716 0436      IF (FCLEAR) THEN
717          BEGIN
718 0437      PRINT ADI,"%p ALARM CLEAR",STRNGO
719 0438      LET FCLEAR = 0
720          END
721      ;
722      ; If STARTUP, print startup message:
723      ;
724 0439      IF (FSTART) THEN
725          BEGIN
726 0440      LET FSTART = 0
727 0441      PRINT ADI,"%p",STRNGO
728          END
729 0442      RETURN
730      ;
731      ; If ERROR - print message, start timer, and set flag:
732      ;
733 0443  ERRSUB: IF ERN != ERRNUM() THEN
734          BEGIN
735 0444      TIMERCLR STIME0
736 0445      LET FSNO = 1
737 0446      LET ERN = ERRNUM()
738 0447      LET ERRL = ERRLIN()
739 0448      PRINT ADI,"ERROR # %i AT LINE %i",ern,erri
740          END
741 0449      ERSCLR
742 0450      GOTO AGAIN
743      ;
744      ; If FABORT = 1, then abort program:
745      ;
746      ;
747      ; TERM: LET FABORT = 0
748 0451  TERM: LET STRNGO = "PB2BATCH TERMINATED  "
749 0452      LET ILOOP = STAR + 2
750 0453      LET FSTART = 1
751 0454      GOSUB ACKK
752          END

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 9136 Bytes, 571 Pages
23 Forward Reference(s)

EA1BATCH

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

12:38:26 24-Jan-96

Page 1

```

1      ; E A 1 B A T C H . S E Q                                VERSION 4
2      ;
3      ; Function:      Sequence program for Lamp function,
4      ;                message display on Graphic number 013,
5      ;                and printer messages.
6      ;
7      ; Written By    : R. M. Kersey
8      ; Date         : 05-31-92
9      ; Facility     : 242-A Evaporator Bldg.
10     ; Machine      : VAX 4104-A
11     ; Op. System   : VAX/VMS 5.5-2AH
12     ; D/3 Version  : 6.3-3 BASELINE
13     ; Compiler     : SIC
14     ; Build command : @C320,201EA1BATCH
15     ;
16     ; UNIT = 9, SKDKEY = 33
17     ;
18     ; Assoc. UPF:    [300,1] _PROCESS_      PROCESS.LST
19     ; Assoc. RPF:    [300,1] _242A_        242A.LST
20     ; SDB:          [300,1] PCMOA.SDB      PCMOASDB.CMD
21     ;
22     ; POINTER      SDB VAR.      FUNCTION
23     ; -----
24     ; PSTAT      INTn      Current EPN alarm state is an integer tied to
25     ;                graphic flashing string of value:
26     ;                0 = previously not alarmed
27     ;                1 = previously alarmed, not ack
28     ;                2 = previously alarmed, is ack
29     ;                3 = alarm is inhibited
30     ;                4 = input is off scan
31     ;                5 = input is bad
32     ;
33     ; POIN      LEAKn      Special INT variable for multiple white alarms on an EPN
34     ;           'EPN'      ---
35     ; PTYPE      TYPEn      alarm type 0 = none, 1 = white, 2 = yellow,
36     ;                3 = yellow and yellow, 4 = red,
37     ;                6 = yellow and red
38     ; PREV      FRVn      Reverse acting (not fail safe) flag (1=set)
39     ; PSKID      SKIDn      Used for analog alarms
40     ; PPASS      FPASn      Turn lamp off when alarm is ACKnowledged
41     ; PALARM      FALMn      Analog alarm flag
42     ;           FNING      Unused error flag
43     ;
44     ; MODIFICATION LOG:
45     ;
46     ; 30-DEC-92      Fixed white alarm print message.                M. Teats
47     ;
48     ; 07-MAY-93      Delete KY-EA1-1, move CI-EC1-U1 to              M. Teats
49     ;                Unit EC1.
50     ;
51     ; 07-JUN-93      Added comments and baseline for                M. Teats
52     ;                D/3 version 3.3. CCN# 93JUN01-01.
53     ;
54     ; 15-DEC-94      Revised header to indicate the new VMS        J. Berger
55     ;                system. Rebase line document. Removed
56     ;                obsolete Select() statements.

```


MHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

12:38:26 24-Jan-96

Page 2

```

56      ;
57      ;      21-APR-95      Revised white alarm routine to set the      J. Berger
58      ;
59      ;      EE-95-00006 and EDN 622495. (JFB01)
60      ;
61      ;      04-OCT-95      Revised to remove PSL-EA1-1 form program J. Berger
62      ;      See EDN 623939 and EE-95-00006. JFB02
63      ;
64      ;      17-JAN-96      Added EPNs PIC-EA116 & ZI-EA1-16 to system      J. Berger
65      ;      per EDNs 620355 & 627421 and per package
66      ;      EE-94-00769. JFB03.
67      ;
68      ;      END MOD LOG
69      ;
70      ;
71      ;      EPN      TYPE      IPN      LOOP      WHITE
72      ;      EA1-ALARM      DEV      65      -      -
73      ;      FV-EA1-1      DEV      66      -      -
74      ;      EA1-INLK1      DEV      67      -      -
75      ;      EA1-INLK2      DEV      68      -      -
76      ;      TV-DSH1/2      DEV      69      -      -
77      ;      SELECT-TI      DEV      70      -      -
78      ;      SPARES      DEV      71-79      -      -
79      ;
80      ;      PSL-EA1-1      DI      105      0      -
81      ;      TSH-DSH-3      DI      106      1      -
82      ;      ZS-EA1-1C      DI      107      2      -
83      ;      SPARE      DI      108      -      -
84      ;      SPARE      DI      109      -      -
85      ;
86      ;      TI-EA1-7      AI      120      3      -
87      ;      TI-EA1-7S      AI      121      4      -
88      ;      TI-EA-1      AI      122      5      -
89      ;      TI-EA1-1S      AI      123      6      -
90      ;      TDI-EA1-1      AI      124      7      -
91      ;      PI-EA1-12      AI      125      8      -
92      ;      PI-EA1-2      AI      126      9      -
93      ;      PD1-EA1-1      AI      127      10     -
94      ;      PI-STM-1      AI      128      11     -
95      ;      PI-EA1-13      AI      129      12     24,25
96      ;      PI-EA1-14      AI      130      13     -
97      ;      FCI-EA1-1      AI      131      14     26
98      ;      ZI-EA1-1      AI      132      15     -
99      ;      TI-DSH-2      AI      133      16     -
100     ;      PI-EA1-1      AI      134      17     27,28
101     ;      PD-EA1-11      AI      135      18     -
102     ;      PD1-EA1-2      AI      136      19     -
103     ;      TI-DSH-3      AI      137      20     -
104     ;      TI-EA1-5      AI      138      21     29
105     ;      CI-EA1-U2      AI      139      22     -
106     ;      CI-EA1-U1      AI      140      23     -
107     ;      PIC-EA116      AI      141      -      -
108     ;      ZI-EA1-16      AI      142      -      -
109     ;      SPARES      AI
110     ;

```

```

111      ;
112      UNIT    EA1
113      ;
114      COMMON WHOLE0
115      FLAGBITS == PSPARO
116      FVEA1 == 'FV-EA1-1':DEV_TMPO
117      INLK1 == 'EA1-INLK1':DEV_TMPO
118      INLK2 == 'EA1-INLK2':DEV_TMPO
119      G bits == PB2GLOBAL:DEV_TMPO
120      STAT_BITS == 'FV-EA1-1':DEV_STAT
121      ;
122      0000      NOP
123      0001      LET STAR = 34                      ; Start message
124      0002      LET SKDKEY = 33                    ; SKID Key
125      0003      LET UNIT1 = 9                      ; SEQ Unit number
126      0004      ASSIGN PDEV = 'EA1-ALARM'          ; Alarm device
127      0005      ASSIGN PTYPE = TYPE34
128      0006      LET AD1 = 0                        ; Alarm destination index
129      0007      LET FSN0 = 0                      ; No alarms yet
130      0008      TIMEROFF STIME0                   ; Alarm timer off
131      0009      TIMERCLR STIME0                   ; Alarm timer clear
132      0010      LET STRNG0 = " "                  ; Blank 30 character string
133      0011      LET ERRN = 0                      ; Set error number to zero
134      0012      LET ERRL = 0                      ; Set error line to zero
135      0013      GETUNIT TRIP                       ; Which sequence unit is program running in
136      0014      IF TRIP != UNIT1 THEN              ; Abort if running in wrong seq. unit
137          BEGIN
138      0015          FSTART = 1
139      0016          ILOOP = STAR + 1
140      0017          PTYPE = 0
141      0018          GOSUB ACKK
142      0019          GOTO TERM
143      0020      END
144      ; IF ISELECT() THEN GOTO AGAIN
145      ;
146      ; BEGIN INITIALIZING
147      ;
148      0020      INIT: LET FABORT = 0                ; If FABORT = 1 then abort program
149      0021      LET FSTART = 0                      ; Program start flag = 0
150      0022      LAMP AD1,SKDKEY,0                  ; Turn off annunciator key
151      0023      LET LSG0 = " "                    ; Clear the string variables
152      0024      LET LSG1 = " "
153      0025      LET LSG2 = " "
154      0026      LET LSG3 = " "
155      0027      LET LSG4 = " "
156      0028      PDEV:DEV_TMPO = 1                  ; Set alarm device to non-alarm state
157      0029      IF AND(PDEV:DEV_ALMCT,16) THEN      ; If alarm device is inhibited, remove the inhibit
158      0030          PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
159      0031      IF AND(PDEV:DEV_ALMCT,32) THEN      ; If alarm device is ack, clear it
160      0032          PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
161      ;
162      ; First, set all flags and integers:
163      ;
164      0033      ASSIGN PDIN = 'TSH-DSH-3'           ; Assign pointer to first DIN
165      0034      ASSIGN PSTAT = INT1                 ; JF802

```

```

166 0035      ASSIGN PTYPE = TYPE1
167 0036      ASSIGN PREV = FRV1
168 0037      ASSIGN PSKID = SKID1
169 0038      ASSIGN PPASS = FPAS1
170
171 0039      ;
172 0040      FOR ILOOP = 1 TO 29                      ; JFB02
173 0041      LET PPASS = 0
174 0043      IF PSTAT > 5 THEN PSTAT = 0              ; Reset current status if undefined
175 0044      LET PSKID = 0                            ; Default previous status to no alarms
176 0045      LET PTYPE = 0                            ; Default to no alarms
177 0046      IF ILOOP < 3 THEN                      ; For DINs
178          BEGIN
179 0047          LET PTYPE = 0                        ; Default to no alarm
180 0048          IF AND(PDIN:DIN_OFLG,3) THEN        ; Check for alarms if defined (must be red or yellow)
181              BEGIN
182 0049              LET PTYPE = 2                    ; Default to yellow alarm
183 0051              IF AND(PDIN:DIN_OFLG,1) THEN LET PREV = 1 ; Reverse acting
184 0053              IF AND(PDIN:DIN_OFLG,4) THEN LET PTYPE = 4 ; Set to red critical alarm
185              END
186          END
187 0054      ELSE LET PTYPE = 0                      ; Set to no alarm
188 0056      IF ILOOP < 2 THEN STEP PDIN BY 1        ; Point to next DIN
189 0058      IF ILOOP > 23 THEN LET PTYPE = 1        ; Set to white alarm
190 0059      STEP PPASS BY 1
191 0060      STEP PREV BY 1
192 0061      STEP PSTAT BY 1
193 0062      STEP PSKID BY 1
194 0063      STEP PTYPE BY 1
195 0064      NEXT ILOOP
196
197 0065      ;
198 0066      LET TYPE0 = 1
199 0067      LET TYPE3 = 4
200 0068      LET TYPE4 = 4
201 0069      LET TYPE5 = 4
202 0070      LET TYPE6 = 4
203 0071      LET TYPE7 = 4
204 0072      LET TYPE11 = 2
205 0072      LET TYPE20 = 2
206
207 0073      ;
208 0074      LET FRV0 = 0
209 0075      LET FRV1 = 0
210 0076      LET FRV2 = 1
211 0077      LET FRV24 = 1
212 0078      LET FRV25 = 1
213 0079      LET FRV26 = 1
214 0080      LET FRV27 = 1
215
216 0081      ;
217 0082      LET A10 = 75.0
218 0083      LET A11 = 80.0
219 0084      LET A12 = 3000.0
220 0085      LET A13 = 17.7
221 0086      LET A14 = 29.7
222 0087      LET A15 = 190.0

```

```

221 0086      LET A119 = 0.0                ; For FIC-EA1-1 output (O/C Display Bar)
222      ;
223      ; END INITIALIZING
224      ;
225      ; SEND PROGRAM START MESSAGE
226      ;
227 0087      ANALOG = 0
228 0088      ILOOP = STAR
229 0089      FSTART = 1
230 0090      PTYPE = 0
231 0091      GOSUB ACK
232 0092      ON ERROR GOSUB ERRSUB
233      ;
234      ; BEGIN PROGRAM LOOP
235      ;
236 0093  AGAIN: MARK STEP "WTUNT1",0,0
237      ; WAITUNTIL SELECT() ; FABORT      ; Wait until micro is selected or program aborted
238      ; IF (FABORT) THEN GOTO TERM      ; Terminate program if abort flag set
239 0094      MARK STEP "RNINNG",0,0
240 0095      WAIT 1
241 0096      IF (FSNO) THEN                ; If alarm flag is set
242 0097      IF (FSNO) > 60 THEN            ; If alarm flag active for 1 minute
243      BEGIN                              ; Reset timer and turn off flag
244      TIMEROFF STIME0
245 0099      TIMERCLR STIME0
246 0100      LET FSNO = 0
247 0101      LET ERRN = 0
248 0102      LET ERRL = 0
249      END
250      ;
251      ; If EA1-ALARM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
252      ;
253 0103      IF (PDEV:DEV_TMPO = 2) & AND(PDEV:DEV_ALMCT,32) THEN
254      BEGIN
255 0104      LET PDEV:DEV_TMPO = 1 ; Put EA1-ALARM in ack state
256 0105      LET FTRIP4 = 1          ; Set for unit alarm ack, system to graphic
257 0106      LET ACK = 1            ; Set for unit alarm ack, graphic to system
258 0107      LET FACK = 1
259      CRIT
260 0108      LAMP ADI,SKDKEY,1      ; Turn lamp on
261      END
262      ;
263      ; BEGIN DINs:
264      ;
265 0109      ASSIGN PDIN = 'TSH-DSH-3' ; JFB02
266 0110      ASSIGN PTYPE = TYPE1
267 0111      ASSIGN PREV = FRV1
268 0112      ASSIGN PSTAT = INT1
269 0113      ASSIGN PPASS = FPAS1
270 0114      LET CURENT = 0
271 0115      LET LAST = 0           ; Used to set lamp status
272 0116      LET ANALOG = 0
273 0117      LET NXT = 0
274 0118      LMPVAR = 0
275 0119      BITVAL = 0

```

```

276 ;
277 0120 LOP: FOR ILOOP = 1 TO 2
278 ; IF ISELECT() THEN GOTO AGAIN
279 0121 IF ( PTYP > 0 ) THEN ; If some type of alarm
280 BEGIN
281 IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3 ;be1 ; If EPN inhibited
282 0123 IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5 ; If EPN is bad input
283 0125 IF IAND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4 ; If EPN is disabled
284 0127 IF ( PSTAT < 3 ) THEN ; If not inhibited
285 0128 BEGIN ;be2
286 0129 IF (PSTAT = 1) THEN ;be3
287 BEGIN
288 0130 IF (PTYP != 1) THEN
289 0132 IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
290 0134 IF (FTRIP4) THEN GOSUB ACK1
291 END ;en3
292 0137 IF PSTAT = 0 THEN IF (PDIN:DIN_VAL = PREV) THEN GOSUB ALM
293 0138 IF (PSTAT) THEN
294 BEGIN ;be4
295 0140 IF (PDIN:DIN_VAL != PREV) THEN GOSUB CLR
296 0141 NXT = PSTAT
297 0143 IF (PPASS) & (PSTAT = 2) THEN NXT = 0
298 0144 LAST = OR(LAST,NXT)
299 END ;en4
300 END ;en2
301 0146 ELSE IF IAND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0
302 END ;en1
303 0147 STEP PDIN BY 1
304 0148 STEP PTYP BY 1
305 0149 STEP PREV BY 1
306 0150 STEP PSTAT BY 1
307 0151 STEP PPASS BY 1
308 0152 NEXT ILOOP
309 ;
310 ;
311 ;
312 ; BEGIN ANALOG RED AND YELLOWS:
313 ;
314 0153 ASSIGN PAI = 'T1-EA1-7'
315 0154 ASSIGN PSKID = SKID3
316 0155 FOR JLOOP = 3 TO 23
317 ; IF ISELECT() THEN GOTO AGAIN
318 0156 IF (PTYP) THEN ; If some alarm exists
319 BEGIN ;be1
320 BITVAL = PAI:AI_ALMCT
321 0158 IF IAND(BITVAL,16) THEN ; If not inhibited
322 BEGIN ;be2
323 0159 IF AND(BITVAL,15) THEN ; If alarm exists and is current
324 BEGIN ;be3
325 0161 IF AND(BITVAL,32) THEN NXT = 2 ; If alarm is ack
326 ELSE ; If alarm is unack
327 BEGIN ;be4
328 0162 IF (PSKID != BITVAL) THEN
329 BEGIN ;be5
330 0163 LET PSKID = BITVAL

```

```

331 0165          IF BITVAL < 3 THEN FALMON = 1
332 0166          PDEV:DEV_TMP0 = 2
333 0167          NXT = 1
334          END                                ;en5
335 0168          ELSE NXT = 1
336          END                                ;en4
337          END                                ;en3
338          ELSE
339              BEGIN
340 0169              NXT = 0
341 0170              LET PSKID = 0
342              END
343          END                                ;en2
344          ELSE                                ; Else alarm is inhibited
345              BEGIN
346 0171              NXT = 0
347 0172              LET PSKID = 0
348              END
349 0173          ANALOG = OR(ANALOG,NXT)
350          END                                ;en1
351 0174          STEP PREV BY 1
352 0175          STEP PSTAT BY 1
353 0176          STEP PSKID BY 1
354 0177          STEP PTYPE BY 1
355 0178          STEP PAI BY 1
356 0179          NEXT JLOOP
357          ;
358          END ANALOG RED AND YELLOWS
359          ;
360          BEGIN ANALOG WHITE ALARMS:
361          ;
362 0180  WHT:  WHITE = 0
363 0181          ASSIGN PALARM = A10
364 0182          ASSIGN PREV = FRV24
365 0183          ASSIGN PPASS = FPAS24                ; JFB01
366 0184          ASSIGN PSTAT = INT24
367 0185          ASSIGN PTYPE = TYPE24
368          ;
369 0186          ASSIGN PAI = 'P1-EA1-13'
370 0187          ILOOP = 24
371 0188          GOSUB ANLG                        ; LOLO
372 0189          GOSUB ANLG                        ; LO
373 0190          ASSIGN PAI = 'F1C-EA1-1'
374 0191          GOSUB ANLG                        ; LO
375 0192          ASSIGN PAI = 'P1-EA1-1'
376 0193          GOSUB ANLG                        ; LO
377 0194          GOSUB ANLG                        ; HI
378 0195          ASSIGN PAI = 'TI-EA1-5'
379 0196          GOSUB ANLG                        ; HI
380          ;
381 0198          IF(INT24 = 3 | INT25 = 3) THEN LET LEAK0 = 9
382 0200          ELSE IF(INT24) THEN LET LEAK0 = INT24
383 0202          ELSE IF(INT25) THEN LET LEAK0 = INT25 + 2
384 0203          ELSE LET LEAK0 = 0
385 0205          IF(INT27 = 3 | INT28 = 3) THEN LET LEAK1 = 9

```

```

386 0207      ELSE IF(INT27)THEN LET LEAK1 = INT27 + 2
387 0209      ELSE IF(INT28)THEN LET LEAK1 = INT28 + 4
388 0210      ELSE LET LEAK1 = 0
389 0211      FTRIP4 = 0
390
391      ;
392      ;
393      ;
394 0212      MARK STEP "BELOW",0,0
395 0214      IF LAST > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
396 0215      CURENT = OR(LAST,WHITE)
397 0216      CURENT = OR(CURENT,ANALOG)
398 0218      IF AND(CURENT,1) THEN LMPVAR = 2
399 0220      ELSE IF AND(CURENT,2) THEN LMPVAR = 1
400 0221      ELSE LMPVAR = 0
401 0222      LAMP ADI,SKDKEY,LMPVAR
402 0223      MARK STEP "BOTTOM",0,0
403
404      ;
405      ;
406      ;
407 0224      NOCRIT
408 0225      ASSIGN FLAGBITS = AT2
409 0226      BVAL = 4
410 0227      FOR IN1 = 2 TO 13
411 0228      FLAGBITS = AND(FVEA1,BVAL)
412 0229      BVAL = BVAL ( 1
413 0230      STEP FLAGBITS BY 1
414 0231      NEXT IN1
415 0232      ASSIGN FLAGBITS = SAO
416 0233      BVAL = 1
417 0234      FOR IN1 = 0 TO 4
418 0235      FLAGBITS = AND(INLK2,BVAL)
419 0236      BVAL = BVAL ( 1
420 0237      STEP FLAGBITS BY 1
421 0238      NEXT IN1
422 0239      SA5 = AND(INLK1,1)
423 0240      SA6 = AND(INLK1,2)
424 0241      SA7 = AND(INLK1,4)
425 0242      GT7 = AND(G_bits,128)
426 0243      GT12 = AND(G_bits,4096)
427 0244      GT14 = AND(G_bits,16384)
428 0245      AT10 = AND(FVEA1,2048)
429 0246      AT18 = AND(STAT_BITS,4)
430 0247      AT19 = AND(STAT_BITS,32)
431 0248      AT20 = AND('RSH-CA1-1':DIN_VAL,'PSH-CA111':DIN_VAL)
432 0249      AT21 = AND('RSH-RC1-1':DIN_VAL,'RSH-EA1-1':DIN_VAL)
433 0250      AT22 = AND('PSH-FC51':DIN_VAL,'RSH-VVB/G':DIN_VAL)
434 0251      AT23 = AND('TSH-DSH-3':DIN_VAL,AT20)
435 0252      AT24 = AND(AT20,AT21)
436 0253      AT25 = AND(AT22,AT23)
437 0254      AT26 = AND(AT24,AT25)
438 0255      AT27 = AND(AT26,AT18)
439 0256      SB0 = AND('EA1-INLK1':DEV_STAT,32)
440 0257      SB1 = AND('EA1-INLK2':DEV_STAT,32)
441 0258      SB2 = AND('TV-DSH1/2':DEV_STAT,32)
442 0259      BT0 = AND(AT8,AT9)

```

```

441 0259      IF AND('EA1-ALARM':DEV_FLG,64) THEN PUTMANL 'EA1-ALARM'
442 0261      IF AND('FV-EA1-1':DEV_FLG,64) THEN PUTMANL 'FV-EA1-1'
443 0262      IF AND('PIC-EA116':DEV_FLG,64) THEN PUTMANL 'PIC-EA116'
444 0264      IF AND('EA1-INLK1':DEV_FLG,64) THEN PUTMANL 'EA1-INLK1'
445 0266      IF AND('EA1-INLK2':DEV_FLG,64) THEN PUTMANL 'EA1-INLK2'
446 0268      IF AND('TV-DSH1/2':DEV_FLG,64) THEN PUTMANL 'TV-DSH1/2'
447 0270      IF AND('SELECT-TI':DEV_FLG,64) THEN PUTMANL 'SELECT-TI'
448          ;
449          ;
450          ;
451          ;
452          ;
453          ;
454          ;
455          ;
456 0272      IF AND('FV-EA1-1':DEV_TMP1,4) THEN DLEAK3 = 1          ; Permissive ($T18) must be Hi (1)
457 0273      IF !AND('FV-EA1-1':DEV_TMP1,4) & AND(DLEAK3,1) THEN    ; Permissive ($T18) lost and 1st pass
458          BEGIN
459          PUTMANL 'FIC-EA1-1'
460 0274      PUTMANL 'PIC-EA116'          ; JF803
461 0276      PRINT AD1," PERMISSIVE LOST >>> FIC-EA1-1 PUT IN MANUAL"
462 0277      PRINT AD1," PERMISSIVE LOST >>> PIC-EA116 PUT IN MANUAL"
463 0278      PUTOUT 'FIC-EA1-1',A119
464 0279      PUTOUT 'PIC-EA116',A119
465 0280      PRINT AD1," PERMISSIVE LOST >>> FIC-EA1-1 OUTPUT SET TO ZERO"
466 0281      PRINT AD1," PERMISSIVE LOST >>> PIC-EA116 OUTPUT SET TO ZERO"
467 0282      DLEAK3 = 0
468          END
469          ;
470          ;
471          ;
472          ;
473 0283      NOCRIT
474          GOTO AGAIN
475          ;
476          ;
477          ;
478          ;
479          ;
480 0284      ANLG: IF !SELECT() THEN GOTO AGAIN          ; JF802
481          ANLG: IF (PTYPE = 1) THEN                  ; If some alarm
482          BEGIN
483          BITVAL = PAI:AI_ALMCT          ;be1
484 0285      IF !AND(BITVAL,16) THEN          ; If not inhibited
485          BEGIN
486 0288      IF PSTAT = 3 THEN LET PSTAT = 0          ;be2
487 0289      IF (PREV) THEN
488          BEGIN
489          IF (PSTAT = 0) THEN          ;be3
490          ; If rev acting, LOL0, or LO
491          ; If not previously alarmed

```


WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

12:38:26 24-Jan-96

Page 10

```

489      BEGIN                                ;be4
490      0292      IF (PAI:AI_MEAS <= PALARM) THEN GOSUB ALM
491      END                                        ;en4
492      ELSE
493      BEGIN
494      0293      IF (PSTAT = 1) THEN                ;be5      ; If previously alarmed
495      BEGIN
496      0295      IF (FTRIP4) THEN GOSUB ACK1          ; If ACK flag set, then ack it
497      0298      IF PTYPE != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
498      END
499      0300      IF (PAI:AI_MEAS >= PALARM + PAI:AI_DEAD) THEN GOSUB CLR
500      END
501      END
502      ELSE
503      BEGIN
504      0301      IF (PSTAT = 0) THEN                ;be10      ; If fail-safe, HIHI, or HI
505      BEGIN
506      0303      IF (PAI:AI_MEAS >= PALARM) THEN GOSUB ALM
507      END
508      ELSE
509      BEGIN
510      0304      IF (PSTAT = 1) THEN                ;be12
511      BEGIN
512      0306      IF (FTRIP4) THEN GOSUB ACK1          ;be13
513      0309      IF PTYPE != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
514      END
515      0311      IF (PAI:AI_MEAS <= PALARM - PAI:AI_DEAD) THEN GOSUB CLR
516      END
517      END
518      0312      NXT = PSTAT
519      0314      IF (PPASS) & (PSTAT = 2) THEN NXT = 0
520      END
521      ELSE
522      BEGIN
523      0315      LET PSTAT = 3
524      0316      NXT = 0
525      END
526      0317      WHITE = OR(WHITE,NXT)
527      END
528      0318      ELSE LET PSTAT = 0
529      0319      ILOOP = ILOOP + 1
530      0320      STEP PALARM BY 1
531      0321      STEP PSTAT BY 1
532      0322      STEP PTYPE BY 1
533      0323      STEP PREV BY 1
534      0324      STEP PPASS BY 1
535      0325      RETURN
536      ;
537      ;
538      ;
539      ;
540      ;
541      0326      CLR:      LET PSTAT = 0
542      ;
543      ; If not white, then unack:

```

```

544      ;
545 0329      IF PTYPE = 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
546 0331      IF ILOOP = 24 THEN INT25 = 0
547 0332      LET FCLEAR = 1 ; Clear flag
548 0333      GOSUB ACKK ; Print message
549 0334      RETURN
550      ;
551 0335      ALM: LET PSTAT = 1 ; Int set to one
552      ;
553      ; If not white, then unack:
554      ;
555      ; IF PTYPE = 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
556      ;
557      ; Turn audible alarm on if white or yellow:
558      ;
559      ; CRIT
560 0337      IF AND(PTYPE,3) THEN FALMON = 1
561 0338      PDEV:DEV_TMPD = 2
562 0339      GOSUB ACKK ; Print message
563 0340      RETURN
564      ;
565 0341      ACK1: LET PSTAT = 2 ; Ack the alarm
566 0342      LET FACK = 1 ; Set ACK flag for C program
567      ;
568      ; ALARM TYPER STRINGS:
569      ;
570      ;ACKK: IF ILOOP = 0 THEN STRNGO = "E-A-1 10 LB STEAM PRESSURE LO"
571 0344      ACKK: IF ILOOP = 1 THEN STRNGO = "DESUPER HEAT OUTLET TEMP HI "
572 0346      IF ILOOP = 24 THEN STRNGO = "PI-EA1-13 STM HDR PRESS LOLO"
573 0348      IF ILOOP = 25 THEN STRNGO = "PI-EA1-13 STM HDR PRESS LO "
574 0350      IF ILOOP = 26 THEN STRNGO = "FIC-EA1-1 REBOIL STM FLO LOW"
575 0352      IF ILOOP = 27 THEN STRNGO = "PI-EA1-1 REBOIL STM PRESS LO"
576 0354      IF ILOOP = 28 THEN STRNGO = "PI-EA1-1 REBOIL STM PRESS HI"
577 0356      IF ILOOP = 29 THEN STRNGO = "TI-EA1-5 STM CONDNSB TEMP HI "
578 0358      IF ILOOP = 34 THEN STRNGO = "START EA1BATCH PGM "
579 0360      IF ILOOP = 35 THEN STRNGO = "EA1BATCH WRONG UNIT"
580 0362      IF ILOOP = 36 THEN STRNGO = "TERMINATE EA1BATCH "
581      ;
582      ; If acknowledging the alarm, then:
583      ;
584      ;
585 0363      MSG: IF PTYPE = 1 THEN
586      BEGIN
587      ;
588      ; If white alarm, print:
589      ;
590 0365      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNGO
591 0366      IF (FACK) THEN
592      BEGIN
593      ;
594      ; If white, print it is ack and reset the flag:
595      ;
596 0367      PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
597 0368      LET FACK = 0
598      END
599      END

```

```

599      ;
600      ; If CLEAR, print it is clear and reset the flag:
601      ;
602      0369      IF (FCLEAR) THEN
603      BEGIN
604      0370      PRINT AD1,"%p ALARM CLEAR",STRNGO
605      0371      LET FCLEAR = 0
606      END
607      ;
608      ; If STARTUP, print startup message:
609      ;
610      0372      IF (FSTART) THEN
611      BEGIN
612      0373      LET FSTART = 0
613      0374      PRINT AD1,"%p",STRNGO
614      END
615      0375      RETURN
616      ;
617      ; If ERROR - print message, start timer, and set flag:
618      ;
619      0376      ERRSUB: IF ERRN != ERRNUM() THEN
620      BEGIN
621      0377      TIMERON STIMEO
622      0378      TIMERCLR STIMEO
623      0379      LET FSN0 = 1
624      0380      LET ERRN = ERRNUM()
625      0381      LET ERRL = ERRLIN()
626      0382      PRINT AD1,"ERROR # %i AT LINE %i",errn,errl
627      END
628      0383      ERSCLR
629      0384      GOTO AGAIN
630      ;
631      ; If FABORT = 1, then abort program:
632      ;
633      ; TERM: LET FABORT = 0
634      0385      TERM: LET STRNGO = "EAT1BATCH TERMINATED "
635      0386      LET ILOOP = STAR + 2
636      0387      LET FSTART = 1
637      0388      GOSUB ACKK
638      END

```

```

Total Errors Detected 7      Total Warnings 0
Sequence Program Size = 7504 Bytes, 469 Pages
24 Forward Reference(s)

```

EC1BATCH

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:34:42 24-Feb-95

Page 1

```

1      ;      E C 1 B A T C H . S E Q      VERSION 2
2      ;
3      ;      Function:      Sequence program for Lamp function,
4      ;                      message display on Graphic number 016,
5      ;                      and printer messages.
6      ;
7      ;
8      ;      Written By      : R. N. Kersey
9      ;      Date           : 05-31-92
10     ;      Facility        : 242-A Evaporator Bldg.
11     ;      Machine         : VAX 4104-A
12     ;      Op. System       : VAX/VMS 5.5-2AH
13     ;      D/3 Version      : 6.3-2 BASELINE
14     ;      Compiler         : SIC
15     ;      Build command    : @[320,20]EC1BATCH
16     ;
17     ;      UNIT = 10, SKDKEY = 38
18     ;
19     ;      Assoc. UPF:      [300,1]_PROCESS.____ PROCESS.LST
20     ;      Assoc. RPF:      [300,1]_242A.____ 242A.LST
21     ;      SDB:            [300,1]PCMOA.SDB PCMOASDB.CMD
22     ;
23     ;      POINTER      SDB VAR.      FUNCTION
24     ;      -----
25     ;      PSTAT      INTn      Current EPN alarm state is an integer tied to
26     ;                      graphic flashing string of value:
27     ;                      0 = previously not alarmed
28     ;                      1 = previously alarmed, not ack
29     ;                      2 = previously alarmed, is ack
30     ;                      3 = alarm is inhibited
31     ;                      4 = input is off scan
32     ;                      5 = input is bad
33     ;
34     ;      LEAKn      Special INT variable for multiple white alarms on an EPN
35     ;      PDIN      'EPN'
36     ;      PTYPE      TYPEn      alarm type 0 = none, 1 = white, 2 = yellow,
37     ;                      3 = yellow and yellow, 4 = red,
38     ;                      6 = yellow and red
39     ;      PREV      FRVn      Reverse acting (not fail safe) flag (1=set)
40     ;      PSKID      SKIDn      Used for analog alarms
41     ;      PPASS      FPA5n      Turn lamp off when alarm is ACKnowledged
42     ;      PALARM      FALMn      Analog alarm flag
43     ;      ---      FWING      Unused error flag
44     ;
45     ;      MODIFICATION LOG:
46     ;
47     ;      09-DEC-92      Changed FIC-EC3-1 LO alarm from 140 to 50      M. Teats
48     ;                      and added comments for FIC-EC3-1.
49     ;
50     ;      30-DEC-92      Fixed white alarm print message.      M. Teats
51     ;
52     ;      07-JUN-93      Added comments. CCM# 93JUN01-01.      M. Teats
53     ;
54     ;      mct1+ 21-JUN-93      Changed TYPE25 = 0 to TYPE25 = 1. Baselined      M. Teats
55     ;                      for D/3 version 3.3. CCM# 93JUN21-01.
56     ;
57     ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger

```

```

56      ;          system. Rebaselined document. Removed
57      ;          obsolet Select() statement.
58      ;
59      ;   END MOD LOG
60      ;
61      UNIT      EC1
62      ;
63      COMMON WHOLE0
64      HVEC1 == 'HV-EC1-1':DEV_TMP1
65      HVEC2 == 'HVEC2/3-1':DEV_TMP1
66      FLAGBITS == PSPARO
67      G_bits == PB2GLOBAL:DEV_TMPO
68      ;
69      0000      NOP
70      0001      STAR = 34          ; Start message
71      0002      LET SKDKEY = 38          ; SKID Key
72      0003      LET UNITT = 10          ; SEQ Unit number
73      0004      ASSIGN PDEV = 'EC1-ALARM' ; Alarm device
74      0005      ASSIGN PTYPE = TYPE34
75      0006      LET ADI = 0          ; Alarm destination index
76      0007      LET FSN0 = 0          ; No alarm yet
77      0008      TIMEROFF STIME0          ; Alarm timer off
78      0009      TIMERCCLR STIME0          ; Alarm timer clear
79      0010      LET STRNG0 = " "          ; Blank 30 character string
80      0011      LET ERRN = 0          ; Set error number to zero
81      0012      LET ERRL = 0          ; Set error line to zero
82      0013      LET FTRIP4 = 0          ; FTRIP4 = 1 when audible horn is to be turned on
83      0014      GETUNIT TRIP          ; Which sequence unit is program running in
84      0015      IF TRIP != UNITT THEN          ; Abort if running in wrong seq. unit
85          BEGIN
86      0016          FSTART = 1
87      0017          ILOOP = STAR + 1
88      0018          PTYPE = 0
89      0019          GOSUB ACKK
90      0020          GOTO TERM
91      END
92      ;   IF !SELECT() THEN GOTO AGAIN
93      ;
94      ;   B E G I N   I N I T I A L I Z I N G
95      ;
96      0021      INIT: LET FABORT = 0          ; If FABORT = 1 then abort program
97      0022      LET FSTART = 0          ; Program start flag = 0
98      0023      LAMP ADI,SKDKEY,0          ; Turn off the annunciator key
99      0024      LET LSG0 = " "          ; Clear the string variables
100     0025      LET LSG1 = " "
101     0026      LET LSG2 = " "
102     0027      LET LSG3 = " "
103     0028      LET LSG4 = " "
104     0029      PDEV:DEV_TMPO = 1          ; Set alarm device to non-alarm state
105     0030      IF AND(PDEV:DEV_ALMCT,16) THEN          ; If alarm device is inhibited, remove the inhibit
106     0031      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
107     0032      IF AND(PDEV:DEV_ALMCT,32) THEN          ; If alarm device is ack, clear it
108     0033      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
109      ;
110      ;   First, set all flags and integers:

```

```

111      ;
112 0034      ASSIGN PDIN = 'RSH-CA1-1'      ; Assign pointer to first DIN
113 0035      ASSIGN PSTAT = INTO
114 0036      ASSIGN PTYPE = TYPE0
115 0037      ASSIGN PREV = FRV0
116 0038      ASSIGN PSKID = SKID0
117 0039      ASSIGN PPASS = FPAS0
118      ;
119 0040      FOR ILOOP = 0 TO 25
120 0041          PPASS = 0
121 0042          PREV = 0
122 0044          IF PSTAT > 5 THEN PSTAT = 0      ; Reset currnet status if undefined
123 0045          PSKID = 0      ; Default previous status to no alarms
124 0046          PTYPE = 0      ; Default to no alarms
125 0047          IF ILOOP = 0 THEN      ; For the only DIN
126              BEGIN
127 0048                  IF AND(PDIN:DIN_DFLG,3) THEN      ; Check for alarms if defined (must be red or yellow)
128                      BEGIN
129 0049                          LET PTYPE = 2      ; Default to yellow alarm
130 0051                          IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1      ; Reverse acting
131 0053                          IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4      ; Set to red critical alarm
132                          END
133                      END
134 0055                  IF ILOOP > 22 THEN PTYPE = 1      ; Set to white alarm
135 0056                  STEP PPASS BY 1
136 0057                  STEP PREV BY 1
137 0058                  STEP PSTAT BY 1
138 0059                  STEP PSKID BY 1
139 0060                  STEP PTYPE BY 1
140 0061                  NEXT ILOOP
141      ;
142 0062          TYPE19 = 4
143 0063          TYPE23 = 1      ; PI-EC1-5
144 0064          TYPE24 = 1      ; FIC-EC3-1
145 0065          TYPE25 = 1      ; TIC-EC1-2 +mct1-
146      ;
147 0066          FRV0 = 0
148 0067          FRV1 = 0
149 0068          FRV24 = 1      ; FIC-EC3-1 LO
150      ;
151 0069          A10 = 100.0      ; PI-EC1-5 HI
152 0070          A11 = 50.0      ; FIC-EC3-1 LO
153 0071          A12 = 110.0      ; TIC-EC1-2 HI
154      ;
155      END INITIALIZING
156      ;
157      SEND PROGRAM START MESSAGE
158      ;
159 0072      ANALOG = 0
160 0073      ILOOP = STAR
161 0074      FSTART = 1
162 0075      PTYPE = 0
163 0076      GOSUB ACKK
164 0077      ON ERROR GOSUB ERRSUB
165      ;

```

```

166      ; BEGIN PROGRAM LOOP
167      ;
168 0078 AGAIN: MARK STEP "WTUNTIL",0,0
169      ; WAITUNTIL SELECT() ; FABORT ; Wait until micro is selected or program aborted
170      ; IF (FABORT) THEN GOTO TERM ; Terminate program if abort flag set
171 0079 MARK STEP "RNNING",0,0
172 0080 WAIT 1
173 0081 IF (FSNO) THEN ; If alarm flag is set
174 0082 IF STIME0 > 60 THEN ; If alarm flag active for 1 minute
175      BEGIN ; Reset timer and turn off flag
176 0083 TIMEROFF STIME0
177 0084 TIMERCLR STIME0
178 0085 LET FSNO = 0
179 0086 LET ERRN = 0
180 0087 LET ERRL = 0
181      END
182      ;
183      ; If EC1-ALARM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
184      ;
185 0088 IF (PDEV:DEV_TMPO = 2) & AND(PDEV:DEV_ALMCT,32) THEN
186      BEGIN
187 0089 PDEV:DEV_TMPO = 1 ; Put EC1-ALARM in ack state
188 0090 LET FTRIP4 = 1 ; Set for unit alarm ack, system to graphic
189 0091 LET ACK = 1 ; Set for unit alarm ack, graphic to system
190 0092 CRIT
191 0093 LAMP ADI,SKDKEY,1 ; Turn lamp on
192      END
193      ;
194      ; BEGIN DINS:
195      ;
196 0094 ASSIGN PDIN = 'RSH-CA1-1'
197 0095 ASSIGN PTYPE = TYPE0
198 0096 ASSIGN PREV = FRVO
199 0097 ASSIGN PSTAT = INTO
200 0098 ASSIGN PPASS = FPASO
201 0099 LET CURENT = 0 ; Used to set lamp status
202 0100 LET LAST = 0
203 0101 LET ANALOG = 0
204 0102 LET NXT = 0
205 0103 LMPVAR = 0
206 0104 BITVAL = 0
207      ;
208 0105 LOP: IF ( PTYPE > 0 ) THEN ; If some type of alarm
209      BEGIN ;be1
210 0107 IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3 ; If EPN inhibited
211 0109 IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5 ; If EPN is bad input
212 0111 IF (AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4 ; If EPN is disabled
213 0112 IF ( PSTAT < 3 ) THEN ; If not inhibited
214      BEGIN ;be2
215 0113 IF (PSTAT = 1) THEN
216      BEGIN ;be3
217 0114 IF (PTYPE != 1) THEN
218 0116 IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
219 0118 IF (FTRIP4) THEN GOSUB ACK1
220      END ;en3

```


[illegible]

```

276          END
277 0156      ANALOG = OR(ANALOG,NXT)
278          END                                ;en1
279 0157      STEP PREV BY 1
280 0158      STEP PSTAT BY 1
281 0159      STEP PSKID BY 1
282 0160      STEP PTYPER BY 1
283 0161      STEP PAI BY 1
284 0162      NEXT JLOOP
285          ;
286          ;
287          ;
288          ;
289          ;
290 0163      WHITE = 0
291 0164      ASSIGN PALARM = A10
292 0165      ASSIGN PREV = FRV23
293 0166      ASSIGN PSTAT = INT23
294 0167      ASSIGN PTYPER = TYPE23
295          ;
296 0168      ASSIGN PAI = 'PI-EC1-5'
297 0169      LOOP = 23
298 0170      GOSUB ANLG
299 0171      ASSIGN PAI = 'FIC-EC3-1'
300 0172      GOSUB ANLG
301 0173      ASSIGN PAI = 'TIC-EC1-2'
302 0174      GOSUB ANLG
303          ;
304          ;
305          ;
306 0175      MARK STEP "BELOW",0,0
307 0176      FTRIP4 = 0
308 0178      IF LAST > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
309 0179      CURENT = OR(LAST,WHITE)
310 0180      CURENT = OR(CURENT,ANALOG)
311 0182      IF AND(CURENT,1) THEN LMPVAR = 2
312 0184      ELSE IF AND(CURENT,2) THEN LMPVAR = 1
313 0185      ELSE LMPVAR = 0
314 0186      LAMP ADI,SKDKEY,LMPVAR
315 0187      MARK STEP "BOTTOM",0,0
316 0188      NOCRIT
317          ;
318          ;
319          ;
320 0189      AT25 = AND(HVEC1,512)
321 0190      AT26 = AND(HVEC1,1024)
322 0191      AT27 = AND(HVEC1,2048)
323 0192      AT28 = AND(HVEC1,4096)
324 0193      AT29 = AND(HVEC1,8192)
325 0194      AT31 = AND(HVEC1,32768)
326          ;
327 0195      SA0 = 'ZS-EC1-1N':DIN_VAL
328 0196      SA1 = 'ZS-EC1-1C':DIN_VAL
329          ;
330 0197      SA4 = and('HV-EC1-1':DEV_STAT,4) ; RESET

```

```

331 0198      SA5 = AND('HV-EC1-1':DEV_STAT,16)      ; CHGSTATE
332      ;
333 0199      BT27 = AND(HVEC2,2048)                  ; START TIMER
334 0200      BT28 = AND(HVEC2,4096)                  ; TIMER OVER
335 0201      BT29 = AND(HVEC2,8192)                  ; EX-C-1 OFF
336 0202      BT31 = AND(HVEC2,32768)                  ; INTERLOCK
337      ;
338 0203      SA2 = 'ZSEC2/31N':DIN_VAL
339 0204      SA3 = 'ZSEC2/31C':DIN_VAL
340      ;
341 0205      SB0 = AND('HVEC2/3-1':DEV_STAT,4)
342 0206      SB1 = AND('HVEC2/3-1':DEV_STAT,16)
343 0207      SB2 = AND('HVEC2/3-1':DEV_STAT,32)
344      ;
345 0208      AT3 = AND('RSH-VVB/G':DIN_VAL,'TSH-DSH-3':DIN_VAL)
346 0209      AT4 = AND('WFSH-CA11':DIN_VAL,'WFSH-CA12':DIN_VAL)
347 0210      AT5 = AND('RSH-CA1-1':DIN_VAL,'PDSH-CA1':DIN_VAL)
348 0211      AT6 = AND(AT3,AT4)
349 0212      AT7 = AND(AT5,AT6)
350      ;
351 0213      IF !BT27 THEN
352 0214          IF !FLGO THEN
353              BEGIN
354 0215                  TIMERCLR OSTMR
355 0216                  TIMERON OSTMR
356 0217                  FLGO = 1
357                      END
358      ;
359 0218      IF FLGO THEN
360          BEGIN
361 0219              IF OSTMR >= 255 THEN
362                  BEGIN
363 0220                      TIMEROFF OSTMR
364 0221                      TIMERCLR OSTMR
365                      END
366          END
367      IF BT27 THEN FLGO = 0
368 0225      IF AND('EC1-ALARM':DEV_FLG,64) THEN PUTMANL 'EC1-ALARM'
369 0227      IF AND('HV-EC1-1':DEV_FLG,64) THEN PUTMANL 'HV-EC1-1'
370 0229      IF AND('HVEC2/3-1':DEV_FLG,64) THEN PUTMANL 'HVEC2/3-1'
371      ;
372      ;
373      ;
374 0230      GOTO AGAIN
375      ;
376      ;
377      ;
378      ;
379      ;
380      ;
381 0231      ANLG: IF !SELECT() THEN GOTO AGAIN
382      ;
383 0232      IF (PTYPE = 1) THEN                                ;be1
384 0233          BEGIN
385              BITVAL = PAI:AI_ALMCT
386              IF !(AND(BITVAL,16)) THEN                                ; If not inhibited
387                  BEGIN
388                      ;be2

```

```

386 0235      IF PSTAT = 3 THEN LET PSTAT = 0
387 0236      IF (PREV) THEN                                ; If rev acting, LOLO, or LO
388          BEGIN                                           ;be3
389 0237      IF (PSTAT = 0) THEN                                ; If not previously alarmed
390          BEGIN                                           ;be4
391 0239      IF (PAI:AI_MEAS <= PALARM) THEN GOSUB ALM
392          END                                             ;en4
393      ELSE
394          BEGIN                                           ;be5
395 0240      IF (PSTAT = 1) THEN                                ; If previously alarmed
396          BEGIN                                           ;be6
397 0242      IF (FTRIP4) THEN GOSUB ACK1                        ; If ACK flag set, then ack it
398 0245      IF PTYPE != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
399          END                                             ;en6
400 0247      IF (PAI:AI_MEAS >= PALARM + PAI:AI_DEAD) THEN GOSUB CLR
401          END                                             ;en5
402      END                                             ;en3
403  ELSE
404          BEGIN                                           ;be10
405 0248      IF (PSTAT = 0) THEN
406          BEGIN                                           ;be11
407 0250      IF (PAI:AI_MEAS >= PALARM) THEN GOSUB ALM
408          END                                             ;en11
409  ELSE
410          BEGIN                                           ;be12
411 0251      IF (PSTAT = 1) THEN
412          BEGIN                                           ;be13
413 0253      IF (FTRIP4) THEN GOSUB ACK1
414 0256      IF PTYPE != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
415          END                                             ;en13
416 0258      IF (PAI:AI_MEAS <= PALARM - PAI:AI_DEAD) THEN GOSUB CLR
417          END                                             ;en12
418      END                                             ;en10
419 0259      NXT = PSTAT
420 0261      IF (PPASS) & (PSTAT = 2) THEN NXT = 0
421      END                                             ;en2
422  ELSE
423          BEGIN
424 0262      LET PSTAT = 3
425 0263      NXT = 0
426      END
427 0264      WHITE = OR(WHITE,NXT)
428      END
429 0265      ELSE LET PSTAT = 0
430 0266      ILOOP = ILOOP + 1
431 0267      STEP PALARM BY 1
432 0268      STEP PSTAT BY 1
433 0269      STEP PTYPE BY 1
434 0270      STEP PREV BY 1
435 0271      STEP PPASS BY 1
436 0272      RETURN
437      ;
438      ;
439      ;
440      ;

```

END ANALOG WHITE SUBROUTINE

BEGIN CLEAR, ALARM, ACKNOWLEDGE, MESSAGE, ERROR, AND TERMINATE SUBROUTINES:

```

441      ;
442 0273 CLR: LET PSTAT = 0                                ; Int set to zero
443      ;
444      ; If not white, then unack:
445      ;
446 0276 IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
447 0277 FCLEAR = 1                                         ; Clear flag
448 0278 GOSUB ACKK                                         ; Print message
449 0279 RETURN
450      ;
451 0280 ALM: LET PSTAT = 1                                ; Int set to one
452      ;
453      ; If not white, then unack:
454      ;
455 0283 IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
456      ;
457      ; Turn audible alarm on if white or yellow:
458      ;
459 0284 CRIT
460 0286 IF AND(PTYPE,3) THEN FALMON = 1
461 0287 PDEV:DEV_TMP0 = 2
462 0288 GOSUB ACKK                                         ; Print message
463 0289 RETURN
464      ;
465 0290 ACK1: LET PSTAT = 2                                ; Ack the alarm
466 0291 FACK = 1                                           ; Set ACK flag for C program
467      ;
468      ; ALARM TYPER STRINGS:
469      ;
470 0293 ACKK: IF ILOOP = 0 THEN STRNG0 = "E-C-1 PC RADIATION HI/FAIL"
471 0295 IF ILOOP = 23 THEN STRNG0 = "E-C-1 RAW WATER INLET PRESS HI"
472 0297 IF ILOOP = 24 THEN STRNG0 = "E-C-3 USED RAW WATER FLOW LOW"
473 0299 IF ILOOP = 25 THEN STRNG0 = "E-C-1 OUTLET TEMPERATURE HIGH"
474 0301 IF ILOOP = 34 THEN STRNG0 = "START ECTBATCH PGM "
475 0303 IF ILOOP = 35 THEN STRNG0 = "ECTBATCH WRONG UNIT"
476 0305 IF ILOOP = 36 THEN STRNG0 = "TERMINATE ECTBATCH "
477      ;
478      ; If acknowledging alarm, then:
479      ;
480 0306 AMSG: IF PTYPE = 1 THEN
481      BEGIN
482      ;
483      ; If white alarm, print:
484      ;
485 0308 IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT AD1,"WHITE ALARM %p",STRNG0
486 0309 IF (FACK) THEN
487      BEGIN
488      ;
489      ; If white, print it is ack and reset the flag:
490      ;
491 0310 PRINT AD1,"%p ALARM ACKNOWLEDGE",STRNG0
492 0311 FACK = 0
493      END
494      END
495      ;

```

```

496      ; If CLEAR, print it is clear and reset flag:
497      ;
498      0312      IF (FCLEAR) THEN
499              BEGIN
500      0313          PRINT ADI,"%p ALARM CLEAR",STRNGO
501      0314          FCLEAR = 0
502              END
503      ;
504      ; If STARTUP, print startup message:
505      ;
506      0315      IF (FSTART) THEN
507              BEGIN
508      0316          FSTART = 0
509      0317          PRINT ADI,"%p",STRNGO
510              END
511      0318      RETURN
512      ;
513      ; If ERROR - print message, start timer, and set flag:
514      ;
515      0320      ERRSUB: IF(ERRNUM() = -187)THEN ERSRET
516              ELSE
517                  BEGIN
518      0321          IF ERRN != ERRNUM() THEN
519                  BEGIN
520      0322              TIMERON STIMEO
521      0323              TIMERCLR STIMEO
522      0324              FSNO = 1
523      0325              ERRN = ERRNUM()
524      0326              ERRL = ERRLIN()
525      0327              PRINT ADI,"ERROR # %i AT LINE %i",errn,errl
526      0328              ERSCLR
527                  END
528              END
529      0329      GOTO AGAIN
530      ;
531      ; If FABORT = 1, then abort program:
532      ;
533      ; TERM: FABORT = 0
534      ; TERM: STRNGO = "EC1BATCH TERMINATED"
535      0330      TERM: STRNGO = "EC1BATCH TERMINATED"
536      0331      ILOOP = STAR + 2
537      0332      FSTART = 1
538      0333      GOSUB ACKK
539      END

```

```

Total Errors Detected      0      Total Warnings      0
Sequence Program Size = 6288 Bytes, 393 Pages
21 Forward Reference(s)

```

~~XXXX~~ RMIS View/Print Document Cover Sheet ~~XXXX~~

This document was retrieved from the Documentation and Records Management (DRM) ISEARCH System. It is intended for Information only and may not be the most recent or updated version. Contact a Document Service Center (see Hanford Info for locations) if you need additional retrieval information.

Accession #: D196079540

Document #: SD-534-CSWL-001

Title/Desc:

SEQUENCE & BATCH LANGUAGE PROGRAMS & ALARM RELATED
C PROGRAMS FOR THE 242A MCS [SEC 4 OF 6 PAGE 384
TO 511]

Pages: 130

This document was too large to scan as a whole document,
therefore it required breaking into smaller sections.

Document number: SD-534-CSWL-001

Section 4 of 6

Title: Sequence and Batch Language programs
and Alarm Related "C" Programs for the
242-A MCS

Date: 4/15/96 Revision: 3

Originator: Burger, J.F

Co: _____

Recipient: _____

Co: _____

References: ECN- 627504

RWBATCH

```

1      ;      R W B A T C H . S E Q                      VERSION 2
2      ;
3      ;      Function:      Sequence program for Lamp function,
4      ;                      message display on Graphic number 011,
5      ;                      and printer messages.
6      ;
7      ;      Written By      : R. N. Kersey
8      ;      Date           : 05-31-92
9      ;      Facility       : 242-A Evaporator Bldg.
10     ;      Machine        : VAX 4104-A
11     ;      Op. System     : VAX/VMS 5.5-2AH
12     ;      D/3 Version    : 6.3-2 BASELINE
13     ;      Compiler       : SIC
14     ;      Build command  : @([320,20]RWBATCH
15     ;
16     ;      UNIT = 11, SKDKEY = 43
17     ;
18     ;      Assoc. UPF:      ([300,1]_PROCESS,___)    PROCESS.LST
19     ;      Assoc. RPF:      ([300,1]_242A,___)      242A.LST
20     ;      SDB:            ([300,1]PCMDA.SDB)       PCMDASDB.CMD
21     ;
22     ;      POINTER      SDB VAR.      FUNCTION
23     ;      -----
24     ;      PSTAT      INTn      Current EPN alarm state is an integer tied to
25     ;                      graphic flashing string of value:
26     ;      .      (n=0-100)      0 = previously not alarmed
27     ;                      1 = previously alarmed, not ack
28     ;                      2 = previously alarmed, is ack
29     ;                      3 = alarm is inhibited
30     ;                      4 = input is off scan
31     ;                      5 = input is bad
32     ;
33     ;      PDIN      LEAKn      Special INT variable for multiple white alarms on an EPN
34     ;      PTYPE      'EPN'      ---
35     ;                      alarm type 0 = none, 1 = white, 2 = yellow,
36     ;                      3 = yellow and yellow, 4 = red,
37     ;                      6 = yellow and red
38     ;      PREV      FRVn      Reverse acting (not fail safe) flag (1=set)
39     ;      PSKID      SKIDn      Used for analog alarms
40     ;      PPASS      FPASn      Turn lamp off when alarm is ACKnowledged
41     ;      PALARM      FALMn      Analog alarm flag
42     ;      ---      FWING      Unused error flag
43     ;
44     ;      MODIFICATION LOG:
45     ;
46     ;      11-AUG-91      - Fixed order of variables so alarms will      M. Teats
47     ;                      appear on graphic (PDI, TI, and TSL alarms
48     ;                      were not working)
49     ;      - Added comments
50     ;
51     ;      30-DEC-92      Fixed white alarm print message.      M. Teats
52     ;
53     ;      07-JUN-93      Added comments and baselined for      M. Teats
54     ;                      D/3 version 3.3. CCN# 93JUN01-01.
55     ;
56     ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger

```

```

56      ;                      system. Rebaselined document. Removed
57      ;                      obsolete Select() statement
58      ;
59      ;   END MOD LOG
60      ;
61      UNIT      RWATER
62      COMMON XYZFRM,PB2
63      ;
64      TSL == 'TSL242A81'
65      AUTO1 == 'FQ1-RW-1':CB_FLG:1      ; For FQ1-RESET AUTO mode check
66      AUTO1B == 'FQ1-RW-1B':CB_FLG:1    ; For FQ1-RESET AUTO mode check
67      CALOUT1 == 'FQ1-RW-1':CB_OTVL:1   ; For FQ1-RESET calc block outvalue
68      CALOUT1B == 'FQ1-RW-1B':CB_OTVL:1 ; For FQ1-RESET calc block outvalue
69      ;
70      0000      NOP
71      0001      LET STAR = 33              ; Start message
72      0002      LET SKDKEY = 43            ; SKID Key
73      0003      LET UNITT = 11            ; SEQ Unit number
74      0004      ASSIGN PDEV = 'RW-ALARM' ; Alarm device
75      0005      ASSIGN PTYPE = TYPE33
76      0006      PTYPE = 0                  ; No start message yet
77      0007      LET ADI = 0                ; Alarm destination index
78      0008      LET FSWO = 0              ; No alarm yet
79      0009      TIMEROFF STIMEO           ; Alarm timer off
80      0010      TIMERCLR STIMEO           ; Alarm timer clear
81      0011      STRNGO = " "              ; Blank 30 character string
82      0012      LET ERRN = 0              ; Set error number to zero
83      0013      LET ERRL = 0              ; Set error line to zero
84      0014      FTRIP4 = 0                ; FTRIP4 = 1 when audible horn is to be turned on
85      0015      GETUNIT TRIP              ; Which sequence unit is program running in
86      0016      IF TRIP != UNITT THEN     ; Abort if running in wrong seq. unit
87      BEGIN
88      0017      FSTART = 1
89      0018      ILOOP = STAR + 1
90      0019      GOSUB ACKK
91      0020      FSTART = 1
92      0021      PRINT ADI," >>>>> ATTEMPT TO RUN RWBATCH IN WRONG UNIT"
93      0022      GOTO TERM
94      END
95      ;   IF (SELECT())=0 THEN GOTO AGAIN
96      ;
97      ;   BEGIN  I N I T I A L I Z I N G
98      ;
99      0023      INIT: LET FABORT = 0        ; If FABORT = 1 then abort program
100     0024      LET FSTART = 0              ; Program start flag = 0
101     0025      LET SDO = 0                 ; FQ1-RESET manual mode flag = 0
102     0026      LET SD1 = 0                 ; FQ1-RESET manual mode flag = 0
103     0027      LAMP ADI,SKDKEY,0          ; Turn off the annunciator key
104     0028      LSGO = " "                 ; Clear the string variables
105     0029      LSG1 = " "
106     0030      LSG2 = " "
107     0031      LSG3 = " "
108     0032      LSG4 = " "
109     0033      PDEV:DEV.TMPO = 1           ; Set alarm device to non-alarm state
110     0034      IF AND(PDEV:DEV_ALMCT,16) THEN ; If alarm device is inhibited, remove the inhibit

```

```

111 0035      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
112 0036      IF AND(PDEV:DEV_ALMCT,32) THEN      ; If alarm device is ack, clear it
113 0037      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
114      ;
115      ; AIN, DIN, DEV - ILOOP, TYPE, LIMITS:
116      ;
117      ; Digitals:
118      ; -----
119      ; ILOOP  EPN          TYPE
120      ; 0      LDS-BFP-1    1
121      ; 1      LSL-RWSMP    1
122      ; 2      LDS-BFP-2    1
123      ; 3      RSH-RW-1     4
124      ; 4      RXS-RW-1     2
125      ;
126      ; Analog Reds or Yellows:
127      ; -----
128      ; ILOOP  EPN          TYPE  LIMIT
129      ; 5      PDI-RW-1     0      NA
130      ; 6      PI-RW-1     2      60 Low, 120 Hi
131      ; 7      PDI-RW-2     0      NA
132      ; 8      RI-RW-1     4      NA
133      ; 9      PI-RW-2     0      NA
134      ;
135      ; Analog Whites:
136      ; -----
137      ; ILOOP  EPN          TYPE  AI  LIMIT
138      ; 10     PDI-RW-1     1      A10 13.0 Hi
139      ; 11     PDI-RW-2     1      A11 13.0 Hi
140      ; 12     PI-RW-2     1      A12 60.0 Low
141      ;
142      ; PCN1 DIN:
143      ; -----
144      ; ILOOP  EPN          TYPE
145      ; 13     TSL242A81    2
146      ;
147      ; PCN0 DEV:
148      ; -----
149      ; ILOOP  EPN          TYPE
150      ; 14     HV-RW-1     2
151      ; 15     HV-RW-2     2
152      ;
153      ; Set all PCN0 flags and integers:
154      ;
155 0038      ASSIGN PDIN = 'LDS-BFP-1'      ; Assign pointer to first DIN
156 0039      ASSIGN PTYPE = TYPE0
157 0040      ASSIGN PREV = FRV0
158 0041      ASSIGN PSTAT = INTO
159 0042      ASSIGN PSKID = SKID0
160 0043      ASSIGN PPASS = FPAS0
161      ;
162 0044      FOR ILOOP = 0 TO 4      ; For PCN0 DINs
163 0045      LET PPASS = 0
164 0046      LET PREV = 0
165 0048      IF PSTAT > 5 THEN PSTAT = 0      ; Reset current status if undefined

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:36:25 24-Feb-95

Page 4

```

166 0049      LET PSKID = 0                ; Default previous status to no alarms
167 0050      LET PTYPE = 1                ; Default to white alarm
168 0051      IF AND(PDIN:DIN_DFLG,3) THEN ; Check for alarms (Must be red or yellow)
169          BEGIN
170 0052          PTYPE = 2                ; Default to Yellow
171 0054          IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1 ; Reverse acting
172 0056          IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4 ; Critical red alarm
173          END
174 0057      STEP PDIN BY 1
175 0058      STEP PTYPE BY 1
176 0059      STEP PREV BY 1
177 0060      STEP PSTAT BY 1
178 0061      STEP PSKID BY 1
179 0062      STEP PPASS BY 1
180 0063      NEXT ILOOP
181
182 0064      ; FOR ILOOP = 5 TO 15          ; For everything else
183 0065      PTYPE = 0
184 0066      PREV = 0
185 0067      PSTAT = 0
186 0068      PSKID = 0
187 0069      PPASS = 0
188 0070      STEP PTYPE BY 1
189 0071      STEP PREV BY 1
190 0072      STEP PSTAT BY 1
191 0073      STEP PSKID BY 1
192 0074      STEP PPASS BY 1
193 0075      NEXT ILOOP
194
195      ;
196      ; Set alarm type if not zero for above analogs:
197      ;
197 0076      LET TYPE6 = 2
198 0077      LET TYPE8 = 4
199 0078      LET TYPE10 = 1
200 0079      LET TYPE11 = 1
201 0080      LET TYPE12 = 1
202 0081      LET TYPE13 = 2
203 0082      LET TYPE14 = 2
204 0083      LET TYPE15 = 2
205
206 0084      ; LET FRV12 = 1                ; PI-RW-2 low
207      ;
208      ; Set White alarm limits:
209      ;
210 0085      LET A10 = 13.0                ; PD1-RW-1 hi
211 0086      LET A11 = 13.0                ; PD1-RW-2 hi
212 0087      LET A12 = 60.0                ; PI-RW-2 low
213
214      ;
215      ; I N I T I A L I Z I N G   D O N E
216      ;
217      ; S E N D   P R O G R A M   S T A R T   M E S S A G E
218 0088      ANALOG = 0
219 0089      ILOOP = STAR
220 0090      FSTART = 1

```

```

221 0091      GOSUB ACKK
222 0092      ON ERROR GOSUB ERRSUB
223      ;
224      ;      B E G I N   P R O G R A M   L O O P
225      ;
226 0093  AGAIN: MARK STEP "WTUNTL",0,0
227      ;      WAITUNTIL SELECT() ; FABORT      ; Wait until micro is selected or program aborted
228      ;      IF (FABORT) THEN GOTO TERM      ; Terminate program if abort flag set
229 0094      MARK STEP "RNNING",0,0
230 0095      WAIT 1
231      ;
232 0096      IF (FSNO) THEN      ; If alarm flag is set
233 0097      IF STIME0 > 60 THEN      ; If alarm flag active for 1 minute
234      BEGIN      ; Reset timer and turn off flag
235 0098      TIMEROFF STIME0
236 0099      TIMERCLR STIME0
237 0100      LET FSNO = 0
238 0101      LET ERRN = 0
239 0102      LET ERRL = 0
240      END
241      ;
242      ;
243      ;
244 0103      IF (PDEV:DEV_TMPO = 2) & AND(PDEV:DEV_ALMCT,32) THEN
245      BEGIN
246 0104      PDEV:DEV_TMPO = 1      ; Put RW-ALARM in ack state
247 0105      LET FTRIP4 = 1      ; Set for unit alarm ack, system to graphic
248 0106      LET ACK = 1      ; Set for unit alarm ack, graphic to system
249 0107      CRIT
250 0108      LAMP ADI,SKDKEY,1      ; Turn lamp on
251      END
252      ;
253      ;
254      ;      B E G I N   D I G I T A L S   ( P C M O ) :
255 0109      ASSIGN PDIN = 'LDS-BFP-1'
256 0110      ASSIGN PTYPE = TYPE0
257 0111      ASSIGN PREV = FRV0
258 0112      ASSIGN PSTAT = INI0
259 0113      ASSIGN PSKD = SKID0
260 0114      ASSIGN PPASS = FPAS0
261 0115      LET LAST = 0
262 0116      LET NXT = 0
263 0117      LMPVAR = 0
264 0118      BITVAL = 0
265      ;
266 0119  LOP:   FOR ILOOP = 0 TO 4
267      ;      IF !SELECT() THEN GOTO AGAIN
268 0120      IF (PTYPE > 0) THEN      ; If some type of alarm
269      BEGIN      ;be1
270 0122      IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3      ; If EPN Inhibited
271 0124      IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5      ; If EPN is Bad input
272 0126      IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4      ; If EPN is disabled
273 0127      IF (PSTAT < 3) THEN      ; If not inhibited
274      BEGIN      ;be2
275 0128      IF (PSTAT = 1) THEN

```

```

276          BEGIN                                ;be3
277 0129      IF (PTYPE != 1) THEN                    ; If not white
278 0131      IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
279 0133      IF (PTRIP4) THEN GOSUB ACK1
280          END                                ;en3
281 0136      IF PSTAT = 0 THEN IF (PDIN:DIN_VAL = PREV) THEN GOSUB ALM
282 0137      IF (PSTAT) THEN
283          BEGIN                                ;be4
284 0139      IF (PDIN:DIN_VAL != PREV) THEN GOSUB CLR
285 0140      NXT = PSTAT
286 0142      IF (PPASS) & (PSTAT = 2) THEN NXT = 0
287 0143      LAST = OR(LAST,NXT)
288          END                                ;en4
289          END                                ;en2
290 0145      ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0
291          END                                ;en1
292 0146      STEP PDIN BY 1
293 0147      STEP PTYPE BY 1
294 0148      STEP PREV BY 1
295 0149      STEP PSTAT BY 1
296 0150      STEP PSKID BY 1
297 0151      STEP PPASS BY 1
298 0152      NEXT ILOOP
299      ;
300      ;      END DIGITALS (PCMD)
301      ;
302      ;      BEGIN ANALOG RED AND YELLOWS:
303      ;
304 0153      STRNGO = "RUN"
305 0154      LET ANALOG = 0
306 0155      ASSIGN PAI = 'PDI-RW-1'
307 0156      ASSIGN PSKID = SKID5
308 0157      ASSIGN PTYPE = TYPE5
309      ;
310 0158      FOR ILOOP = 5 TO 9
311      ;      IF !SELECT() THEN GOTO AGAIN
312 0159      IF (PTYPE) THEN                                ; If some alarm exists
313          BEGIN
314 0160      BITVAL = PAI:AI_ALMCT                                ;be1
315 0161      IF !AND(BITVAL,16) THEN                                ; If not inhibited
316          BEGIN
317 0162      IF AND(BITVAL,15) THEN                                ; If alarm exists and is current
318          BEGIN
319 0164      IF AND(BITVAL,32) THEN NXT = 2                                ; If alarm is ack
320          ELSE                                ; If alarm is unack
321          BEGIN
322 0165      IF PSKID !=BITVAL THEN                                ;be4
323          BEGIN
324 0167      IF BITVAL < 3 THEN FALMON = 1                                ;be5
325 0168      LET PSKID = BITVAL
326 0169      PDEV:DEV_TMP0 = 2
327 0170      NXT = 1
328          END                                ;en5
329 0171      ELSE NXT = 1
330          END                                ;en4

```

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:36:25 24-Feb-95

Page 7

```

331          END                                ;en3
332      ELSE                                ; Else there is no alarm
333          BEGIN
334      0172      NXT = 0
335      0173      LET PSKID = 0
336          END
337      END                                ;en2
338      ELSE                                ; Else alarm is inhibited
339          BEGIN
340      0174      NXT = 0
341      0175      LET PSKID = 0
342          END
343      0176      ANALOG = OR(ANALOG,NXT)
344          END                                ;en1
345      0177      STEP PSKID BY 1
346      0178      STEP PTYPY BY 1
347      0179      STEP PAI BY 1
348      0180      NEXT ILOOP
349      ;
350      ;
351      ;
352      ;
353      ;
354      0181      WHITE = 0
355      0182      ASSIGN PTYPY = TYPE10
356      0183      ASSIGN PSTAT = INT10
357      0184      ASSIGN PALARM = A10
358      0185      ASSIGN PPASS = FPAS10
359      0186      ASSIGN PREV = FRV10
360      0187      ILOOP = 10
361      ;
362      0188      ASSIGN PAI = 'PDI-RW-1'
363      0189      GOSUB ANLG                                ; Hi
364      0190      ASSIGN PAI = 'PDI-RW-2'
365      0191      GOSUB ANLG                                ; Hi
366      0192      ASSIGN PAI = 'PI-RW-2'
367      0193      GOSUB ANLG                                ; Low
368      ;
369      ;
370      ;
371      ;
372      ;
373      0195      IF (ERRNUM() = -187) THEN GOTO SKP1
374      0196      IF TYPE13 > 0 THEN                                ; Skip TSL242A81 if PCM1 micros down
375          BEGIN                                ; If some type of alarm
376      0198      IF AND('TSL':DIN ALMCT,16) THEN INT13 = 3
377      0200      IF AND('TSL':DIN ALMCT,4) THEN INT13 = 5
378      0202      IF (AND('TSL':DIN FLG,8) THEN INT13 = 4
379      0203      IF (INT13 < 3) THEN
380          BEGIN                                ;be2
381      0204      IF (INT13 = 1) THEN
382          BEGIN                                ;be3
383      0205      IF (TYPE13 != 1) THEN
384      0207      IF AND('TSL':DIN ALMCT,32) THEN GOSUB ACK2
385      0209      IF (FTRIP4) THEN GOSUB ACK2

```



```

386          END                                ;en3
387 0210      IF (INT13 = 0) THEN
388 0212          IF ('TSL':DIN_VAL = FRV13) THEN GOSUB ALM2
389 0213      IF (INT13) THEN
390          BEGIN                                ;be4
391 0215          IF ('TSL':DIN_VAL != FRV13) THEN GOSUB CLR2
392 0216          NXT = INT13
393 0218          IF (FPAS13) & (INT13 = 2) THEN NXT = 0
394 0219          LAST = OR(LAST,NXT)
395          END                                ;en4
396          END                                ;en2
397 0221      ELSE IF !AND('TSL':DIN_ALMCT,16)&!AND('TSL':DIN_ALMCT,4)&AND('TSL':DIN_FLG,8)THEN INT13 = 0
398          END                                ;en1
399      ;
400      ;
401      ;
402      ;
403      ;
404 0222  HV:  DEVO = 0
405 0223          ASSIGN PDIN = 'HV-RW-1'
406 0224          ASSIGN PTYPE = TYPE14
407 0225          ASSIGN PSTAT = INT14
408      ;
409 0226      FOR ILOOP = 14 TO 15
410 0227      IF PTYPE > 0 THEN                                ; If some type of alarm
411          BEGIN                                ;be1
412 0229          IF AND(PDIN:DEV_ALMCT,16) THEN LET PSTAT = 3      ; If EPN is Inhibited
413 0231          IF AND(PDIN:DEV_ALMCT,4) THEN LET PSTAT = 5      ; If EPN is Bad input
414 0233          IF !AND(PDIN:DEV_FLG,8) THEN LET PSTAT = 4      ; If EPN is disabled
415 0234          IF (PSTAT < 3) THEN                                ; If not inhibited
416              BEGIN                                ;be2
417 0235              IF (PSTAT = 1) THEN
418                  BEGIN                                ;be3
419 0236                  IF (PTYPE != 1) THEN
420 0238                      IF AND(PDIN:DEV_ALMCT,32) THEN GOSUB ACK1
421 0240                      IF (FTRIP4) THEN GOSUB ACK1
422                      END                                ;en3
423 0242                  IF AND(PDIN:DEV_CMD,4) THEN GOSUB ALM1
424 0243                  IF (PSTAT) THEN
425                      BEGIN                                ;be4
426 0245                      IF !AND(PDIN:DEV_CMD,4) THEN GOSUB CLR
427 0246                      NXT = PSTAT
428 0248                      IF (PPASS) & (PSTAT = 2) THEN NXT = 0
429 0249                      LAST = OR(LAST,NXT)
430                      END                                ;en4
431                  END                                ;en2
432 0251              ELSE IF !AND(PDIN:DEV_ALMCT,16)&!AND(PDIN:DEV_ALMCT,4)&AND(PDIN:DEV_FLG,8)THEN LET PSTAT = 0
433                  END                                ;en1
434 0252          STEP PDIN BY 1
435 0253          STEP PTYPE BY 1
436 0254          STEP PREV BY 1
437 0255          STEP PSTAT BY 1
438 0256          STEP PSKID BY 1
439 0257          STEP PPASS BY 1
440 0258          NEXT ILOOP

```

```

441      ;
442      ;      END HANDVALVE DEVICE ALARMS
443      ;
444      0259      SKP1:  MARK STEP "BELOW",0,0
445      0260                      SORT = 0
446      0261                      FTRIP4 = 0
447      0263                      IF LAST > 3 THEN PRINT AD1," LAMP VARIABLE IS GREATER THAN 3!"
448      0264                      CURENT = OR(WHITE,ANALOG)
449      0265                      CURENT = OR(LAST,CURENT)
450      0266                      CURENT = OR(DEVO,CURENT)
451      0268                      IF AND(CURENT,1) THEN LMPVAR = 2
452      0270                      ELSE IF AND(CURENT,2) THEN LMPVAR = 1
453      0271                      ELSE LMPVAR = 0
454      0272                      LAMP AD1,SKDKEY,LMPVAR
455      0273                      MARK STEP "BOTTOM",0,0
456      0274                      NOCRIT
457      ;
458      ;      BEGIN FQI-RESET:
459      ;
460      ;      Function:      For totalizers that have a CALC block which performs the
461      ;                      integration, set the output of the CALC block to zero
462      ;                      after the FQI loop has been placed in AUTO from MANUAL.
463      ;                      Reset both FQI-RW-1 and -1B if either one is reset.
464      ;
465      0276                      IF !AND(AUTO1,64) THEN SD0 = 1                      ; If manual, set manual flag
466      0277                      IF AND(AUTO1,64) & AND(SD0,1) THEN                      ; For manual-AUTO transition
467      BEGIN
468      0278                      SD0 = 0
469      0279                      CALOUT1 = 0.0
470      0280                      CALOUT1B = 0.0
471      END
472      ;
473      0282                      IF !AND(AUTO1B,64) THEN SD1 = 1                      ; If manual, set manual flag
474      0283                      IF AND(AUTO1B,64) & AND(SD1,1) THEN                      ; For manual-AUTO transition
475      BEGIN
476      0284                      SD1 = 0
477      0285                      CALOUT1B = 0.0
478      0286                      CALOUT1 = 0.0
479      END
480      ;
481      ;      END FQI-RESET
482      ;
483      0288                      IF AND('RW-ALARM':DEV_FLG,64) THEN PUTMANL 'RW-ALARM'
484      0289                      GOTO AGAIN
485      ;
486      ;      E N D   P R O G R A M   L O O P
487      ;
488      ;      BEGIN ANALOG WHITE SUBROUTINE:
489      ;
490      ;      ANLG:  IF !SELECT() THEN GOTO AGAIN
491      0290      ANLG:  IF (PTYPE = 1) THEN                      ; If some alarm
492      BEGIN
493      0291                      BITVAL = PAI:A1_ALMCT                      ;be1
494      0292                      IF !AND(BITVAL,16) THEN                      ; If not Inhibited
495      BEGIN

```

```

496 0294      IF PSTAT = 3 THEN LET PSTAT = 0
497 0295      IF (PREV) THEN                                ; If rev acting, LOLO, or LO
498          BEGIN                                          ;be3
499 0296      IF (PSTAT = 0) THEN                            ; If not previously alarmed
500          BEGIN                                          ;be4
501 0298      IF (PA1:AI_MEAS <= PALARM) THEN GOSUB ALM
502          END                                            ;en4
503      ELSE
504          BEGIN                                          ;be5
505 0299      IF (PSTAT = 1) THEN                            ; If previously alarmed
506          BEGIN                                          ;be6
507 0301      IF (FTRIP4) THEN GOSUB ACK1                    ; If ACK flag set, then ack it
508 0304      IF PTYPY != 1 THEN IF AND(PA1:AI_ALMCT,32) THEN GOSUB ACK1
509          END                                            ;en6
510 0306      IF (PA1:AI_MEAS >= PALARM + PA1:AI_DEAD) THEN GOSUB CLR
511          END                                            ;en5
512      END                                              ;en3
513  ELSE
514      BEGIN                                          ;be10
515 0307      IF (PSTAT = 0) THEN
516          BEGIN                                          ;be11
517 0309      IF (PA1:AI_MEAS >= PALARM) THEN GOSUB ALM
518          END                                            ;en11
519      ELSE
520          BEGIN                                          ;be12
521 0310      IF (PSTAT = 1) THEN
522          BEGIN                                          ;be13
523 0312      IF (FTRIP4) THEN GOSUB ACK1
524 0315      IF PTYPY != 1 THEN IF AND(PA1:AI_ALMCT,32) THEN GOSUB ACK1
525          END                                            ;en13
526 0317      IF (PA1:AI_MEAS <= PALARM - PA1:AI_DEAD) THEN GOSUB CLR
527          END                                            ;en12
528      END                                              ;en10
529  NEXT = PSTAT
530 0320      IF (PPASS) & (PSTAT = 2) THEN NEXT = 0
531      END                                              ;en2
532  ELSE
533      BEGIN
534 0321      LET PSTAT = 3
535 0322      NEXT = 0
536      END
537 0323      WHITE = OR(WHITE,NXT)
538      END
539 0324      ELSE LET PSTAT = 0
540 0325      STEP PALARM BY 1
541 0326      STEP PSTAT BY 1
542 0327      STEP PTYPY BY 1
543 0328      STEP PREV BY 1
544 0329      STEP PPASS BY 1
545 0330      ILOOP = ILOOP + 1
546 0331      RETURN
547      ;
548      ;
549      ;
550      ;

```

END ANALOG WHITE SUBROUTINE

BEGIN CLEAR, ALARM, ACKNOWLEDGE, MESSAGE, ERROR, AND TERMINATE SUBROUTINES:

```

551      ;
552 0332 CLR:   LET PSTAT = 0                                ; PCMO DIN, white AI, DEV
553      ;
554      ; If not white, then unack:
555      ;
556 0335      IF PTYP1 != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
557 0336      LET FCLEAR = 1
558 0337      GOSUB ACKK
559 0338      RETURN
560      ;
561 0339 CLR2:  LET INT13 = 0                                ; PCM1 DIN
562      ;
563      ; If not white, then unack:
564      ;
565 0342      IF TYPE13 != 1 THEN IF AND('TSL':DIN_ALMCT,32)THEN 'TSL':DIN_ALMCT = XOR('TSL':DIN_ALMCT,32)
566 0343      LET FCLEAR = 1
567 0344      GOSUB ACKK
568 0345      RETURN
569      ;
570      ; If not white, then unack:
571      ;
572 0348 ALM:   IF PTYP1 != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
573      ;
574 0349 ALM1:  CRIT
575 0350      LET PSTAT = 1
576 0352      IF AND(PTYPE,3) THEN FALMON = 1
577 0353      PDEV:DEV_TMP0 = 2
578 0354      GOSUB ACKK
579 0355      RETURN
580      ;
581 0356 ALM2:  LET INT13 = 1                                ; PCM1 DIN
582      ;
583      ; If not white, then unack:
584      ;
585 0359      IF TYPE13 != 1 THEN IF AND('TSL':DIN_ALMCT,32)THEN 'TSL':DIN_ALMCT = XOR('TSL':DIN_ALMCT,32)
586 0360      'RW-ALARM':DEV_TMP0 = 2
587 0361      FALMON = 1
588 0362      GOSUB ACKK
589 0363      RETURN
590      ;
591 0364 ACK1:  LET PSTAT = 2
592 0365      LET FACK = 1
593 0366      GOSUB ACKK
594 0367      RETURN
595      ;
596 0368 ACK2:  LET INT13 = 2                                ; PCM1 DIN
597 0369      LET FACK = 1
598      ;
599      ; For multiple AI white alarms on one EPN (use with string packet 78):
600      ;
601 0371 ACKK:  IF (INT12 & INT12 < 3) THEN LET LEAKO = INT12 + 2          ; PI-RW-2 Lo
602 0373      ELSE IF INT12 = 3 THEN LET LEAKO = 9                      ; PI-RW-2 INHA
603 0374      ELSE LET LEAKO = 0                                         ; No alarm
604      ;
605      ; End multiple AI white alarms on one EPN

```

```

606      ;
607      ;      ALARM TYPER STRINGS:
608      ;
609      0376  MSSG:  IF ILOOP = 0 THEN STRNGO = "RW BLDG BFP-1 LEAK  " ; LDS-BFP-1
610      0378      IF ILOOP = 1 THEN STRNGO = "RW BLDG SUMP LEVEL LO " ; LSL-RWSMP
611      0380      IF ILOOP = 2 THEN STRNGO = "RW BLDG BFP-2 LEAK  " ; LDS-BFP-2
612      0382      IF ILOOP = 3 THEN STRNGO = "RW SUPPLY RAD HI    " ; RSH-RW-1
613      0384      IF ILOOP = 4 THEN STRNGO = "RW SUPPLY RAD SYS FAIL" ; RKS-RW-1
614      0386      IF ILOOP = 10 THEN STRNGO = "RW BLDG F-RW-1 DP HI " ; PDI-RW-1
615      0388      IF ILOOP = 11 THEN STRNGO = "RW BLDG F-RW-2 DP HI " ; PDI-RW-2
616      0390      IF ILOOP = 12 THEN STRNGO = "F-RW-1/2 IN PRESS LO " ; PI-RW-2
617      0392      IF ILOOP = 13 THEN STRNGO = "RW BLDG TEMP LO    " ; TSL242A81
618      0394      IF ILOOP = 14 THEN STRNGO = "HV-RW-1 OPEN 30 + SEC."
619      0396      IF ILOOP = 15 THEN STRNGO = "HV-RW-2 OPEN 30 + SEC."
620      0398      IF ILOOP = 33 THEN STRNGO = "RWBATCH PGM STARTING "
621      0400      IF ILOOP = 34 THEN STRNGO = "RWBATCH WRONG UNIT  "
622      0402      IF ILOOP = 35 THEN STRNGO = "RWBATCH TERMINATING "
623      ;
624      0403  AMSG:  IF PTYPE = 1 THEN
625      ;      BEGIN
626      ;
627      ;      ; If white alarm, print:
628      ;
629      0405      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNGO
630      0406      IF (FACK) THEN
631      ;      BEGIN
632      ;
633      ;      ; If white, print it is ack and reset flag:
634      ;
635      0407      PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
636      0408      LET FACK = 0
637      ;      END
638      ;
639      ;
640      ;      ; If CLEAR, print it is clear and reset flag:
641      ;
642      0409      IF (FCLEAR) THEN
643      ;      BEGIN
644      0410      PRINT ADI,"%p ALARM CLEAR",STRNGO
645      0411      LET FCLEAR = 0
646      ;      END
647      ;
648      ;      ; If STARTUP, print startup message:
649      ;
650      0412      IF (FSTART) THEN
651      ;      BEGIN
652      0413      LET FSTART = 0
653      0414      PRINT ADI,"%p",STRNGO
654      ;      END
655      0415      RETURN
656      ;
657      ;      ; If error - print message, start timer, and set flag:
658      ;
659      0416  ERRSUB: IF ERRN != ERRNUM() THEN
660      ;      BEGIN

```

```

661      0417      TIMEROW STIMEO
662      0418      TIMERCLR STIMEO
663      0419      LET ESNO = 1
664      0420      LET ERBN = ERBNLN()
665      0421      LET ERRL = ERRLIN()
666      0422      PRINT ADI,"ERROR # %i AT LINE %i",errn,errl
667      0423      END
668      0424      ERSCLR
669      0425      GOTD AGAIN
670      0426      ;
671      0427      ; If FABORT = 1, then abort program:
672      0428      ;
673      0429      ; TERM: LET FABORT = 0
674      0430      STRING = "RMBATCH TERMINATED "
675      0431      LET ILOOP = STAR + 2
676      0432      LET FSTART = 1
677      0433      SSNGO = "TERM"
678      0434      GOSUB ACKX
679      0435      END

```

Total Errors Detected 0 Total Warnings 0
 Sequence Program Size = 8224 Bytes, 514 Pages
 33 Forward Reference(s)

URWBATCH

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:36:56 24-Feb-95

Page 1

```

1      ; U R W B A T C H . S E Q                                VERSION 2
2      ;
3      ; Function:      Sequence program for Lamp function,
4      ;                message display on Graphic number 017,
5      ;                and printer messages.
6      ;
7      ; Written By      : R. W. Kersey
8      ; Date            : 05-31-92
9      ; Facility        : 242-A Evaporator Bldg.
10     ; Machine         : VAX 4104-A
11     ; Op. System      : VAX/VMS 5.5-2AH
12     ; D/3 Version     : 6.3-2 BASELINE
13     ; Compiler        : SIC
14     ; Build command   : @[320,20]URWBATCH
15     ;
16     ; UNIT = 12, SKDKEY = 42
17     ;
18     ; Assoc. UPF:      [300,1]_PROCESS.____ PROCESS.LST
19     ; Assoc. RPF:      [300,1]_242A.____ 242A.LST
20     ; SDB:            [300,1]PCMOA.SDB PCMOASDB.CMD
21     ;
22     ; POINTER      SDB VAR.      FUNCTION
23     ; -----
24     ; PSTAT      INTn      Current EPN alarm state is an integer tied to
25     ;                graphic flashing string of value:
26     ;                (n=0-100)      0 = previously not alarmed
27     ;                .              1 = previously alarmed, not ack
28     ;                .              2 = previously alarmed, is ack
29     ;                .              3 = alarm is inhibited
30     ;                .              4 = input is off scan
31     ;                .              5 = input is bad
32     ;                LEAKn      Special INT variable for multiple white alarms on an EPN
33     ;                PDIN      'EPN'
34     ;                PTYPEn      alarm type 0 = none, 1 = white, 2 = yellow,
35     ;                .              3 = yellow and yellow, 4 = red,
36     ;                .              6 = yellow and red
37     ;                PREV      FRVn      Reverse acting (not fail safe) flag (1=set)
38     ;                PSKID      SKIDn      Used for analog alarms
39     ;                PPASS      FPASn      Turn lamp off when alarm is ACKnowledged
40     ;                PALARM      FALMn      Analog alarm flag
41     ;                ---      FWING      Unused error flag
42     ;
43     ; MODIFICATION LOG:
44     ;
45     ; 02-JUN-92      Deleted O/C display bar feature.      M. Teats
46     ;
47     ; 08-JUL-92      Removed synchronized PID feature.      M. Teats
48     ;
49     ; 30-DEC-92      Fixed white alarm print message.      M. Teats
50     ;
51     ; 08-JUN-93      Added comments. CCN# 93JUN01-01.      M. Teats
52     ;
53     ; mct1+ 29-JUN-93      Remove IBAD on RC2-PIG and baselined for M. Teats
54     ;                D/3 version 3.3. CCN# 93JUN29-01.
55     ;

```



```

56      ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
57      ;
58      ;
59      ;
60      ;      END MOD LOG
61      ;
62      ;      UNIT      URW
63      ;
64      ;      COMMON PB2
65      ;      DEVON == 'RC2-SAMP':DEV_STAT      ; For FQIC-RESET device on/off check
66      ;      CALOUT == 'FQIC-RC2':CB_OTVL:3      ; For FQIC-RESET calc block outvalue
67      ;      RC2_PIG == 'RC2-PIG':DEV_TMP0
68      ;
69      0000      NOP
70      0001      STAR = 24      ; Start message
71      0002      LET SKDKEY = 42      ; SKID Key
72      0003      LET UNITT = 12      ; SEQ Unit number
73      0004      ASSIGN PDEV = 'URW-ALARM'      ; Alarm device
74      0005      ASSIGN PTYPE = TYPE24
75      0006      ADI = 0      ; Alarm destination index
76      0007      FSWO = 0      ; No alarm yet
77      0008      TIMEROFF STIMEO      ; Alarm timer off
78      0009      TIMERCLR STIMEO      ; Alarm timer clear
79      0010      STRNGO = " "      ; Blank 30 character string
80      0011      ERRN = 0      ; Set error number to zero
81      0012      ERRL = 0      ; Set error line to zero
82      0013      FTRIP4 = 0      ; FTRIP4 = 1 when audible horn is to be turned on
83      0014      GETUNIT TRIP      ; Which sequence unit is program running in
84      0015      IF TRIP != UNITT THEN      ; Abort if running in wrong seq. unit
85          BEGIN
86      0016          FSTART = 1
87      0017          ILOOP = STAR + 1
88      0018          PTYPE = 0
89      0019          GOSUB ACKK
90      0020          GOTO TERM
91      0021          END
92      ;      IF !SELECT() THEN GOTO AGAIN
93      ;
94      ;      B E G I N   I N I T I A L I Z I N G
95      ;
96      0021      STRNGO = "INIT"
97      0022      INIT:      FABORT = 0      ; If FABORT = 1, then abort program
98      0023      FSTART = 0      ; Program start flag = 0
99      0024      SDO = 0      ; For FQIC-RESET, RC2-SAMP device status flag
100     0025      LAMP ADI,SKDKEY,0      ; Turn off the annunciator key
101     0026      LSG0 = " "      ; Clear the string variables
102     0027      LSG1 = " "
103     0028      LSG2 = " "
104     0029      LSG3 = " "
105     0030      LSG4 = " "
106     0031      PDEV:DEV_TMP0 = 1      ; Set alarm device to non-alarm state
107     0032      IF AND(PDEV:DEV_ALMCT,16) THEN      ; If alarm device is inhibited, remove the inhibit
108     0033          PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
109     0034      IF AND(PDEV:DEV_ALMCT,32) THEN      ; If alarm device is ack, clear it
110     0035          PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)

```

```

111      ;
112      ; First, set all flags and integers:
113      ;
114 0036      ASSIGN PDIN = 'FSL-RC2-1' ; Assign pointer to first DIN
115 0037      ASSIGN PSTAT = INTO
116 0038      ASSIGN PTYPE = TYPE0
117 0039      ASSIGN PREV = FRV0
118 0040      ASSIGN PSKID = SKID0
119 0041      ASSIGN PPASS = FPAS0
120      ;
121 0042      FOR ILOOP = 0 TO 15
122 0043          PPASS = 0
123 0044          PREV = 0
124 0046          IF PSTAT > 5 THEN PSTAT = 0 ; Reset current status if undefined
125 0047          PSKID = 0 ; Default previous status to no alarms
126 0048          PTYPE = 0 ; Default to no alarms
127 0049          IF ILOOP < 4 THEN ; For DINs
128              BEGIN
129 0050              IF AND(PDIN:DIN_DFLG,3) THEN ; Check for alarms if defined (must be red or yellow)
130                  BEGIN
131 0051                  PTYPE = 2 ; Default to yellow alarm
132 0053                  IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1 ; Reverse acting
133 0055                  IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4 ; Set to red critical alarm
134                  END
135              END
136 0057          IF ILOOP > 3 THEN PTYPE = 0
137 0058          STEP PPASS BY 1
138 0059          STEP PREV BY 1
139 0060          STEP PSTAT BY 1
140 0061          STEP PSKID BY 1
141 0062          STEP PTYPE BY 1
142 0063          NEXT ILOOP
143      ;
144 0064      TYPE0 = 0
145 0065      TYPE1 = 4
146 0066      TYPE2 = 2
147 0067      TYPE3 = 2
148 0068      TYPE4 = 1
149 0069      TYPE10 = 4
150 0070      TYPE11 = 2
151 0071      TYPE12 = 2
152 0072      TYPE15 = 1
153 0073      TYPE16 = 0
154 0074      TYPE17 = 2
155      ;
156 0075      FRV0 = 0
157 0076      FRV1 = 0
158 0077      FRV2 = 0
159 0078      FRV3 = 1
160 0079      FRV15 = 1
161 0080      FRV16 = 0
162 0081      FRV17 = 1
163      ;
164 0082      A10 = 1200.0
165      ;

```

```

166      ;      END      INITIALIZING
167      ;
168      ;      SEND      PROGRAM      START      MESSAGE
169      ;
170      0083      ANALOG = 0
171      0084      LOOP = STAR
172      0085      PTYPE = 0
173      0086      FSTART = 1
174      0087      GOSUB ACKK
175      0088      ON ERROR GOSUB ERRSUB
176      ;
177      ;      BEGIN      PROGRAM      LOOP
178      ;
179      0089      AGAIN: MARK STEP "WTUNTL",0,0
180      ;      WAITUNTIL SELECT() ; FABORT      ; Wait until micro is selected or program aborted
181      ;      IF ( FABORT) THEN GOTO TERM      ; Terminate program if abort flag set
182      0090      MARK STEP "RNNING",0,0
183      0091      WAIT 1
184      0092      IF (FSNO) THEN      ; If alarm flag is set
185      0093      IF (FSNO) > 60 THEN      ; If alarm flag active for 1 minute
186      BEGIN      ; Reset timer and turn off flag
187      0094      TIMEROFF STIMEO
188      0095      TIMERCLR STIMEO
189      0096      LET FSNO = 0
190      0097      LET ERRN = 0
191      0098      LET ERRL = 0
192      END
193      ;
194      ;      If URW-ALARM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
195      ;
196      0099      IF (PDEV:DEV_TMO = 2) & AND(PDEV:DEV_ALMCT,32) THEN
197      BEGIN
198      0100      PDEV:DEV_TMO = 1      ; Put URW-ALARM in ack state
199      0101      LET FTRIP4 = 1      ; Set for unit alarm ack, system to graphic
200      0102      LET ACK = 1      ; Set for unit alarm ack, graphic to system
201      0103      CRIT
202      0104      LAMP ADI,SKDKEY,1      ; Turn lamp on
203      END
204      ;
205      ;      BEGIN DINs:
206      ;
207      0105      ASSIGN PDIN = 'FSL-RC2-1'
208      0106      ASSIGN PTYPE = TYPEO
209      0107      ASSIGN PREV = FRVO
210      0108      ASSIGN PSTAT = INTO
211      0109      ASSIGN PPASS = FPASO
212      0110      LET CURENT = 0      ; Used to set lamp status
213      0111      LET LAST = 0
214      0112      LET ANALOG = 0
215      0113      LET NXT = 0
216      0114      LMPVAR = 0
217      0115      BITVAL = 0
218      ;
219      0116      LOP:      FOR ILOOP = 0 TO 3
220      ;      IF ISELECT() THEN GOTO AGAIN

```

```

221 0117      IF ( PTYPE > 0 ) THEN                                ; If some type of alarm
222      BEGIN                                                    ;be1
223      IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3            ; If EPN is inhibited
224 0121      IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5        ; If EPN is bad input
225 0123      IF 1AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4        ; If EPN is disabled
226 0124      IF ( PSTAT < 3 ) THEN                                ; If not inhibited
227      BEGIN                                                    ;be2
228 0125      IF (PSTAT = 1) THEN
229      BEGIN                                                    ;be3
230 0126      IF (PTYPE != 1) THEN
231 0128      IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
232 0130      IF (FTRIP4) THEN GOSUB ACK1
233      END                                                    ;en3
234 0133      IF PSTAT = 0 THEN IF (PDIN:DIN_VAL = PREV) THEN GOSUB ALM
235 0134      IF (PSTAT) THEN
236      BEGIN                                                    ;be4
237 0136      IF PDIN:DIN_VAL != PREV THEN GOSUB CLR
238 0137      NXT = PSTAT
239 0139      IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
240 0140      LAST = OR(LAST,NXT)
241      END                                                    ;en4
242      END                                                    ;en2
243 0142      ELSE IF 1AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0
244      END                                                    ;en1
245 0143      STEP PDIN BY 1
246 0144      STEP PTYPE BY 1
247 0145      STEP PREV BY 1
248 0146      STEP PSTAT BY 1
249 0147      STEP PPASS BY 1
250 0148      NEXT ILOOP
251      ;
252      ;
253      ;
254      ;
255      ;
256 0149      ASSIGN PSKID = SKID4
257 0150      ASSIGN PAI = 'FIC-EC1-1'
258 0151      FOR JLOOP = 4 TO 14
259      ;
260 0152      IF !SELECT() THEN GOTO AGAIN
261      IF (PTYPE) THEN                                ; If some alarm exists
262      BEGIN                                                    ;be1
263 0153      BITVAL = PAI:AI_ALMCT
264 0154      IF 1AND(BITVAL,16) THEN                                ; If not inhibited
265      BEGIN                                                    ;be2
266 0155      IF AND(BITVAL,15) THEN                                ; If alarm exists and is current
267      BEGIN                                                    ;be3
268 0157      IF AND(BITVAL,32) THEN NXT = 2                        ; If alarm is ack
269      ELSE                                                    ; If alarm is unack
270      BEGIN                                                    ;be4
271 0158      IF PSKID !=BITVAL THEN
272      BEGIN                                                    ;be5
273 0160      IF BITVAL < 3 THEN FALMON = 1
274 0161      LET PSKID = BITVAL
275 0162      PDEV:DEV_TMPO = 2
276 0163      NXT = 1

```

```

276                                     END                      ;en5
277 0164                             ELSE NXT = 1
278                                 END                      ;en4
279                                 END                      ;en3
280                             ELSE                      ; Else there is no alarm
281                                 BEGIN
282 0165                             NXT = 0
283 0166                             LET PSKID = 0
284                                 END
285                                 END                      ;en2
286                             ELSE                      ; Else alarm is inhibited
287                                 BEGIN
288 0167                             NXT = 0
289 0168                             LET PSKID = 0
290                                 END
291 0169                             ANALOG = OR(ANALOG,NXT)
292                                 END                      ;en1
293 0170                             STEP PREV BY 1
294 0171                             STEP PSTAT BY 1
295 0172                             STEP PSKID BY 1
296 0173                             STEP PTYPE BY 1
297 0174                             STEP PAI BY 1
298 0175                             NEXT JLOOP
299
300                                     ;
301                                     ;
302                                     ;
303                                     ;
304 0176                             WHITE = 0
305 0177                             ASSIGN PALARM = A10
306 0178                             ASSIGN PREV = FRV15
307 0179                             ASSIGN PSTAT = INT15
308 0180                             ASSIGN PTYPE = TYPE15
309                                     ;
310 0181                             ASSIGN PAI = 'FIC-EC1-1'
311 0182                             ILOOP = 15
312 0183                             GOSUB ANLG
313                                     ;
314                                     ; Use LEAKO with string packet 78 in URW graphic for FIC-EC1-1 LO:
315                                     ;
316 0185                             IF INT15 = 3 | INT16 = 3 THEN LET LEAKO = 9
317 0187                                 ELSE IF (INT16) THEN LET LEAKO = INT16 + 4
318 0189                                 ELSE IF (INT15) THEN LET LEAKO = INT15 + 2
319 0190                                 ELSE LET LEAKO = 0
320
321                                     ;
322                                     ;
323                                     ;
324                                     ;
325 0191 RC2:                             ASSIGN PTYPE = TYPE17
326 0192                             ASSIGN PREV = FRV17
327 0193                             ASSIGN PSTAT = INT17
328 0194                             ASSIGN PPASS = FPAS17
329                                     ;
330 0195                             IF !SELECT() THEN GOTO AGAIN
                                    IF ( PTYPE > 0 ) THEN                      ; If some type of alarm

```

```

331          BEGIN
332 0197      IF AND('RC2-PIG':DEV_ALMCT,16) THEN LET PSTAT = 3 ;be1      ; If EPN is inhibited
333          IF AND('RC2-PIG':DEV_ALMCT,4) THEN BLET PSTAT = 5 ;      ; No I6AD on devices
334 0199      IF !AND('RC2-PIG':DEV_FLG,8) THEN LET PSTAT = 4 ;      ; If EPN is disabled
335 0200      IF ( PSTAT < 3 ) THEN ;      ; If not inhibited
336          BEGIN ;be2
337 0201      IF (PSTAT = 1) THEN
338          BEGIN ;be3
339 0202      IF (PTYPE != 1) THEN
340 0204      IF AND('RC2-PIG':DEV_ALMCT,32) THEN GOSUB ACK1
341 0206      IF (FTRIP4) THEN GOSUB ACK1
342          END ;en3
343 0208      IF PSTAT = 0 THEN IF ('RC2-PIG':DEV_ALMCT = 7) THEN
344          BEGIN ;be4
345 0209      CRIT
346 0210      LET PSTAT = 1
347 0211      'URW-ALARM':DEV_TMPO = 2
348 0212      FALMON = 1
349 0213      LAMP ADI,SKDKEY,2
350 0214      GOSUB ACKK
351 0215      NOCRIT
352          END ;en4
353 0216      IF (PSTAT) THEN
354          BEGIN ;be5
355 0217      IF !AND('RC2-PIG':DEV_ALMCT,7) THEN
356          BEGIN ;be6
357 0218      LET PSTAT = 0
358 0219      LET FCLEAR = 1
359 0220      GOSUB ACKK
360          END ;en6
361 0221      NXT = PSTAT
362 0223      IF (PPASS) & (PSTAT = 2) THEN NXT = 0
363 0224      LAST = OR(LAST,NXT)
364          END ;en5
365          END ;en2
366 0226      ELSE IF !AND('RC2-PIG':DEV_ALMCT,16)&AND('RC2-PIG':DEV_FLG,8) THEN LET PSTAT = 0
367 0228      IF PSTAT = 5 THEN PSTAT = 0
368          END ;en1
369      ;
370      ;
371      ;
372 0229      MARK STEP "BELOW",0,0
373 0230      FTRIP4 = 0
374 0232      IF LAST > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 31"
375 0233      CURENT = OR(LAST,WHITE)
376 0234      CURENT = OR(CURENT,ANALOG)
377 0236      IF AND(CURENT,1) THEN LMPVAR = 2
378 0238      ELSE IF AND(CURENT,2) THEN LMPVAR = 1
379 0239      ELSE LMPVAR = 0
380 0240      LAMP ADI,SKDKEY,LMPVAR
381 0241      MARK STEP "BOTTOM",0,0
382 0242      NOCRIT
383      ;
384      ;
385      ;
386      ;
387      ;
388      ;
389      ;
390      ;
391      ;
392      ;
393      ;
394      ;
395      ;
396      ;
397      ;
398      ;
399      ;
400      ;
401      ;
402      ;
403      ;
404      ;
405      ;
406      ;
407      ;
408      ;
409      ;
410      ;
411      ;
412      ;
413      ;
414      ;
415      ;
416      ;
417      ;
418      ;
419      ;
420      ;
421      ;
422      ;
423      ;
424      ;
425      ;
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432      ;
433      ;
434      ;
435      ;
436      ;
437      ;
438      ;
439      ;
440      ;
441      ;
442      ;
443      ;
444      ;
445      ;
446      ;
447      ;
448      ;
449      ;
450      ;
451      ;
452      ;
453      ;
454      ;
455      ;
456      ;
457      ;
458      ;
459      ;
460      ;
461      ;
462      ;
463      ;
464      ;
465      ;
466      ;
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473      ;
474      ;
475      ;
476      ;
477      ;
478      ;
479      ;
480      ;
481      ;
482      ;
483      ;
484      ;
485      ;
486      ;
487      ;
488      ;
489      ;
490      ;
491      ;
492      ;
493      ;
494      ;
495      ;
496      ;
497      ;
498      ;
499      ;
500      ;
501      ;
502      ;
503      ;
504      ;
505      ;
506      ;
507      ;
508      ;
509      ;
510      ;
511      ;
512      ;
513      ;
514      ;
515      ;
516      ;
517      ;
518      ;
519      ;
520      ;
521      ;
522      ;
523      ;
524      ;
525      ;
526      ;
527      ;
528      ;
529      ;
530      ;
531      ;
532      ;
533      ;
534      ;
535      ;
536      ;
537      ;
538      ;
539      ;
540      ;
541      ;
542      ;
543      ;
544      ;
545      ;
546      ;
547      ;
548      ;
549      ;
550      ;
551      ;
552      ;
553      ;
554      ;
555      ;
556      ;
557      ;
558      ;
559      ;
560      ;
561      ;
562      ;
563      ;
564      ;
565      ;
566      ;
567      ;
568      ;
569      ;
570      ;
571      ;
572      ;
573      ;
574      ;
575      ;
576      ;
577      ;
578      ;
579      ;
580      ;
581      ;
582      ;
583      ;
584      ;
585      ;
586      ;
587      ;
588      ;
589      ;
590      ;
591      ;
592      ;
593      ;
594      ;
595      ;
596      ;
597      ;
598      ;
599      ;
600      ;
601      ;
602      ;
603      ;
604      ;
605      ;
606      ;
607      ;
608      ;
609      ;
610      ;
611      ;
612      ;
613      ;
614      ;
615      ;
616      ;
617      ;
618      ;
619      ;
620      ;
621      ;
622      ;
623      ;
624      ;
625      ;
626      ;
627      ;
628      ;
629      ;
630      ;
631      ;
632      ;
633      ;
634      ;
635      ;
636      ;
637      ;
638      ;
639      ;
640      ;
641      ;
642      ;
643      ;
644      ;
645      ;
646      ;
647      ;
648      ;
649      ;
650      ;
651      ;
652      ;
653      ;
654      ;
655      ;
656      ;
657      ;
658      ;
659      ;
660      ;
661      ;
662      ;
663      ;
664      ;
665      ;
666      ;
667      ;
668      ;
669      ;
670      ;
671      ;
672      ;
673      ;
674      ;
675      ;
676      ;
677      ;
678      ;
679      ;
680      ;
681      ;
682      ;
683      ;
684      ;
685      ;
686      ;
687      ;
688      ;
689      ;
690      ;
691      ;
692      ;
693      ;
694      ;
695      ;
696      ;
697      ;
698      ;
699      ;
700      ;
701      ;
702      ;
703      ;
704      ;
705      ;
706      ;
707      ;
708      ;
709      ;
710      ;
711      ;
712      ;
713      ;
714      ;
715      ;
716      ;
717      ;
718      ;
719      ;
720      ;
721      ;
722      ;
723      ;
724      ;
725      ;
726      ;
727      ;
728      ;
729      ;
730      ;
731      ;
732      ;
733      ;
734      ;
735      ;
736      ;
737      ;
738      ;
739      ;
740      ;
741      ;
742      ;
743      ;
744      ;
745      ;
746      ;
747      ;
748      ;
749      ;
750      ;
751      ;
752      ;
753      ;
754      ;
755      ;
756      ;
757      ;
758      ;
759      ;
760      ;
761      ;
762      ;
763      ;
764      ;
765      ;
766      ;
767      ;
768      ;
769      ;
770      ;
771      ;
772      ;
773      ;
774      ;
775      ;
776      ;
777      ;
778      ;
779      ;
780      ;
781      ;
782      ;
783      ;
784      ;
785      ;
786      ;
787      ;
788      ;
789      ;
790      ;
791      ;
792      ;
793      ;
794      ;
795      ;
796      ;
797      ;
798      ;
799      ;
800      ;
801      ;
802      ;
803      ;
804      ;
805      ;
806      ;
807      ;
808      ;
809      ;
810      ;
811      ;
812      ;
813      ;
814      ;
815      ;
816      ;
817      ;
818      ;
819      ;
820      ;
821      ;
822      ;
823      ;
824      ;
825      ;
826      ;
827      ;
828      ;
829      ;
830      ;
831      ;
832      ;
833      ;
834      ;
835      ;
836      ;
837      ;
838      ;
839      ;
840      ;
841      ;
842      ;
843      ;
844      ;
845      ;
846      ;
847      ;
848      ;
849      ;
850      ;
851      ;
852      ;
853      ;
854      ;
855      ;
856      ;
857      ;
858      ;
859      ;
860      ;
861      ;
862      ;
863      ;
864      ;
865      ;
866      ;
867      ;
868      ;
869      ;
870      ;
871      ;
872      ;
873      ;
874      ;
875      ;
876      ;
877      ;
878      ;
879      ;
880      ;
881      ;
882      ;
883      ;
884      ;
885      ;
886      ;
887      ;
888      ;
889      ;
890      ;
891      ;
892      ;
893      ;
894      ;
895      ;
896      ;
897      ;
898      ;
899      ;
900      ;
901      ;
902      ;
903      ;
904      ;
905      ;
906      ;
907      ;
908      ;
909      ;
910      ;
911      ;
912      ;
913      ;
914      ;
915      ;
916      ;
917      ;
918      ;
919      ;
920      ;
921      ;
922      ;
923      ;
924      ;
925      ;
926      ;
927      ;
928      ;
929      ;
930      ;
931      ;
932      ;
933      ;
934      ;
935      ;
936      ;
937      ;
938      ;
939      ;
940      ;
941      ;
942      ;
943      ;
944      ;
945      ;
946      ;
947      ;
948      ;
949      ;
950      ;
951      ;
952      ;
953      ;
954      ;
955      ;
956      ;
957      ;
958      ;
959      ;
960      ;
961      ;
962      ;
963      ;
964      ;
965      ;
966      ;
967      ;
968      ;
969      ;
970      ;
971      ;
972      ;
973      ;
974      ;
975      ;
976      ;
977      ;
978      ;
979      ;
980      ;
981      ;
982      ;
983      ;
984      ;
985      ;
986      ;
987      ;
988      ;
989      ;
990      ;
991      ;
992      ;
993      ;
994      ;
995      ;
996      ;
997      ;
998      ;
999      ;
1000      ;

```

```

386 0243      AT0 = AND(RC2_PIG,1)
387 0244      AT1 = AND(RC2_PIG,2)
388 0245      AT2 = AND(RC2_PIG,4)
389 0246      AT3 = AND(RC2_PIG,8)
390 0247      AT4 = AND(RC2_PIG,16)
391 0248      AT5 = AND('RC2-SAMP':DEV_TMP0,32)
392 0249      AT6 = AND(RC2_PIG,64)
393 0250      AT7 = and('RC2-SAMP':DEV_TMP0,128)
394 0251      AT8 = and(rc2_pig,256)
395 0252      SA3 = AND('RC2-PIG':DEV_STAT,8) ; DI/DRAIN
396 0253      SA4 = AND('RC2-PIG':DEV_STAT,16) ; DI/VN/DR
397 0254      SA5 = AND('RC2-PIG':DEV_STAT,32) ; INTERLOCK
398 0255      SA6 = AND('RC2-PIG':DEV_STAT,64) ; LOW FLOW
399 0256      GT0 = AND(PB2GLOBAL:DEV_TMP1,2)
400
401      ;
402      ;
403      ;
404      ;
405      ;
406      ;
407      ;
408      ;
409 0258      IF AND(DEVON,2) THEN SDO = 1 ; If on (2), set device stat flag
410 0259      IF AND(DEVON,1) & AND(SD0,1) THEN ; For on-off transition
411          BEGIN
412 0260          SDO = 0
413 0261          PRINT ADI,"... FQIC-RC2 LAST READING = %f GAL",CALOUT
414 0262          CALOUT = 0.0
415          END
416      ;
417      ;
418      ;
419      ;
420      ;
421 0264      IF AND('URW-ALARM':DEV_FLG,64) THEN PUTMANL 'URW-ALARM'
422 0266      IF AND('RC2-SAMP':DEV_FLG,64) THEN PUTMANL 'RC2-SAMP'
423 0268      IF AND('RC2-PIG':DEV_FLG,64) THEN PUTMANL 'RC2-PIG'
424      ;
425 0269      GOTO AGAIN
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432 0270      ANLG: IF !SELECT() THEN GOTO AGAIN
433      ANLG: IF (P1YPE = 1) THEN ;be1 ; If some alarm
434          BEGIN
435 0271          BITVAL = PA1:AI_ALMCT ;be2 ; If not inhibited
436          IF !(AND(BITVAL,16)) THEN
437 0274          IF PSTAT = 3 THEN LET PSTAT = 0
438 0275          IF (PREV) THEN ;be3 ; If rev acting, LOL0, or L0
439          BEGIN
440 0276          IF (PSTAT = 0) THEN ; If not previously alarmed

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:36:56 24-Feb-95

Page 9

```

441          BEGIN                                ;be4
442 0278      IF (PAI:AI_MEAS <= PALARM) THEN GOSUB ALM ;en4
443          END                                  ;en4
444      ELSE
445          BEGIN                                ;be5
446 0279      IF (PSTAT = 1) THEN                    ; If previously alarmed
447          BEGIN                                ;be6
448 0281      IF (FTRIP4) THEN GOSUB ACK1            ; If ACK flag set, then ack it
449 0284      IF PTYPY != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
450          END                                  ;en6
451 0286      IF (PAI:AI_MEAS >= PALARM + PAI:AI_DEAD) THEN GOSUB CLR
452          END                                  ;en5
453      END                                      ;en3
454      ELSE                                    ; If fail-safe, HIHI, or HI
455          BEGIN                                ;be10
456 0287      IF (PSTAT = 0) THEN
457          BEGIN                                ;be11
458 0289      IF (PAI:AI_MEAS >= PALARM) THEN GOSUB ALM
459          END                                  ;en11
460      ELSE
461          BEGIN                                ;be12
462 0290      IF (PSTAT = 1) THEN
463          BEGIN                                ;be13
464 0292      IF (FTRIP4) THEN GOSUB ACK1
465 0295      IF PTYPY != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
466          END                                  ;en13
467 0297      IF (PAI:AI_MEAS <= PALARM - PAI:AI_DEAD) THEN GOSUB CLR
468          END                                  ;en12
469      END                                      ;en10
470 0298      NXT = PSTAT
471 0300      IF (PPASS) & (PSTAT = 2) THEN NXT = 0
472          END                                  ;en2
473      ELSE
474          BEGIN
475 0301      LET PSTAT = 3
476 0302      NXT = 0
477          END
478 0303      WHITE = OR(WHITE,NXT)
479          END                                  ;en1
480 0304      ELSE LET PSTAT = 0
481 0305      STEP PALARM BY 1
482 0306      STEP PSTAT BY 1
483 0307      STEP PTYPY BY 1
484 0308      STEP PREV BY 1
485 0309      STEP PPASS BY 1
486 0310      ILOOP = ILOOP + 1
487 0311      RETURN
488      ;
489      ;
490      ;
491      ;
492      ;
493 0312 CLR: LET PSTAT = 0
494 0315      IF PTYPY != 1 THEN IF AND(PDIN:DIW_ALMCT,32) THEN PDIN:DIW_ALMCT = XOR(PDIN:DIW_ALMCT,32)
495 0316      LET FCLEAR = 1

```



```

496 0317      GOSUB ACKK
497 0318      RETURN
498
499 0319      ; ALM:  LET PSTAT = 1
500 0322      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
501 0323      POEV:DEV_TMP0 = 2
502 0324      LAMP ADI,SKOKEY,2
503 0326      IF AND(PTYPE,3)THEN FALMON = 1
504 0327      CRIT
505 0328      GOSUB ACKK
506 0329      RETURN
507
508 0330      ; ACK1:  LET PSTAT = 2
509 0331      LET FACK = 1
510
511      ;
512      ; ALARM TYPER STRINGS:
513      ;
513 0333      ACKK:  IF ILOOP = 0 THEN STRNG0 = "URW SAMPLER FLOW LOW"
514 0335      IF ILOOP = 1 THEN STRNG0 = "URW SAMPLER RADIATION HI"
515 0337      IF ILOOP = 2 THEN STRNG0 = "URW SYSTEM FAIL"
516 0339      IF ILOOP = 3 THEN STRNG0 = "RC2 RADIATION BYPASS ON "
517 0341      IF ILOOP = 15 THEN STRNG0 = "E-C-1 CONDENSER URW FLOW LO"
518 0343      IF ILOOP = 16 THEN STRNG0 = "E-C-1 CONDENSER URW FLOW HI"
519 0345      IF ILOOP = 17 THEN STRNG0 = "RC1-PIG LOW FLOW"
520 0347      IF ILOOP = 24 THEN STRNG0 = "START URMATCH PGM "
521 0349      IF ILOOP = 25 THEN STRNG0 = "URMBATCH WRONG UNIT"
522 0351      IF ILOOP = 26 THEN STRNG0 = "TERMINATE URMATCH "
523
524 0352      ; AMSG:  IF PTYPE = 1 THEN
525      BEGIN
526 0354      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNG0
527 0355      IF (FACK) THEN
528      BEGIN
529 0356      PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNG0
530 0357      LET FACK = 0
531      END
532      END
533 0358      IF (FCLEAR) THEN
534      BEGIN
535 0359      PRINT ADI,"%p ALARM CLEAR",STRNG0
536 0360      LET FCLEAR = 0
537      END
538 0361      IF (FSTART) THEN
539      BEGIN
540 0362      LET FSTART = 0
541 0363      PRINT ADI,"%p",STRNG0
542      END
543 0364      RETURN
544
545 0365      ; ERRSUB: IF ERRN != ERRNUM() THEN
546      BEGIN
547 0366      TIMERON STIME0
548 0367      TIMERCLR STIME0
549 0368      LET FSN0 = 1
550 0369      LET ERRN = ERRNUM()

```

```

551 0370          LET ERRL = ERRLIN()
552 0371          PRINT ADI,"ERROR # %i AT LINE %i",errn,errl
553          END
554 0372          ERSCLR
555 0373          GOTO AGAIN
556
557          ;
558          ; TERM: LET FABORT = 0
559 0374 TERM: LET STRNGO = "URMBATCH TERMINATED  "
560 0375          LET ILOOP = STAR + 2
561 0376          LET FSTART = 1
562 0377          LET STRNGO = "TERM"
563 0378          GOSUB ACKK
564          END

```

```

Total Errors Detected      0      Total Warnings      0
Sequence Program Size = 7248 Bytes, 453 Pages
      23 Forward Reference(s)

```

VENTBATCH

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

08:26:09 25-May-95

Page 1

```

1      ; VENTBATCH.SEQ                                VERSION 3
2      ;
3      ; Function: Sequence program for Lamp function,
4      ;           message display on Graphic number 020,
5      ;           and printer messages.
6      ;
7      ; Written By : R. N. Kersey
8      ; Date      : 05-31-92
9      ; Facility  : 242-A Evaporator Bldg.
10     ; Machine   : VAX 4104-A
11     ; Op. System : VAX/VMS 5.5-2AH
12     ; D/3 Version : 6.3-2 BASELINE
13     ; Compiler   : SIC
14     ; Build command : @[320,20]VENTBATCH
15     ;
16     ; UNIT = 13, SKDKEY = 39
17     ;
18     ; Assoc. UPF: [300,1]_PROCESS.____ PROCESS.LST
19     ; Assoc. RPF: [300,1]_242A.____ 242A.LST
20     ; SDB: [300,1]PCMOA.SDB PCMOASDB.CMD
21     ;
22     ; POINTER SDB VAR. FUNCTION
23     ; -----
24     ; PSTAT INTn Current EPN alarm state is an integer tied to
25     ;           graphic flashing string of value:
26     ;           0 = previously not alarmed
27     ;           1 = previously alarmed, not ack
28     ;           2 = previously alarmed, is ack
29     ;           3 = alarm is inhibited
30     ;           4 = input is off scan
31     ;           5 = input is bad
32     ;
33     ; PDIN LEAKn Special INT variable for multiple white alarms on an EPN
34     ; PTYP TYPEn ---
35     ;           alarm type 0 = none, 1 = white, 2 = yellow,
36     ;           3 = yellow and yellow, 4 = red,
37     ;           6 = yellow and red
38     ; PREV FRVn Reverse acting (not fail safe) flag (1=set)
39     ; PSKID SKIDn Used for analog alarms
40     ; PPASS FPASn Turn lamp off when alarm is ACKnowledged
41     ; PALARM FALMn Analog alarm flag
42     ; --- FNING Unused error flag
43     ;
44     ; MODIFICATION LOG:
45     ;
46     ; 29-DEC-92 Added ammonia monitor signals, O/C Display M. Teats
47     ;           Bar code for TDIC-HC11, and comments.
48     ;
49     ; 31-DEC-92 Fixed white alarm print message. M. Teats
50     ;
51     ; 27-JAN-93 Added NTR-PRM1, INTERLOK statuses for M. Teats
52     ;           HC1-INLK1 (SB3, SB5)
53     ;
54     ; 08-JUN-93 Added comments and baselined for M. Teats
55     ;           D/3 version 3.3. CCN# 93JUN01-01.

```

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

08:26:09 25-May-95

Page 2

```

56      ;      15-DEC-94      Revised header to indicate th enew VMS      J. Berger
57      ;
58      ;
59      ;
60      ;      25-MAY-95      Changed FRV16 from 0 to 1 for AXS-NH3-1      M. Teats
61      ;      (search +mct1-). Per EE-95-00006 and
62      ;      ECN 613434.
63      ;
64      ;      END MOD LOG
65      ;
66      UNIT      VVENT
67      ;
68      COMMON PB2
69      FLAGBITS == PSPARD
70      EXC_TMP1 == 'EXC1-INLK':DEV_TMP1
71      ;
72      0000      NOP
73      0001      STAR = 35      ; Start message
74      0002      LET SKDKEY = 39      ; SKID Key
75      0003      LET UNIT = 13      ; SEQ Unit number
76      0004      ASSIGN PDEV = 'VV-ALARM' ; Alarm device
77      0005      ASSIGN PTYPE = TYPE35
78      0006      LET ADI = 0      ; Alarm destination index
79      0007      LET FSN0 = 0      ; No alarms yet
80      0008      TIMEROFF STIMEO      ; Alarm timer off
81      0009      TIMERCLR STIMEO      ; Alarm timer clear
82      0010      LET STRNG0 = " "      ; Blank 30 character string
83      0011      LET ERRN = 0      ; Set error number to zero
84      0012      LET ERRL = 0      ; Set error line to zero
85      0013      LET FTRIP4 = 0      ; FTRIP4 = 1 when audible horn is to be turned on
86      0014      GETUNIT TRIP      ; Which sequence unit is program running in
87      0015      IF TRIP != UNIT THEN      ; Abort if running in wrong seq. unit
88          BEGIN
89      0016          FSTART = 1
90      0017          ILOOP = STAR + 1
91      0018          PTYPE = 0
92      0019          GOSUB ACKK
93      0020          GOTO TERM
94      0021      END
95      ;      IF !SELECT() THEN GOTO AGAIN
96      ;
97      ;
98      ;      BEGIN INITIALIZING:
99      0021      STRNG0 = "INIT"
100     0022      INIT:      LET FABORT = 0      ; If FABORT = 1 then abort program
101     0023      LET FSTART = 0      ; Program start flag = 0
102     0024      LAMP ADI,SKDKEY,0      ; Turn off the annunciator key
103     0025      LET LSG0 = " "      ; Clear the string variables
104     0026      LET LSG1 = " "
105     0027      LET LSG2 = " "
106     0028      LET LSG3 = " "
107     0029      LET LSG4 = " "
108     0030      PDEV:DEV_TMP0 = 1      ; Set alarm device to non-alarm state
109     0031      IF AND(PDEV:DEV_ALMCT,16) THEN      ; If alarm device is inhibited, remove the inhibit
110     0032      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)

```

```

111 0033      IF AND(PDEV:DEV_ALMCT,32) THEN                ; If alarm device is ack, clear it
112 0034      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
113      ;
114      ; First, set all flags and integers:
115      ;
116      ; ILOOP 0-16 are DINS, 17-32 are AINS red and yellow, 33-34 are AIN white
117      ;
118      EPN          TYPE      INT      ILOOP
119      ;
120      PDSH-FC51     4         INT0     0
121      PDSL-FC51     2         INT1     1
122      ZS-FC5-1      2         INT2     2
123      PDSH-FC52     4         INT3     3
124      ZS-FC5-2      2         INT4     4
125      YS-EXC-1      0         INT5     5
126      YS-EXC3RL     2         INT6     6
127      TSHH-HC13     4         INT7     7
128      RXS-VVALP     2         INT8     8
129      RSH-VVALP     4         INT9     9
130      RXS-VVB/G      2         INT10    10
131      RSH-VVB/G      4         INT11    11
132      QXS-VV-1       2         INT12    12
133      (SPARE)        0         INT13    13      SKIP
134      PDSH-AS-1      2         INT14    14
135      PDSL-AS-2      2         INT15    15
136      AXS-NH3-1      2         INT16    16
137      ;
138      PDI-FC5-2       2         SKID17   17
139      PDI-FC5-1       2         SKID18   18
140      PDI-DUC1        2         SKID19   19
141      II-EXC-1        0         SKID20   20
142      PDI-FC6-1       2         SKID21   21
143      FQI-AS-5        0         SKID22   22
144      TI-HC1-1        0         SKID23   23
145      TI-HC1-2        2         SKID24   24
146      TI-FC5-1        2         SKID25   25
147      TDIC-HC11      6         SKID26   26
148      FI-AS-5         6         SKID27   27
149      PDI-FC512       6         SKID28   28
150      RI-VVALP        4         SKID29   29
151      RI-VVB/G        4         SKID30   30
152      II-AS-1         0         SKID31   31
153      AI-NH3-1        2         SKID32   32
154      ;
155      II-EXC-1        1         SKID33   33      LO      1.0
156      II-AS-1        1         SKID34   34      LO      2.0
157      ;
158 0035      ASSIGN PDIN = 'PDSH-FC51'                ; Assign pointer to first DIN
159 0036      ASSIGN PSTAT = INTO
160 0037      ASSIGN PTYPE = TYPE0
161 0038      ASSIGN PREV = FRVO
162 0039      ASSIGN PSKID = SKID0
163 0040      ASSIGN PPASS = FPAS0
164      ;
165 0041      FOR ILOOP = 0 TO 34

```



```

221      ;
222 0097      AI19 = 0.0          ; For TDIC-HC11 PID output, O/C Display Bar
223      ;
224      END INITIALIZING
225      ;
226      SEND PROGRAM START MESSAGE
227      ;
228 0098      ANALOG = 0
229 0099      ILOOP = STAR
230 0100      FSTART = 1
231 0101      GOSUB ACKK
232 0102      ON ERROR GOSUB ERRSUB
233      ;
234      ;
235      BEGIN PROGRAM LOOP :
236 0103  AGAIN:  MARK STEP "WTUNTIL",0,0
237      ;          WAITUNTIL SELECT() ; FABORT          ; Wait until micro is selected or program aborted
238      ;          IF (FABORT) THEN GOTO TERM          ; Terminate program is FABORT flag set
239 0104      MARK STEP "RNNING",0,0
240 0105      WAIT 1
241 0106      IF (FSNO) THEN          ; If alarm flag is set
242 0107      IF STIMEO > 60 THEN          ; If alarm flag is active for 1 minute
243      ;          ; Reset timer and turn off flag
244 0108      BEGIN
245 0109      TIMEROFF STIMEO
246 0110      TIMERCLR STIMEO
247 0111      FSNO = 0
248 0112      ERRM = 0
249      ;          ERRRL = 0
250      ;          END
251      ;
252      ;          If VV-ALARM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
253 0113      IF (PDEV:DEV_TMP0 = 2) & AND(PDEV:DEV_ALMCT,32) THEN
254      BEGIN
255 0114      PDEV:DEV_TMP0 = 1          ; Put VV-ALARM in ACKNWLGD state
256 0115      FTrip4 = 1          ; Set for Unit alarm acknowledgement, system to graphic
257 0116      ACK = 1          ; Set for Unit alarm acknowledgement, graphic to system
258 0117      CRIT
259 0118      LAMP ADI,SKDKEY,1          ; Turn LAMP on
260      ;          END
261      ;
262      ;          BEGIN DINs:
263      ;
264 0119      ASSIGN PDIN = 'POSH-FC51'
265 0120      ASSIGN PTYPE = TYPEO
266 0121      ASSIGN PREV = FRVO
267 0122      ASSIGN PSTAT = INTO
268 0123      ASSIGN PPASS = FPASO
269 0124      LET CURENT = 0          ; Used to set lamp status
270 0125      LET LAST = 0
271 0126      LET ANALOG = 0
272 0127      LET NXT = 0
273 0128      LMPVAR = 0
274 0129      BITVAL = 0
275      ;

```


WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

08:26:09 25-May-95

Page 6

```

276 0130 LOP:   FOR ILOOP = 0 TO 16
277       ;     IF (SELECT()) THEN GOTO AGAIN
278 0131       ;     IF ( PTYPE > 0 ) THEN                                ; If some type of alarm
279           BEGIN                                                         ;be1
280 0133       IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3                ; If EPN inhibited
281 0135       IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5                ; If EPN is bad input
282 0137       IF IAND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4                 ; If EPN is disabled
283 0138       IF ( PSTAT < 3 ) THEN                                        ; If not inhibited
284           BEGIN                                                         ;be2
285 0139       IF (PSTAT = 1) THEN
286           BEGIN                                                         ;be3
287 0140       IF (PTYPE != 1) THEN
288 0142       IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
289 0144       IF (FTRIIP4) THEN GOSUB ACK1
290           END                                                         ;en3
291 0147       IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
292 0148       IF (PSTAT) THEN
293           BEGIN                                                         ;be4
294 0150       IF (PDIN:DIN_VAL != PREV) THEN GOSUB CLR
295 0151       NXT = PSTAT
296 0153       IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
297 0154       LAST = OR(LAST,NXT)
298           END                                                         ;en4
299           ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0
300       END                                                         ;en1
301 0156       STEP PDIN BY 1
302 0157       STEP PTYP BY 1
303 0158       STEP PREV BY 1
304 0159       STEP PSTAT BY 1
305 0160       STEP PPASS BY 1
306 0161       STEP ILOOP
307 0162       NEXT ILOOP
308       ;
309       ;
310       ;
311       ;
312       ;
313 0163       ASSIGN PAI = 'PDI-FC5-2'
314 0164       ASSIGN PREV = FRV17
315 0165       ASSIGN PSTAT = INT17
316 0166       ASSIGN PSKID = SKID17
317 0167       ASSIGN PTYP = TYPE17
318 0168       FOR JLOOP = 17 TO 32
319       ;     IF (SELECT()) THEN GOTO AGAIN
320 0169       IF (PTYPE) THEN                                              ; If some alarm exists
321           BEGIN                                                         ;be1
322 0170       BITVAL = PAI:AI_ALMCT
323 0171       IF IAND(BITVAL,16) THEN                                      ; If not inhibited
324           BEGIN                                                         ;be2
325 0172       IF AND(BITVAL,15) THEN                                        ; If alarm exists and is current
326           BEGIN                                                         ;be3
327 0174       IF AND(BITVAL,32) THEN NXT = 2                              ; If alarm is ack
328       ELSE                                                            ; If alarm is unack
329           BEGIN                                                         ;be4
330 0175       IF PSKID !=BITVAL THEN

```

```

331      BEGIN
332      IF BITVAL < 3 THEN FALCON = 1
333      LET PSKID = BITVAL
334      DEV:DEV_TMO = 2
335      NXT = 1
336      END
337      ELSE NXT = 1
338      END
339      END
340      ELSE
341      BEGIN
342      NXT = 0
343      LET PSKID = 0
344      END
345      ELSE
346      BEGIN
347      NXT = 0
348      LET PSKID = 0
349      END
350      ANALOG = OR(ANALOG,NXT)
351      END
352      STEP PREV BY 1
353      STEP PREV BY 1
354      STEP PSKID BY 1
355      STEP PSKID BY 1
356      STEP PTYPE BY 1
357      NEXT JLOOP
358      END
359      END ANALOG RED AND YELLOWS
360      ;
361      ;
362      ;
363      ;
364      ;
365      ;
366      ;
367      ;
368      ;
369      ;
370      ;
371      ;
372      ;
373      ;
374      ;
375      ;
376      ;
377      ;
378      ;
379      ;
380      ;
381      ;
382      ;
383      ;
384      ;
385      ;
386      ;
387      ;
388      ;
389      ;
390      ;
391      ;
392      ;
393      ;
394      ;
395      ;
396      ;
397      ;
398      ;
399      ;
400      ;
401      ;
402      ;
403      ;
404      ;
405      ;
406      ;
407      ;
408      ;
409      ;
410      ;
411      ;
412      ;
413      ;
414      ;
415      ;
416      ;
417      ;
418      ;
419      ;
420      ;
421      ;
422      ;
423      ;
424      ;
425      ;
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432      ;
433      ;
434      ;
435      ;
436      ;
437      ;
438      ;
439      ;
440      ;
441      ;
442      ;
443      ;
444      ;
445      ;
446      ;
447      ;
448      ;
449      ;
450      ;
451      ;
452      ;
453      ;
454      ;
455      ;
456      ;
457      ;
458      ;
459      ;
460      ;
461      ;
462      ;
463      ;
464      ;
465      ;
466      ;
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473      ;
474      ;
475      ;
476      ;
477      ;
478      ;
479      ;
480      ;
481      ;
482      ;
483      ;
484      ;
485      ;
486      ;
487      ;
488      ;
489      ;
490      ;
491      ;
492      ;
493      ;
494      ;
495      ;
496      ;
497      ;
498      ;
499      ;
500      ;
501      ;
502      ;
503      ;
504      ;
505      ;
506      ;
507      ;
508      ;
509      ;
510      ;
511      ;
512      ;
513      ;
514      ;
515      ;
516      ;
517      ;
518      ;
519      ;
520      ;
521      ;
522      ;
523      ;
524      ;
525      ;
526      ;
527      ;
528      ;
529      ;
530      ;
531      ;
532      ;
533      ;
534      ;
535      ;
536      ;
537      ;
538      ;
539      ;
540      ;
541      ;
542      ;
543      ;
544      ;
545      ;
546      ;
547      ;
548      ;
549      ;
550      ;
551      ;
552      ;
553      ;
554      ;
555      ;
556      ;
557      ;
558      ;
559      ;
560      ;
561      ;
562      ;
563      ;
564      ;
565      ;
566      ;
567      ;
568      ;
569      ;
570      ;
571      ;
572      ;
573      ;
574      ;
575      ;
576      ;
577      ;
578      ;
579      ;
580      ;
581      ;
582      ;
583      ;
584      ;
585      ;
586      ;
587      ;
588      ;
589      ;
590      ;
591      ;
592      ;
593      ;
594      ;
595      ;
596      ;
597      ;
598      ;
599      ;
600      ;
601      ;
602      ;
603      ;
604      ;
605      ;
606      ;
607      ;
608      ;
609      ;
610      ;
611      ;
612      ;
613      ;
614      ;
615      ;
616      ;
617      ;
618      ;
619      ;
620      ;
621      ;
622      ;
623      ;
624      ;
625      ;
626      ;
627      ;
628      ;
629      ;
630      ;
631      ;
632      ;
633      ;
634      ;
635      ;
636      ;
637      ;
638      ;
639      ;
640      ;
641      ;
642      ;
643      ;
644      ;
645      ;
646      ;
647      ;
648      ;
649      ;
650      ;
651      ;
652      ;
653      ;
654      ;
655      ;
656      ;
657      ;
658      ;
659      ;
660      ;
661      ;
662      ;
663      ;
664      ;
665      ;
666      ;
667      ;
668      ;
669      ;
670      ;
671      ;
672      ;
673      ;
674      ;
675      ;
676      ;
677      ;
678      ;
679      ;
680      ;
681      ;
682      ;
683      ;
684      ;
685      ;
686      ;
687      ;
688      ;
689      ;
690      ;
691      ;
692      ;
693      ;
694      ;
695      ;
696      ;
697      ;
698      ;
699      ;
700      ;
701      ;
702      ;
703      ;
704      ;
705      ;
706      ;
707      ;
708      ;
709      ;
710      ;
711      ;
712      ;
713      ;
714      ;
715      ;
716      ;
717      ;
718      ;
719      ;
720      ;
721      ;
722      ;
723      ;
724      ;
725      ;
726      ;
727      ;
728      ;
729      ;
730      ;
731      ;
732      ;
733      ;
734      ;
735      ;
736      ;
737      ;
738      ;
739      ;
740      ;
741      ;
742      ;
743      ;
744      ;
745      ;
746      ;
747      ;
748      ;
749      ;
750      ;
751      ;
752      ;
753      ;
754      ;
755      ;
756      ;
757      ;
758      ;
759      ;
760      ;
761      ;
762      ;
763      ;
764      ;
765      ;
766      ;
767      ;
768      ;
769      ;
770      ;
771      ;
772      ;
773      ;
774      ;
775      ;
776      ;
777      ;
778      ;
779      ;
780      ;
781      ;
782      ;
783      ;
784      ;
785      ;
786      ;
787      ;
788      ;
789      ;
790      ;
791      ;
792      ;
793      ;
794      ;
795      ;
796      ;
797      ;
798      ;
799      ;
800      ;
801      ;
802      ;
803      ;
804      ;
805      ;
806      ;
807      ;
808      ;
809      ;
810      ;
811      ;
812      ;
813      ;
814      ;
815      ;
816      ;
817      ;
818      ;
819      ;
820      ;
821      ;
822      ;
823      ;
824      ;
825      ;
826      ;
827      ;
828      ;
829      ;
830      ;
831      ;
832      ;
833      ;
834      ;
835      ;
836      ;
837      ;
838      ;
839      ;
840      ;
841      ;
842      ;
843      ;
844      ;
845      ;
846      ;
847      ;
848      ;
849      ;
850      ;
851      ;
852      ;
853      ;
854      ;
855      ;
856      ;
857      ;
858      ;
859      ;
860      ;
861      ;
862      ;
863      ;
864      ;
865      ;
866      ;
867      ;
868      ;
869      ;
870      ;
871      ;
872      ;
873      ;
874      ;
875      ;
876      ;
877      ;
878      ;
879      ;
880      ;
881      ;
882      ;
883      ;
884      ;
885      ;
886      ;
887      ;
888      ;
889      ;
890      ;
891      ;
892      ;
893      ;
894      ;
895      ;
896      ;
897      ;
898      ;
899      ;
900      ;
901      ;
902      ;
903      ;
904      ;
905      ;
906      ;
907      ;
908      ;
909      ;
910      ;
911      ;
912      ;
913      ;
914      ;
915      ;
916      ;
917      ;
918      ;
919      ;
920      ;
921      ;
922      ;
923      ;
924      ;
925      ;
926      ;
927      ;
928      ;
929      ;
930      ;
931      ;
932      ;
933      ;
934      ;
935      ;
936      ;
937      ;
938      ;
939      ;
940      ;
941      ;
942      ;
943      ;
944      ;
945      ;
946      ;
947      ;
948      ;
949      ;
950      ;
951      ;
952      ;
953      ;
954      ;
955      ;
956      ;
957      ;
958      ;
959      ;
960      ;
961      ;
962      ;
963      ;
964      ;
965      ;
966      ;
967      ;
968      ;
969      ;
970      ;
971      ;
972      ;
973      ;
974      ;
975      ;
976      ;
977      ;
978      ;
979      ;
980      ;
981      ;
982      ;
983      ;
984      ;
985      ;
986      ;
987      ;
988      ;
989      ;
990      ;
991      ;
992      ;
993      ;
994      ;
995      ;
996      ;
997      ;
998      ;
999      ;
1000      ;

```

```

386 0210 IF LAST > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
387 0211 CURENT = OR(LAST,WHITE)
388 0212 CURENT = OR(CURENT,ANALOG)
389 0214 IF AND(CURENT,1) THEN LMPVAR = 2
390 0216 ELSE IF AND(CURENT,2) THEN LMPVAR = 1
391 0217 ELSE LMPVAR = 0
392 0218 LAMP ADI,SKDKEY,LMPVAR
393 0219 MARK STEP "BOTTOM",0,0
394 0220 NOCRIT
395 :
396 : BEGIN O/C DISPLAY BAR:
397 :
398 : Fuction: Upon loss of permissive, put the control loop in manual
399 : and set the PID block output to zero.
400 :
401 0222 IF AND('EX-C-1':DEV_STAT,2) THEN DLEAK3 = 1 ; Permissive must be Hi (1)
402 0223 IF AND('EX-C-1':DEV_STAT,2) & AND(DLEAK3,1) THEN ; Permissive lost and 1st pass
403 : BEGIN
404 0224 PUTMANL 'TDIC-HC11'
405 0225 PRINT ADI," PERMISSIVE LOST >>> TDIC-HC11 PUT IN MANUAL"
406 0226 PUTOUT 'TDIC-HC11',AI19
407 0227 PRINT ADI," PERMISSIVE LOST >>> TDIC-HC11 OUTPUT SET TO ZERO"
408 0228 DLEAK3 = 0
409 : END
410 :
411 : END O/C DISPLAY BAR
412 :
413 : BEGIN LOGIC FOR GRAPHICS:
414 :
415 0229 ASSIGN FLAGBITS = AT23
416 0230 BVAL = 128
417 0231 FOR IN1 = 23 TO 28
418 0232 FLAGBITS = AND(BVAL,EXC_TMP1)
419 0233 BVAL=BVAL(1
420 0234 STEP FLAGBITS BY 1
421 0235 NEXT IN1
422 :
423 0236 AT31 = AND('EX-C-1':DEV_TMP1,32768)
424 0237 AT0 = 'YS-EXC3RL':DIN_VAL
425 0238 AT1 = AND('EX-C-1':DEV_TMP0,1)
426 0239 AT2 = 'YS-EXC-1':DIN_VAL
427 0240 GT14 = AND(PB2GLOBAL:DEV_TMP0,16384)
428 0241 SA0 = AND('EX-C-1':DEV_STAT,16)
429 0242 SA1 = AND('EX-C-1':DEV_STAT,4)
430 0243 SB3 = AND('HC1-INLK1':DEV_STAT,8) ; HTR-PRMI
431 0244 SB5 = AND('HC1-INLK1':DEV_STAT,32) ; INTERLOK
432 :
433 : END LOGIC FOR GRAPHICS
434 :
435 : DEVICES TO LOCK IN MANUAL:
436 :
437 0246 IF AND('VV-ALARM':DEV_FLG,64) THEN PUTMANL 'VV-ALARM'
438 0248 IF AND('EX-C-1':DEV_FLG,64) THEN PUTMANL 'EX-C-1'
439 :
440 0249 GOTO AGAIN

```

```

441      ;
442      ;      END PROGRAM LOOP
443      ;
444      ;      BEGIN ANALOG WHITE SUBROUTINE:
445      ;
446      ; ANLG: IF !SELECT() THEN GOTO AGAIN
447 0250 ANLG: IF (PTYPE = 1) THEN                                ; If some alarm
448      BEGIN                                                    ;be1
449      BITVAL = PAI:AI_ALMCT
450 0252 IF !(AND(BITVAL,16)) THEN                                ; If not inhibited
451      BEGIN                                                    ;be2
452 0254 IF PSTAT = 3 THEN LET PSTAT = 0
453 0255 IF (PREV) THEN                                          ; If rev acting, LOLO, or LO
454      BEGIN                                                    ;be3
455 0256 IF (PSTAT = 0) THEN                                      ; If not previously alarmed
456      BEGIN                                                    ;be4
457 0258 IF (PAI:AI_INVL <= PALARM) THEN GOSUB ALM
458      END                                                    ;en4
459      ELSE
460      BEGIN                                                    ;be5
461 0259 IF (PSTAT = 1) THEN                                      ; If previously alarmed
462      BEGIN                                                    ;be6
463 0261 IF (FTRIP4) THEN GOSUB ACK1                                ; If ACK flag set, then ack it
464 0264 IF PTYPE != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
465      END                                                    ;en6
466 0266 IF (PAI:AI_INVL >= PALARM + PAI:AI_DEAD) THEN GOSUB CLR
467      END                                                    ;en5
468      END                                                    ;en3
469      ELSE
470      BEGIN                                                    ;be10
471 0267 IF PSTAT = 0 THEN
472      BEGIN                                                    ;be11
473 0269 IF (PAI:AI_INVL >= PALARM) THEN GOSUB ALM
474      END                                                    ;en11
475      ELSE
476      BEGIN                                                    ;be12
477 0270 IF (PSTAT = 1) THEN
478      BEGIN                                                    ;be13
479 0272 IF (FTRIP4) THEN GOSUB ACK1
480 0275 IF PTYPE != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
481      END                                                    ;en13
482 0277 IF (PAI:AI_INVL <= PALARM - PAI:AI_DEAD) THEN GOSUB CLR
483      END                                                    ;en12
484      END                                                    ;en10
485 0278 NXT = PSTAT
486 0280 IF (PPASS) & (PSTAT = 2) THEN NXT = 0
487      END                                                    ;en2
488      ELSE
489      BEGIN
490 0281 LET PSTAT = 3
491 0282 NXT = 0
492      END
493 0283 WHITE = OR(WHITE,NXT)
494      END                                                    ;en1
495 0284 ELSE LET PSTAT = 0

```

```

496 0285      STEP PALARM BY 1
497 0286      STEP PSTAT BY 1
498 0287      STEP PTYPE BY 1
499 0288      STEP PREV BY 1
500 0289      STEP PPASS BY 1
501 0290      ILOOP = ILOOP + 1
502 0291      RETURN
503          ;
504          ;
505          ;
506          ;
507          ;
508 0292      CLR:  LET PSTAT = 0
509 0295          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
510 0296          LET FCLEAR = 1
511 0297          GOSUB ACKK
512 0298          RETURN
513          ;
514 0299      ALM:  LET PSTAT = 1
515 0302          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
516 0304          IF AND(PTYPE,3)THEN FALMON = 1
517 0305          PDEV:DEV_TMPO = 2
518 0306          LAMP ADI,SKDKEY,2
519 0307          CRIT
520 0308          GOSUB ACKK
521 0309          RETURN
522          ;
523 0310      ACK1: LET PSTAT = 2
524 0311          LET FACK = 1
525          ;
526          ;
527          ;
528 0313      ACKK: IF ILOOP = 0 THEN STRNGO = "VVENT 1ST HEPA DP HI "
529 0315          IF ILOOP = 1 THEN STRNGO = "VVENT 1ST HEPA DP LO "
530 0317          IF ILOOP = 2 THEN STRNGO = "VVENT 1ST DMPR CLOSED"
531 0319          IF ILOOP = 3 THEN STRNGO = "VVENT 2ND HEPA DP HI "
532 0321          IF ILOOP = 4 THEN STRNGO = "VVENT 2ND DMPR CLOSED"
533 0323          IF ILOOP = 5 THEN STRNGO = "VVENT EXHAUSTER ON "
534 0325          IF ILOOP = 6 THEN STRNGO = "VVENT EXH FAN LOCAL "
535 0327          IF ILOOP = 7 THEN STRNGO = "VVENT HEATER OVERTEMP"
536 0329          IF ILOOP = 8 THEN STRNGO = "VVENT EXH ALPHA FAIL"
537 0331          IF ILOOP = 9 THEN STRNGO = "VVENT EXH ALPHA RD HI"
538 0333          IF ILOOP = 10 THEN STRNGO = "VVENT BETA/GAM FAIL"
539 0335          IF ILOOP = 11 THEN STRNGO = "VVENT BETA/GAMMA HI "
540 0337          IF ILOOP = 12 THEN STRNGO = "VVENT RAD SYS FAIL "
541          ;
542 0339          IF ILOOP = 13 THEN STRNGO = "VVNT AIR SPL PMP OFF"
543 0341          IF ILOOP = 14 THEN STRNGO = "P-AS-1 DELTA P HI "
544 0343          IF ILOOP = 15 THEN STRNGO = "P-AS-1 DELTA P LOW "
545 0345          IF ILOOP = 16 THEN STRNGO = "AMMONIA MONITOR FAIL"
546 0347          IF ILOOP = 33 THEN STRNGO = "II-EXC-1 AMPS LO "
547 0349          IF ILOOP = 34 THEN STRNGO = "II-AS-1 AMPS LO "
548 0351          IF ILOOP = 35 THEN STRNGO = "START VENTBATCH PGM "
549 0353          IF ILOOP = 36 THEN STRNGO = "VENTBATCH WRONG UNIT"
550          ;
551          IF ILOOP = 37 THEN STRNGO = "TERMINATE VENTBATCH "

```

; (SPARE)

```

551 0354 MSG: IF PTYPE = 1 THEN
552      BEGIN
553 0356 IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNGO
554 0357 IF (FACK) THEN
555      BEGIN
556 0358 PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
557 0359 LET FACK = 0
558      END
559      END
560 0360 IF (FCLEAR) THEN
561      BEGIN
562 0361 PRINT ADI,"%p ALARM CLEAR",STRNGO
563 0362 LET FCLEAR = 0
564      END
565 0363 IF (FSTART) THEN
566      BEGIN
567 0364 LET FSTART = 0
568 0365 PRINT ADI,"%p",STRNGO
569      END
570 0366 RETURN
571 ;
572 ; BEGIN ERROR SUBROUTINE:
573 ;
574 0367 ERRSUB: IF ERRN = ABS(ERRNUM()) THEN
575      BEGIN
576 0368 FSN0 = 1
577 ;
578 ; errors are handled differently depending on their severity:
579 ;
580 ; ERR
581 ; NO. ERROR DESCRIPTION ACTION
582 ; ---
583 ; 19 PROGRAM ALREADY IN KICK SUBROUTINE (rekick)
584 ; 32 UNDEFINED COMMUNICATIONS PACKET TYPE (ignore)
585 ; 41 PCM DB HEADER NOT LOADED (ignore)
586 ; 69 IPN NOT DEFINED (ignore)
587 ; 162 NO SUCH PROGRAM IN THIS PCM (ignore)
588 ; 165 SEQUENCE PROGRAM ALREADY RUNNING (ignore)
589 ; 170 SERIAL NUMBER MISMATCH (wait forever)
590 ; 179 ILLEGAL SUBROUTINE STACK CLEAR (ignore)
591 ; 187 BACKUP PCM IS DOWN (ignore)
592 ; 193 NO SUCH PROGRAM IN THIS PCM (ignore)
593 ; 198 ILLEGAL SUBROUTINE STACK NESTING ERROR (ignore)
594 ; 199 ILLEGAL RDI NAME (wait forever)
595 ; 200 RDI NOT LOADED IN MEMORY (wait forever)
596 ; 201 RECIPE LOAD NOT ENABLED (wait forever)
597 ; 222 BOARDS DEAD (ignore)
598 ; 226 BOARD ERROR (ie: broken skid panel) (ignore)
599 ; 245 BOTH PCMS DOWN, SENT REDUNDANT HIGHWAY ACCESS (ignore)
600 ;
601 0369 ERRN = ABS(ERRNUM()) ; get error number
602 0370 ERRL = ERRLIN() ; get error line number
603 ;
604 ; if error is not serious, clear and return to program:
605 ;

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

08:26:09 25-May-95

Page 12

```

606 0371      IF (ERRN=187) | (ERRN=32) | (ERRN=69) | (ERRN=222) THEN
607 0372          ERSRET
608 0373      IF (ERRN=41) | (ERRN=165) | (ERRN=162) | (ERRN=179) THEN
609 0374          ERSRET
610 0375      IF (ERRN=226) | (ERRN=198) | (ERRN=245) THEN
611 0376          ERSRET
612 0377      IF (ERRN=193) | (ERRN=19) THEN
613 0378          ERSRET
614      ;
615      ; else report the error, and take corrective action:
616      ;
617 0379      PRINT ADI,"ERROR NUMBER %i AT LINE %i ****",ERRN,ERRL
618      END
619 0380      ERSCLR                                ; Corrective action
620 0381      GOTO AGAIN                            ; Goto done and cleanup
621      ;
622      ;
623      ;
624      ; TERM: LET FABORT = 0
625 0382      TERM: LET STRNGO = "VENTBATCH TERMINATED  "
626 0383          LET ILOOP = STAR + 2
627 0384          LET FSTART = 1
628 0385          LET STRNGO = "TERM"
629 0386          GOSUB ACKK
630      END

```

Total Errors Detected 0 Total Warnings 0
 Sequence Program Size = 7728 Bytes, 483 Pages
 21 Forward Reference(s)

PCBATCH

VERSION 4

P C B A T C H . S E Q

Function: Sequence program for Lamp function,
message display on Graphic number 018,
and printer messages.

Written By : R. N. Kersey
Date : 05-31-92
Facility : 242A, Evaporator Bldg.
Machine : VAX 4104-A
Op. System : VAX/VMS 5.5-2AH
D/3 Version : 6.3-2 BASELINE
Compiler : SIC
Build command : @ (320,20)PCBATCH

UNIT = 14, SKOKEY = 37

Assoc. UPF: [300,1] PROCESS. _____
SDB: [300,1] 242A. _____
PCMOASDB.CMD

POINTER	SDB VAR.	FUNCTION
PSIAT	INTn	Current EPN alarm state is an integer tied to graphic flashing string of value:
	(n=0-100)	0 = previously not alarmed
		1 = previously alarmed, not ack
		2 = previously alarmed, is ack
		3 = alarm is inhibited
		4 = input is off scan
		5 = input is bad
		Special INT variable for multiple white alarms on an EPN
		alarm type 0 = none, 1 = white, 2 = yellow, 3 = yellow and yellow, 4 = red, 6 = yellow and red
		Reverse acting (not fail safe) flag (1=set)
		Used for analog alarms
		Turn lamp off when alarm is ACKnowledged
		Analog alarm flag
		Unused error flag

MODIFICATION LOG:

note: Lines starting with 'OLD' are Original Lines to be
reimplemented when operational testing is done. Lines
with ;TA are Temporary Additions to be removed when
operational testing is done. Related changes are in FEEDBATCH.
These changes are for level control of FIC-C1005 instead of
FIC-CAT1.

28-AUG-92	Added NOCRIT below MARK STEP BOTTOM	M. Teets
12-SEP-92	Changed LI-C100-2 to LI-C100-1 and changed alarm level from 0.95 to 0.98	M. Teets

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

06:47:32 16-May-95

Page 2

```

56      ;
57      ;      30-DEC-92      Fixed white alarm print message.      M. Teats
58      ;
59      ;      26-JAN-93      Added P-C100 and A-C100 device statuses, M. Teats
60      ;      (SB3,SB4,SB5,SC3,SC4,SC5)
61      ;
62      ;      01-APR-93      In INITIALIZATION fixed flags ASSIGNED to      M. Teats
63      ;      pointers. Changed FALMO and SKID0 to
64      ;      FALM5 and SKID5.
65      ;
66      ;      08-JUN-93      Added comments. CCN# 93JUN01-01.      M. Teats
67      ;
68      ;      -mct1+ 24-JUN-93      Changed "CONFIRM ON" to "SHUTDOWN" in alarm      M. Teats
69      ;      typer string section, and baselined for D/3
70      ;      version 3.3. CCN# 93JUN24-01.
71      ;
72      ;      -mct2+ 12-JAN-94      Deleted II-P-C100 white alarm per      M. Teats
73      ;      CCN# 94JAN12-01.
74      ;
75      ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
76      ;      system. Rebaselined document. Removed
77      ;      obsolete Select() statement.
78      ;
79      ;      26-MAR-95      Revised program to add the effected EPNs J. Berger
80      ;      for the Condensate Recycle Loop per ECN
81      ;      62039 and WP EE-95-00036. (JFB01) Also
82      ;      removed code for temp jumper added for
83      ;      training runs.
84      ;
85      ;      27-APR-95      Revised program to change YS-P-C100 from J. Berger
86      ;      White to Yellow alarm. See WP EE-95-00006
87      ;      and ECN 622499. (JFB02)
88      ;
89      ;
90      ;      END OF MOD LOG
91      ;
92      ;      UNIT      PC
93      ;
94      ;      COMMON PB2,EC1,IX,PB1      ;TA; CA1 for synch FIC-C1005 > LIC
95      ;      DEVON == 'RC3-SAMP':DEV_STAT      ; For FQIC-RESET device on/off check
96      ;      CALOUT == 'FQIC-RC3':CB_OTVL:3      ; For FQIC-RESET calc block outvalue
97      ;      RC3_PIG == 'RC3-PIG':DEV_TMP0
98      ;      PC100 == 'P-C100':DEV_TMP1
99      ;      PC106 == 'P-C106':DEV_TMP1      ;JFB01
100     ;      PCINLK == 'PC-INLK1':DEV_TMP0
101     ;      C100_ACK == 'PC-ALARM':DEV_TMP0      ; ALARM ACK FOR SKID KEY 37
102     ;      IX_ACK == 'IX-ALARM':DEV_TMP0      ; ALARM ACK FOR SKID KEY 36
103     ;      FIC1SP == 'FIC-C1005':PID_SPVL:2 ;TA; For synch FIC-C1005 > LIC
104     ;
105     0000      NOP
106     0001      LET STAR = 24      ; Start message
107     0002      LET UNIT# = 14      ; SEQ Unit number
108     0003      LET SKDKEY = 37      ; SKID Key
109     0004      ASSIGN PDEV = 'PC-ALARM' ; Alarm device
110     0005      ASSIGN PTYPE = TYPE24

```

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

06:47:32 16-May-95

Page 3

```

111 0006      LET ADI = 0                ; Alarm destination index
112 0007      LET FSN0 = 0              ; No alarms yet
113 0008      TIMEROFF STIMED           ; Alarm timer off
114 0009      TIMERCLK STIMED          ; Alarm timer clear
115 0010      STRNG0 = "               ; Blank 30 character string
116 0011      LET ERRN = 0              ; Set error number to zero
117 0012      LET ERRL = 0              ; Set error line to zero
118 0013      LET FTRIP4 = 0            ; FTRIP4 = 1 when audible horn is to be turned on
119 0014      GETUNIT TRIP              ; Which sequence unit is program running in
120 0015      IF TRIP != UNITT THEN     ; Abort if running in wrong seq. unit
121          BEGIN
122 0016          FSTART = 1
123 0017          ILOOP = STAR + 1
124 0018          PTYPE = 0
125 0019          GOSUB ACKK
126 0020          GOTO TERM
127          END
128          ;
129          ;
130          ;
131          ;
132          ;
133          ;
134          ;
135          ;
136          ;
137          ;
138          ;
139          ;
140          ;
141          ;
142          ;
143          ;
144          ;
145          ;
146          ;
147          ;
148          ;
149          ;
150          ;
151          ;
152          ;
153          ;
154          ;
155          ;
156          ;
157          ;
158          ;
159          ;
160          ;
161          ;
162          ;
163          ;
164 0021      STRNG0 = "INIT"
165 0022      INIT:  LET FABORT = 0      ; If FABORT = 1 then abort program

```

```

166 0023      LET FSTART = 0                ; Program start flag = 0
167 0024      LET SDO = 0                  ; For F0IC-RESET RC3-SAMP device status flag
168 0025      LAMP ADI,SKDKEY,0            ; Turn off the annunciator key
169 0026      LET LSG0 = " "               ; Clear the string variables
170 0027      LET LSG1 = " "
171 0028      LET LSG2 = " "
172 0029      LET LSG3 = " "
173 0030      LET LSG4 = " "
174 0031      C100_ACK = 1                  ; Set alarm device to non-alarm state
175 0032      IF AND('PC-ALARM':DEV_ALMCT,16) THEN          ; If alarm device is inhibited, remove the inha
176 0033      'PC-ALARM':DEV_ALMCT=XOR('PC-ALARM':DEV_ALMCT,16)
177 0034      IF AND('PC-ALARM':DEV_ALMCT,32) THEN          ; If alarm device is ack, clear it
178 0035      'PC-ALARM':DEV_ALMCT = XOR('PC-ALARM':DEV_ALMCT,32)
179      ;
180      ;
181      ;
182 0036      ASSIGN PDIN = 'YS-A-C100'          ; Assign pointer to first DIN
183 0037      ASSIGN PPASS = FPASO
184 0038      ASSIGN PREV = FRVO
185 0039      ASSIGN PTYPE = TYPED
186 0040      ASSIGN PSTAT = INTO
187      ;
188 0041      FOR ILOOP = 0 TO 5                ; For DINs JFB01
189 0042      LET PPASS = 0
190 0043      LET PREV = 0
191 0045      IF PSTAT > 5 THEN PSTAT = 0        ; Reset current status if undefined
192 0046      LET PTYPE = 2                    ; Default to yellow alarm
193 0047      IF AND(PDIN:DIN_DFLG,3) THEN      ; Check for alarms (must be a red or yellow)
194      BEGIN
195 0048      LET PTYPE = 2                    ; Default to Yellow alarm
196 0050      IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1    ; Reverse acting
197 0052      IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4    ; Set to red critical
198      END
199 0053      STEP PPASS BY 1
200 0054      STEP PREV BY 1
201 0055      STEP PTYPE BY 1
202 0056      STEP PSTAT BY 1
203 0057      NEXT ILOOP
204      ;
205 0058      ASSIGN PALARM = FALM6              ; For AIs JFB01
206 0059      ASSIGN PSKID = SKID6              ; JFB01
207 0060      FOR JLOOP = 6 TO 19              ; JFB01
208 0061      PALARM = 0
209 0062      PSKID = 0
210 0063      STEP PALARM BY 1
211 0064      STEP PSKID BY 1
212 0065      NEXT JLOOP
213      ;
214 0066      TYPE0 = 1
215 0067      TYPE3 = 2                        ; JFB02
216 0068      TYPE5 = 1                      ; JFB01
217 0069      TYPE7 = 6
218 0070      TYPE11 = 2
219 0071      TYPE15 = 2
220 0072      TYPE16 = 6

```

```

221 0073      TYPE21 = 1
222      ;:-mct2+ TYPE22 = 1      commented out
223      ;
224 0074      FRV0 = 0
225 0075      FRV1 = 0
226 0076      FRV2 = 0
227 0077      FRV3 = 0
228 0078      FRV4 = 1
229 0079      FRV5 = 0
230 0080      FRV6 = 0      ;JFB01
231 0081      FRV21 = 1
232      ;:-mct2+ FRV22 = 1      commented out
233      ;
234 0082      FPAS0 = 1
235 0083      FPAS3 = 1
236      ;
237 0084      A10 = .98
238      ;:-mct2+ A11 = 5.0      commented out
239      ;
240 0085      FLG14 = 0      ; Flag for PID auto mode, for synch PID
241 0086      FLG15 = 0      ; Flag for PID manual mode, for synch PID
242      ;      FLG13 = 0      ;TA; PID casc (1) flag, for synch FIC-C1005 > LIC
243      ;      FLG14 = 0      ;TA; PID auto (1) flag, for synch FIC-C1005 > LIC
244      ;      FLG15 = 0      ;TA; PID manual (1) flag, for synch FIC-C1005 > LIC
245      ;
246      END INITIALIZING
247      ;
248      SEND PROGRAM START MESSAGE
249      ;
250 0087      ANALOG = 0
251 0088      ILOOP = STAR
252 0089      FSTART = 1
253 0090      FPASS = 1
254 0091      GOSUB ACKK
255 0092      ON ERROR GOSUB ERRSUB
256      ;
257      BEGIN PROGRAM LOOP:
258      ;
259 0093  AGAIN:  MARK STEP "WTUNTIL",0,0
260      ;      IF SELECT() THEN FNING = 1
261      ;      ELSE FNING = 0
262      ;      WAITUNTIL SELECT() ! FABORT      ; Wait until micro selected or program aborted
263      ;      IF (FABORT) THEN GOTO TERM      ; Terminate program if abort flag set
264      ;
265      MARK STEP "RNNING",0,0
266 0095      WAIT 1
267 0096      IF (FSNO) THEN      ; If alarm flag is set
268 0097      IF (FSNO) > 60 THEN      ; If alarm flag active for 1 minute
269      BEGIN      ; Reset timer and clear flag
270 0098      TIMEROFF STIME0
271 0099      TIMERCLR STIME0
272 0100      LET FSNO = 0
273 0101      LET ERRN = 0
274 0102      LET ERRLL = 0
275      END

```

```

276      ;
277      ;
278      ;
279      0103      IF (C100_ACK = 2) & AND('PC-ALARM':DEV_ALMCT,32) THEN
280                  BEGIN
281      0104          C100_ACK = 1          ; Put PC-ALARM in ack state
282      0105          FTRIP0 = 1
283      0106          FTRIP4 = 1          ; Set for unit alarm ack
284      0107          ACK = 1          ; Set for unit alarm ack
285      0108          CRIT
286      0109          LAMP ADI,SKDKEY,1      ; Turn lamp on
287                  END
288      ;
289      ;
290      ;
291      0110      ASSIGN PDIN = 'YS-A-C100'
292      0111      ASSIGN PTYPE = TYPE0
293      0112      ASSIGN PREV = FRV0
294      0113      ASSIGN PSTAT = INTO
295      0114      ASSIGN PPASS = FPAS0
296      0115      LET LAST = 0
297      0116      LET NXT = 0
298      0117      BITVAL = 0
299      ;
300      0118      LOP:  FOR ILOOP = 0 TO 5          ;JFB01
301      ;
302      0119      IF !SELECT() THEN GOTO AGAIN
303                  IF ( PTYPE > 0 ) THEN          ; If some type of alarm
304                      BEGIN
305      0121          IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3      ;be1      ; If EPN is inhibited
306      0123          IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5      ; If EPN is bad input
307      0125          IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4      ; If EPN is disabled
308      0126          IF ( PSTAT < 3 ) THEN          ; If not inhibited
309                      BEGIN
310      0127          IF (PSTAT = 1) THEN          ;be2
311                      BEGIN
312      0128          IF (PTYPE != 1) THEN          ;be3
313      0130          IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
314      0132          IF (FTRIP4) THEN GOSUB ACK1
315                      END
316      0135          IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM      ;en3
317      0136          IF (PSTAT) THEN
318                      BEGIN
319      0138          IF (PDIN:DIN_VAL != PREV) THEN GOSUB CLR          ;be4
320      0139          NXT = PSTAT
321      0141          IF (PPASS) & (PSTAT = 2) THEN NXT = 0
322      0142          LAST = OR(LAST,NXT)
323                      END
324                      ;en4
325                      ;en2
326      0144          ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8)THEN LET PSTAT = 0
327                      ;en1
328                      END
329      0145          STEP PDIN BY 1
330      0146          STEP PTYPE BY 1
331      0147          STEP PREV BY 1
332      0148          STEP PSTAT BY 1
333      0149          STEP PPASS BY 1

```

```

331 0150      NEXT ILOOP
332      ;
333      ;
334      ;
335      ;
336      ;
337 0151  ALOG:  LET ANALOG = 0
338 0152          ASSIGN PAI = 'II-A-C100'
339 0153          ASSIGN PSKID = SKID6
340 0154          ASSIGN PALARM = FALM6
341 0155          FOR JLOOP = 6 TO 19
342      ;          IF !SELECT() THEN GOTO AGAIN
343 0156          IF !(PALARM) THEN
344              BEGIN                                     ;be1
345              BITVAL = PAI:AI_ALMCT
346 0158          IF !AND(BITVAL,16) THEN                ; If not inhibited
347              BEGIN                                     ;be2
348 0159          IF AND(BITVAL,15) THEN                  ; If alarm exists and is current
349              BEGIN                                     ;be3
350 0161          IF AND(BITVAL,32) THEN NXT = 2          ; If alarm is ack
351              ELSE                                       ; If alarm is unack
352              BEGIN                                     ;be4
353 0162          IF PSKID !=BITVAL THEN
354              BEGIN                                     ;be5
355 0164          IF BITVAL < 3 THEN FALMON = 1
356 0165          LET PSKID = BITVAL
357 0166          C100_ACK = 2
358 0167          LAMP_AD1,SKDKEY,2
359 0168          NXT = 1
360              END
361 0169          ELSE NXT = 1
362              END
363              END
364          ELSE
365              BEGIN
366 0170          NXT = 0
367 0171          LET PSKID = 0
368              END
369              END
370          ELSE
371              BEGIN
372 0172          NXT = 0
373 0173          LET PSKID = 0
374              END
375          ANALOG = OR(ANALOG,NXT)
376              END
377 0175          STEP PSKID BY 1
378 0176          STEP PAI BY 1
379 0177          STEP PALARM BY 1
380 0178          NEXT JLOOP
381      ;
382      ;
383      ;
384      ;
385      ;
386      ;
387      ;
388      ;
389      ;
390      ;
391      ;
392      ;
393      ;
394      ;
395      ;
396      ;
397      ;
398      ;
399      ;
400      ;
401      ;
402      ;
403      ;
404      ;
405      ;
406      ;
407      ;
408      ;
409      ;
410      ;
411      ;
412      ;
413      ;
414      ;
415      ;
416      ;
417      ;
418      ;
419      ;
420      ;
421      ;
422      ;
423      ;
424      ;
425      ;
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432      ;
433      ;
434      ;
435      ;
436      ;
437      ;
438      ;
439      ;
440      ;
441      ;
442      ;
443      ;
444      ;
445      ;
446      ;
447      ;
448      ;
449      ;
450      ;
451      ;
452      ;
453      ;
454      ;
455      ;
456      ;
457      ;
458      ;
459      ;
460      ;
461      ;
462      ;
463      ;
464      ;
465      ;
466      ;
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473      ;
474      ;
475      ;
476      ;
477      ;
478      ;
479      ;
480      ;
481      ;
482      ;
483      ;
484      ;
485      ;
486      ;
487      ;
488      ;
489      ;
490      ;
491      ;
492      ;
493      ;
494      ;
495      ;
496      ;
497      ;
498      ;
499      ;
500      ;
501      ;
502      ;
503      ;
504      ;
505      ;
506      ;
507      ;
508      ;
509      ;
510      ;
511      ;
512      ;
513      ;
514      ;
515      ;
516      ;
517      ;
518      ;
519      ;
520      ;
521      ;
522      ;
523      ;
524      ;
525      ;
526      ;
527      ;
528      ;
529      ;
530      ;
531      ;
532      ;
533      ;
534      ;
535      ;
536      ;
537      ;
538      ;
539      ;
540      ;
541      ;
542      ;
543      ;
544      ;
545      ;
546      ;
547      ;
548      ;
549      ;
550      ;
551      ;
552      ;
553      ;
554      ;
555      ;
556      ;
557      ;
558      ;
559      ;
560      ;
561      ;
562      ;
563      ;
564      ;
565      ;
566      ;
567      ;
568      ;
569      ;
570      ;
571      ;
572      ;
573      ;
574      ;
575      ;
576      ;
577      ;
578      ;
579      ;
580      ;
581      ;
582      ;
583      ;
584      ;
585      ;
586      ;
587      ;
588      ;
589      ;
590      ;
591      ;
592      ;
593      ;
594      ;
595      ;
596      ;
597      ;
598      ;
599      ;
600      ;
601      ;
602      ;
603      ;
604      ;
605      ;
606      ;
607      ;
608      ;
609      ;
610      ;
611      ;
612      ;
613      ;
614      ;
615      ;
616      ;
617      ;
618      ;
619      ;
620      ;
621      ;
622      ;
623      ;
624      ;
625      ;
626      ;
627      ;
628      ;
629      ;
630      ;
631      ;
632      ;
633      ;
634      ;
635      ;
636      ;
637      ;
638      ;
639      ;
640      ;
641      ;
642      ;
643      ;
644      ;
645      ;
646      ;
647      ;
648      ;
649      ;
650      ;
651      ;
652      ;
653      ;
654      ;
655      ;
656      ;
657      ;
658      ;
659      ;
660      ;
661      ;
662      ;
663      ;
664      ;
665      ;
666      ;
667      ;
668      ;
669      ;
670      ;
671      ;
672      ;
673      ;
674      ;
675      ;
676      ;
677      ;
678      ;
679      ;
680      ;
681      ;
682      ;
683      ;
684      ;
685      ;
686      ;
687      ;
688      ;
689      ;
690      ;
691      ;
692      ;
693      ;
694      ;
695      ;
696      ;
697      ;
698      ;
699      ;
700      ;
701      ;
702      ;
703      ;
704      ;
705      ;
706      ;
707      ;
708      ;
709      ;
710      ;
711      ;
712      ;
713      ;
714      ;
715      ;
716      ;
717      ;
718      ;
719      ;
720      ;
721      ;
722      ;
723      ;
724      ;
725      ;
726      ;
727      ;
728      ;
729      ;
730      ;
731      ;
732      ;
733      ;
734      ;
735      ;
736      ;
737      ;
738      ;
739      ;
740      ;
741      ;
742      ;
743      ;
744      ;
745      ;
746      ;
747      ;
748      ;
749      ;
750      ;
751      ;
752      ;
753      ;
754      ;
755      ;
756      ;
757      ;
758      ;
759      ;
760      ;
761      ;
762      ;
763      ;
764      ;
765      ;
766      ;
767      ;
768      ;
769      ;
770      ;
771      ;
772      ;
773      ;
774      ;
775      ;
776      ;
777      ;
778      ;
779      ;
780      ;
781      ;
782      ;
783      ;
784      ;
785      ;
786      ;
787      ;
788      ;
789      ;
790      ;
791      ;
792      ;
793      ;
794      ;
795      ;
796      ;
797      ;
798      ;
799      ;
800      ;
801      ;
802      ;
803      ;
804      ;
805      ;
806      ;
807      ;
808      ;
809      ;
810      ;
811      ;
812      ;
813      ;
814      ;
815      ;
816      ;
817      ;
818      ;
819      ;
820      ;
821      ;
822      ;
823      ;
824      ;
825      ;
826      ;
827      ;
828      ;
829      ;
830      ;
831      ;
832      ;
833      ;
834      ;
835      ;
836      ;
837      ;
838      ;
839      ;
840      ;
841      ;
842      ;
843      ;
844      ;
845      ;
846      ;
847      ;
848      ;
849      ;
850      ;
851      ;
852      ;
853      ;
854      ;
855      ;
856      ;
857      ;
858      ;
859      ;
860      ;
861      ;
862      ;
863      ;
864      ;
865      ;
866      ;
867      ;
868      ;
869      ;
870      ;
871      ;
872      ;
873      ;
874      ;
875      ;
876      ;
877      ;
878      ;
879      ;
880      ;
881      ;
882      ;
883      ;
884      ;
885      ;
886      ;
887      ;
888      ;
889      ;
890      ;
891      ;
892      ;
893      ;
894      ;
895      ;
896      ;
897      ;
898      ;
899      ;
900      ;
901      ;
902      ;
903      ;
904      ;
905      ;
906      ;
907      ;
908      ;
909      ;
910      ;
911      ;
912      ;
913      ;
914      ;
915      ;
916      ;
917      ;
918      ;
919      ;
920      ;
921      ;
922      ;
923      ;
924      ;
925      ;
926      ;
927      ;
928      ;
929      ;
930      ;
931      ;
932      ;
933      ;
934      ;
935      ;
936      ;
937      ;
938      ;
939      ;
940      ;
941      ;
942      ;
943      ;
944      ;
945      ;
946      ;
947      ;
948      ;
949      ;
950      ;
951      ;
952      ;
953      ;
954      ;
955      ;
956      ;
957      ;
958      ;
959      ;
960      ;
961      ;
962      ;
963      ;
964      ;
965      ;
966      ;
967      ;
968      ;
969      ;
970      ;
971      ;
972      ;
973      ;
974      ;
975      ;
976      ;
977      ;
978      ;
979      ;
980      ;
981      ;
982      ;
983      ;
984      ;
985      ;
986      ;
987      ;
988      ;
989      ;
990      ;
991      ;
992      ;
993      ;
994      ;
995      ;
996      ;
997      ;
998      ;
999      ;
1000     ;

```

```

386 0179      WHITE = 0
387 0180 WHT:   ASSIGN PALARM = A10
388 0181      ASSIGN PSTAT = INT17
389 0182      ASSIGN PTYPE = TYPE17
390 0183      ASSIGN PREV = FRV17
391 0184      ASSIGN PPASS = FPAS17
392 0185      ILOOP = 17
393 0186      ASSIGN PAI = 'LI-C100-1'
394 0187      GOSUB ANLG
395      ; -mct2+ ASSIGN PAI = 'II-P-C100' commented out
396      ; -mct2+ GOSUB ANLG                      commented out
397      ;
398      ;      END ANALOG WHITE ALARMS
399      ;
400 0188      LET FTRIP4 = 0
401 0190      IF LAST > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
402 0191      CURENT = OR(LAST,WHITE)
403 0192      CURENT = OR(CURENT,ANALOG)
404 0194      IF AND(CURENT,1) THEN LMPVAR = 2
405 0196      ELSE IF AND(CURENT,2) THEN LMPVAR = 1
406 0197      ELSE LMPVAR = 0
407 0198      LAMP ADI,SKDKEY,LMPVAR
408 0199      MARK STEP "BOTTOM",0,0
409 0200      NOCRIT
410      ;
411      ;      BEGIN LOGIC FOR GRAPHICS:
412      ;
413 0201      LET AI35 = ((('FQ1-PB1-W':AI_MEAS + 'FQ1-PB2-W':AI_MEAS) - ('FQ1-PB12W':AI_MEAS))
414      ;
415 0202      SA0 = OR('RSH-RC3-1':DIN_VAL,'RSH-CA1-1':DIN_VAL)
416 0203      SA1 = 'YS-RC3-2':DIN_VAL
417 0204      SA2 = AND('RC3-PIG':DEV_STAT,4) ; PIG FLSH
418 0205      SA3 = AND('RC3-PIG':DEV_STAT,8) ; DI/DRAIN
419 0206      SA4 = AND('RC3-PIG':DEV_STAT,32) ; INTERLOCK
420 0207      SA5 = AND('RC3-SAMP':DEV_STAT,8) ; RC3-SAMP SAMPLE
421      ;
422 0208      SB0 = AND('HV-RC3-3':DEV_STAT,4) ; RESET
423 0209      SB1 = AND('HV-RC3-3':DEV_STAT,16) ; CHGSTATE
424 0210      SB2 = AND('HV-RC3-3':DEV_STAT,32) ; INTERLOCK
425      ;
426 0211      SB0 = AND('P-C106':DEV_STAT,4) ; RESET JFB01
427 0212      SB1 = AND('P-C106':DEV_STAT,16) ; SHUTDOWN JFB01
428 0213      SB2 = AND('P-C106':DEV_STAT,32) ; INTERLOCK JFB01
429 0214      SB3 = AND('P-C100':DEV_STAT,4) ; RESET
430 0215      SB4 = AND('P-C100':DEV_STAT,16) ; SHUTDOWN
431 0216      SB5 = AND('P-C100':DEV_STAT,32) ; INTERLOCK
432      ;
433 0217      SC3 = AND('A-C100':DEV_STAT,4) ; RESET
434 0218      SC4 = AND('A-C100':DEV_STAT,16) ; SHUTDOWN
435 0219      SC5 = AND('A-C100':DEV_STAT,32) ; INTERLOCK
436      ;
437 0220      GT0 = AND(PB2GLOBAL:DEV_TMP1,2)
438 0221      AT0 = AND(RC3_PIG,1) ; TIMER ON
439 0222      AT1 = AND(RC3_PIG,2) ; TIME OUT
440 0223      AT2 = AND(RC3_PIG,4)

```



```

441 0224      AT3 = AND(RC3_PIG,8)
442 0225      AT4 = AND(RC3_PIG,16)
443 0226      AT5 = AND('RC3-SAMP':DEV_TMP0,32)
444 0227      AT6 = AND(RC3_PIG,64)
445 0228      AT7 = AND('P-C100':DEV_TMP0,2)
446 0229      AT8 = 'YS-P-C100':DIN_VAL
447 0230      AT9 = XOR(AT0,AT1)
448 0231      AT10= AND(RC3_PIG,1024)
449 0232      AT17= AND('HV-RC3-3':DEV_TMP1,2)
450 0233      AT20= AND('HV-RC3-3':DEV_CMD,2)
451 0234      AT22= AND('HV-RC3-3':DEV_TMP1,32)
452 0235      AT21= AND(AT20,AT22)
453 0236      AT11= AND('RSH-RC3-1':DIN_VAL,AT21)
454 0237      AT12= AND(PC106,512) ; JFB01 $T25
455 0238      AT23= AND(PC100,128)
456 0239      AT24= AND(PC100,256)
457 0240      AT25= AND(PC100,512)
458 0241      AT26= AND(PC100,1024)
459 0242      AT27= AND(PC100,2048)
460 0243      AT28= AND(PC100,4096)
461 0244      AT29= AND(PC100,8192)
462 0245      AT31= AND(PC100,32768)
463
464 ;
465 ;
466 ;
467 ;
468 ;
469 ;
470 0251      GT13 = AND(8192,PB2GLOBAL:DEV_TMP0)
471 0252      CT0 = AND('PC-INLK1':DEV_TMP0,1)
472 0253      CT1 = AND('PC-INLK1':DEV_TMP0,2)
473 0254      CT3 = AND('P-C106':DEV_TMP1,32) ; JFB01 $T21
474 0255      CT4 = AND('P-C106':DEV_TMP1,64) ; JFB01 $T22
475 0256      CT23 = AND('P-C100':DEV_TMP1,128)
476 0257      CT24 = AND('P-C100':DEV_TMP1,256)
477 0258      CT25 = AND('P-C100':DEV_TMP1,512)
478 0259      CT26 = AND('P-C100':DEV_TMP1,1024)
479 0260      CT27 = AND('P-C100':DEV_TMP1,2048)
480 0261      CT28 = AND('P-C100':DEV_TMP1,4096)
481
482 ;
483 ;
484 ;
485 ;
486 ;
487 ;
488 ;
489 ;
490 0263      IF AND(DEVON,2) THEN SDO = 1 ; If on (2), set device stat flag
491 0264      IF AND(DEVON,1) & AND(SDO,1) THEN ; For on-off transition
492          BEGIN
493 0265          SDO = 0
494 0266          PRINT ADI,"... FQIC-RC3 LAST READING = %f GAL",CALOUT
495 0267          CALOUT = 0.0

```

```

496      END
497      ;
498      END FQIC - RESET
499      ;
500      BEGIN SYNCH PID CASCADED SETPT:
501      ;
502      Function: Closes slave controller's PID (SPID) cascade,
503      ;           sets cascade controller's PID (CPID) to auto.
504      ;           (NOTE: the continuous database sets the CPID
505      ;           output equal to the SPID setpoint only on a
506      ;           manual to auto transition in the SPID.)
507      ;
508      0268      IF AND('FIC-C1005':CB_FLG:2,64) THEN          ; If in auto
509      ;           BEGIN                                     ;be1
510      0269      IF (FLG14=0) & (FLG15=1) THEN                ; If previously manual
511      ;           BEGIN                                     ;be2
512      0270      CLOSCASC 'FIC-C1005'
513      0272      IF !AND('WFC-C100':CB_FLG:2,64) THEN PUTAUTO 'WFC-C100'
514      0273      LET FLG14 = 1
515      0274      LET FLG15 = 0
516      ;           END                                     ;en2
517      ;           ELSE                                     ;en1
518      ;           BEGIN                                     ;be4
519      0275      IF ((FLG14=1)&(FLG15=0)) ! ((FLG14=0)&(FLG15=0)) THEN ; If prev auto or init
520      ;           BEGIN                                     ;be5
521      0276      LET FLG14 = 0
522      0277      LET FLG15 = 1
523      ;           END                                     ;en5
524      ;           END                                     ;en4
525      ;           IF AND('FIC-C1005':CB_FLG:2,64) THEN    ;TA; If in auto
526      ;           BEGIN                                     ;TA;be1
527      ;           IF (FLG14=0) & (FLG15=0) THEN            ;TA; If initializing
528      ;           BEGIN                                     ;TA;be2
529      ;           IF AND('FIC-C1005':PID_SPVS:2,64) THEN LET FLG13 = 1 ;TA; If casc closed
530      ;           LET FLG14 = 1                             ;TA
531      ;           LET FLG15 = 0                             ;TA
532      ;           END                                     ;TA;en2
533      ;           IF AND('FIC-C1005':PID_SPVS:2,64) & (FLG13=0) THEN ;TA; If clos casc + prev auto-open
534      ;           BEGIN                                     ;TA;be3
535      ;           IF !AND('LIC-CA1-1':CB_FLG:1,64) THEN ;TA; If LIC-CA1-1 man
536      ;           BEGIN                                     ;TA;be3.1
537      ;           PUTOUT 'LIC-CA1-1',FIC1SP                ;TA
538      ;           PUTAUTO 'LIC-CA1-1'                      ;TA
539      ;           END                                     ;TA;en3.1
540      ;           IF !AND('LIC-CA1-2':CB_FLG:1,64) THEN ;TA; If LIC-CA1-2 man
541      ;           BEGIN                                     ;TA;be3.2
542      ;           PUTOUT 'LIC-CA1-2',FIC1SP                ;TA
543      ;           PUTAUTO 'LIC-CA1-2'                      ;TA
544      ;           END                                     ;TA;en3.2
545      ;           LET FLG13 = 1                             ;TA;en3
546      ;           END                                     ;TA;en3
547      ;           IF !AND('FIC-C1005':PID_SPVS:2,64) & (FLG13=1) THEN ;TA; If open casc + prev auto-clos
548      ;           BEGIN                                     ;TA;be4
549      ;           IF AND('LIC-CA1-1':CB_FLG:1,64) THEN PUTMANL 'LIC-CA1-1' ;TA
550      ;           ;

```

```

551      :      IF AND('LIC-CA1-2':CB_FLG:1,64) THEN PUTMANL 'LIC-CA1-2' ;TA
552      :      LET FLG13 = 0
553      :      END ;TA;en4
554      :      IF (FLG14=0) & (FLG15=1) THEN ;TA; If previously manual
555      :      BEGIN ;TA;be5
556      :      CRIT ;TA
557      :      PUTOUT 'LIC-CA1-1',FIC1SP ;TA
558      :      PUTOUT 'LIC-CA1-2',FIC1SP ;TA
559      :      CLOSCASC 'FIC-C1005' ;TA
560      :      IF !AND('LIC-CA1-1':CB_FLG:1,64) THEN PUTAUTO 'LIC-CA1-1' ;TA
561      :      IF !AND('LIC-CA1-2':CB_FLG:1,64) THEN PUTAUTO 'LIC-CA1-2' ;TA
562      :      LET FLG13 = 1 ;TA
563      :      LET FLG14 = 1 ;TA
564      :      LET FLG15 = 0 ;TA
565      :      NOCRIT ;TA
566      :      END ;TA;en5
567      :      ELSE ;TA;en1
568      :      BEGIN ;TA;be7
569      :      IF (FLG14=1) & (FLG15=0) THEN ;TA; If previously auto
570      :      BEGIN ;TA;be8
571      :      IF AND('LIC-CA1-1':CB_FLG:1,64) THEN PUTMANL 'LIC-CA1-1' ;TA
572      :      IF AND('LIC-CA1-2':CB_FLG:1,64) THEN PUTMANL 'LIC-CA1-2' ;TA
573      :      LET FLG13 = 0 ;TA
574      :      END ;TA;en8
575      :      IF ((FLG14=1)&(FLG15=0)) & ((FLG14=0)&(FLG15=0)) THEN ;TA; If prev auto or init
576      :      BEGIN ;TA;be9
577      :      LET FLG14 = 0 ;TA
578      :      LET FLG15 = 1 ;TA
579      :      END ;TA;en9
580      :      END ;TA;en7
581      :      END
582      :      END SYNCH PID CASCADED SETPT
583      :
584      :
585      0278      GOTO AGAIN
586      :
587      :      END PROGRAM LOOP
588      :
589      :      BEGIN ANALOG WHITE SUBROUTINE:
590      :
591      :      ; ANLG: IF ISELECT() THEN GOTO AGAIN
592      0279      ANLG: IF (PTYPE = 1) THEN ; If some alarm
593      :      BEGIN ;be1
594      :      BITVAL = PAI:AI_ALMCT
595      :      IF !(AND(BITVAL,16)) THEN ; If not inhibited
596      :      BEGIN ;be2
597      :      IF PSTAT = 3 THEN LET PSTAT = 0
598      :      IF (PREV) THEN ; If rev acting, LOLO, or LO
599      :      BEGIN ;be3
600      :      IF (PSTAT = 0) THEN ; If previously not alarmed
601      :      BEGIN ;be4
602      :      IF (PAI:AI_INVL <= PALARM) THEN GOSUB ALM ;en4
603      :      END
604      :      ELSE
605      :      BEGIN ;be5

```

```

606 0288          IF (PSTAT = 1) THEN                ; If previously alarmed
607              BEGIN                                ;be6
608              IF (FTRIP4) THEN GOSUB ACK1           ; If ACK flag set, then ack it
609 0293          IF PTYPE !=1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
610              END                                  ;en6
611 0295          IF (PAI:AI_INVL >= PALARM + PAI:AI_DEAD) THEN GOSUB CLR
612              END                                  ;en5
613              ELSE                                  ;en3
614                  BEGIN                              ;be10
615                  IF (PSTAT = 0) THEN
616 0296          BEGIN                                ;be11
617                  IF (PAI:AI_INVL >= PALARM) THEN GOSUB ALM
618 0298          END                                  ;en11
619              ELSE
620                  BEGIN                              ;be12
621 0299          IF (PSTAT = 1) THEN
622                  BEGIN                                ;be13
623                  IF (FTRIP4) THEN GOSUB ACK1
624 0301          IF PTYPE != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
625 0304          END                                  ;en13
626                  IF (PAI:AI_INVL <= PALARM - PAI:AI_DEAD) THEN GOSUB CLR
627 0306          END                                  ;en12
628              END                                  ;en10
629              NXT = PSTAT
630 0307          IF (PPASS) & (PSTAT = 2) THEN NXT = 0
631 0309          WHITE = OR(WHITE,NXT)
632 0310          END                                  ;en2
633          IF AND(BITVAL,16) THEN LET PSTAT = 3
634 0312          END                                  ;en1
635          ELSE LET PSTAT = 0
636 0313          STEP PALARM BY 1
637 0314          STEP PSTAT BY 1
638 0315          STEP PTYPE BY 1
639 0316          STEP PREV BY 1
640 0317          STEP PPASS BY 1
641 0318          ILOOP = ILOOP + 1
642 0319          RETURN
643 0320
644          ;
645          ;
646          ;
647          ;
648          ;
649          ;
650 0321  CLR:  LET PSTAT = 0
651 0324          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
652 0325          LET FCLEAR = 1
653 0326          GOSUB ACKK
654 0327          RETURN
655          ;
656 0328  ALM:  LET PSTAT = 1
657 0331          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
658 0332          C100 ACK = 2
659 0334          IF AND(PTYPE,3) THEN FALMON = 1
660 0335          LAMP ADI,SKDKEY,2
661 0336          CRIT

```

```

661 0337      GOSUB ACKK
662 0338      RETURN
663      ;
664 0339  ACK1:  LET PSTAT = 2
665 0340      LET FACK = 1
666      ;
667      ;
668      ;
669 0342  ACKK:  IF ILOOP = 0 THEN STRNGO = "TK-C100 AGITATOR SHUTDOWN"      ; -mct1+
670 0344      IF ILOOP = 1 THEN STRNGO = "TK-C100 OVERFLOW TRAP LO"
671 0346      IF ILOOP = 2 THEN STRNGO = "TK-C100 VVENT SEAL POT LVL LO"
672 0348      IF ILOOP = 3 THEN STRNGO = "P-C100 PUMP SHUTDOWN"      ; -mct1+
673 0350      IF ILOOP = 4 THEN STRNGO = "P-C100 RAD INTERLOCK BYPASS ON"
674 0352      IF ILOOP = 5 THEN STRNGO = "P-C106 PUMP SHUTDOWN"      ; JFB01
675 0354      IF ILOOP = 21 THEN STRNGO = "LI-C100-1 TK-C100 INTERFACE LO"      ; JFB01
676      ; -mct2+ IF ILOOP = 22 THEN STRNGO = "II-P-C100: PUMP C100 AMPS HI"      ; commented out
677 0356      IF ILOOP = 24 THEN STRNGO = "START PCBATCH PGM "
678 0358      IF ILOOP = 25 THEN STRNGO = "PCBATCH WRONG UNIT"
679 0360      IF ILOOP = 26 THEN STRNGO = "TERMINATE PCBATCH"
680      ;
681 0361  AMSG:  IF PTYPE = 1 THEN
682      BEGIN
683 0363      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNGO
684 0364      IF (FACK) THEN
685      BEGIN
686 0365      PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
687 0366      LET FACK = 0
688      END
689      END
690 0367      IF (FCLEAR) THEN
691      BEGIN
692 0368      PRINT ADI,"%p ALARM CLEAR",STRNGO
693 0369      LET FCLEAR = 0
694      END
695 0370      IF (FSTART) THEN
696      BEGIN
697 0371      LET FSTART = 0
698 0372      PRINT ADI,"%p",STRNGO
699      END
700 0373      RETURN
701      ;
702 0374  ERRSUB: IF ERRN I= ERRNUM() THEN
703      BEGIN
704 0375      TIMEROM STIMED
705 0376      TIMERCLR STIMED
706 0377      LET FSN0 = 1
707 0378      LET ERRN = ERRNUM()
708 0379      LET ERRL = ERRLIN()
709 0380      PRINT ADI,"ERROR # %i AT LINE %i",errn,errl
710      END
711 0381      ERSCLR
712 0382      GOTO AGAIN
713      ;
714      ; TERM: LET FABORT = 0
715 0383  TERM:  LET STRNGO = "PCBATCH TERMINATED "
```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

06:47:32 16-May-95

Page 14

```
716 0384      LET ILOOP = STAR + 2
717 0385      LET FSTART = 1
718 0386      LET STRNGO = "TERM"
719 0387      GOSUB ACKK
720          END
```

```
Total Errors Detected      0      Total Warnings      0
Sequence Program Size = 7760 Bytes, 485 Pages
      19 Forward Reference(s)
```

IXBATCH

```

1      ; I X B A T C H . S E Q                                VERSION 3
2      ;
3      ; Function:      Sequence program for Lamp function,
4      ;                message display on Graphic number 044,
5      ;                and printer messages.
6      ;
7      ; Written By      : R. N. Kersey
8      ; Date            : 05-31-92
9      ; Facility        : 242-A Evaporator Bldg.
10     ; Machine         : VAX 4104-A
11     ; Op. System      : VAX/VMS 5.5-2AH
12     ; D/3 Version     : 6.3-2 BASELINE
13     ; Compiler        : SIC
14     ; Build command   : @[320,20]IXBATC
15     ;
16     ; UNIT = 15, SKDKEY = 36
17     ;
18     ; Assoc. UPF:      [300,1]_PROCESS.____ PROCESS.LST
19     ; Assoc. RPF:      [300,1]_242A.____ 242A.LST
20     ; SDB:            [300,1]PCMOA.SDB PCMOASDB.CMD
21     ;
22     ; POINTERR SDB VAR. FUNCTION
23     ; -----
24     ; PSTAT INTn Current EPN alarm state is an integer tied to
25     ;                graphic flashing string of value:
26     ;                0 = previously not alarmed
27     ;                1 = previously alarmed, not ack
28     ;                2 = previously alarmed, is ack
29     ;                3 = alarm is inhibited
30     ;                4 = input is off scan
31     ;                5 = input is bad
32     ;
33     ; POIN LEAKn Special INT variable for multiple white alarms on an EPN
34     ; PTYPE 'EPN' ---
35     ;                alarm type 0 = none, 1 = white, 2 = yellow,
36     ;                3 = yellow and yellow, 4 = red,
37     ;                6 = yellow and red
38     ; PREV FRVn Reverse acting (not fail safe) flag (1=set)
39     ; PSKID SKIDn Used for analog alarms
40     ; PALARM FALMn Analog alarm flag
41     ; --- FNING Unused error flag
42     ;
43     ; MODIFICATION LOG:
44     ;
45     ; 30-DEC-92 Fixed white alarm print message. M. Teats
46     ;
47     ; 08-JUN-93 Added comments. CCN# 93JUN01-01. M. Teats
48     ;
49     ; 21-JUN-93 Removed 3 comment lines. CCN# 93JUN21-02. M. Teats
50     ;
51     ; -mct1+ 29-JUN-93 Remove IBAD on RC3-PIG, set FRV33 to 1, M. Teats
52     ;                and baselined for D/3 version 3.3.
53     ;                CCN# 93JUN29-01.
54     ;
55     ; 15-DEC-94 Revised header to indicate the new VMS J. Berger

```



```

56      ;      system. Rebaselined document. Removed
57      ;      obsolete Select() statements.
58      ;
59      ;      02-MAY-95      Added variables AT27, AT28, AT29 to use      J. Berger
60      ;      for interlock graphics. See WP EE-95-00006
61      ;      and ECN 622498.
62      ;
63      ;      01-APR-96      Changed the RC3-PIG DEV code to remove the      J. Berger
64      ;      the section where the continuous control
65      ;      database was getting changed. See MCS-96-013
66      ;      and ECN 627501.
67      ;
68      ;
69      ;      END MOD LOG
70      ;
71      UNIT      IX
72      ;
73      COMMON PB2,EC1
74      RC3_PIG == 'RC3-PIG':DEV_TMPO
75      IX_ACK == 'IX-ALARM':DEV_TMPO
76      RC3 == WHITE
77      ;
78      0000      GTO = AND(PB2GLOBAL:DEV_TMP1,2)
79      0001      NOP
80      0002      LET STAR = 24      ; Start message
81      0003      LET SKDKEY = 36      ; SKID Key
82      0004      LET UNITT = 15      ; SEQ Unit number
83      0005      ASSIGN PDEV = 'IX-ALARM' ; Alarm device
84      0006      ASSIGN PTYPE = TYPE24
85      0007      LET ADI = 0      ; Alarm destination index
86      0008      LET FSN0 = 0      ; No alarm yet
87      0009      TIMEROFF STIME0      ; Alarm timer off
88      0010      TIMERCLR STIME0      ; Alarm timer clear
89      0011      STRNG0 = " "      ; Blank 30 character string
90      0012      LET ERRN = 0      ; Set error number to zero
91      0013      LET ERRL = 0      ; Set error line to zero
92      0014      LET FTRIP4 = 0      ; FTRIP4 = 1 when audible horn is to be turned on
93      0015      GETUNIT TRIP      ; Which sequence unit is program running in
94      0016      IF TRIP != UNITT THEN      ; Abort if running in wrong seq. unit
95      ;
96      0017      BEGIN
97      0018          FSTART = 1
98      0019          ILOOP = STAR + 1
99      0020          PTYPE = 0
100     0021      GOSUB ACKK
101     ;      GOTO TERM
102     ;      END
103     ;
104     ;      IF !SELECT() THEN GOTO AGAIN
105     ;
106     ;      BEGIN INITIALIZING:
107     ;
108     0022      STRNG0 = "INIT"
109     0023      INIT:      LET FABORT = 0      ; If FABORT = 1 then abort program
110     0024      LET FSTART = 0      ; Program start flag = 0
111     0025      LET SDO = 0      ; For FQIC-RESET RC3-SAMP device status flag
112     0026      LAMP ADI,SKDKEY,0      ; Turn off the annunciator key

```

```

111 0027 LET LSGD = " " ; Clear the string variables
112 0028 LET LSG1 = " "
113 0029 LET LSG2 = " "
114 0030 LET LSG3 = " "
115 0031 LET LSG4 = " "
116 0032 PDEV:DEV_IMPO = 1
117 0033 IF ANDV(PDEV:DEV_ALMCT,16) THEN ; Set alarm device to non-alarm state
118 0034 PDEV:DEV_ALMCT = 16 XOR PDEV:DEV_ALMCT ; If alarm device is inhibited, remove the inha
119 0035 IF ANDV(PDEV:DEV_ALMCT,32) THEN ; If alarm device is ack, clear it
120 0036 PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
121
122 ; First, set all flags and integers:
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165

```

DEVS:

IX-ALARM	
HW-RC3-3	
RC3-SAMP	
RC3-PIG	
HW-2-34	
HW-2-37	

DIMS:

ZS-IDX-2	INT0	W
PSH-ILF-1	INT1	W
YS-RC3-6	INT2	Y
RAS-RC3-1	INT3	Y
RSH-RC3-1	INT4	R
PSH-RC3-1	INT5	R
ZS-RC3-4	INT6	W
ZS-RC3-5	INT7	W
ZS-IDX1NM		
ZS-IDX-1B		
ZS-IDX-3B		
ZS-IDX3NM		
ZS-RC3-3D		
ZS-RC33NM		

Als:

PI-IDX-13	8
PI-IDX-14	9
PI-IDX-15	10
PI-IDX-16	11
P01-IDX-2	RY 12
P01-IDX-4	13
P01-IDX-3	RY 14
PI-IDX-12	15
PI-IDX-17	16
P01-IDX-5	RY 17
P01-IDX-1	RY 18
PI-IDX-1B	19
PI-IDX-5	20
PI-IDX-6	21
FI-RC3-1	22
P01-ILS-1	RY 23
F01C-RC3	24
F01-RC3NM	25
F01-RC3-D	26

```

166      ;      KY-RC3-1      -      27
167      ;      RI-RC3-1      R      28
168      ;
169 0037      ASSIGN PDIN = 'ZS-IDX-2'
170 0038      ASSIGN PSTAT = INTO
171 0039      ASSIGN PTYPE = TYPE0
172 0040      ASSIGN PREV = FRV0
173 0041      ASSIGN PSKID = SKID0
174 0042      ASSIGN PPASS = FPAS0
175
176      ;
176 0043      FOR ILOOP = 0 TO 7      ; For DINs
177 0044      PPASS = 0
178 0045      PREV = 0
179 0046      PSTAT = 0
180 0047      PTYPE = 0
181 0048      IF AND(PDIN:DIN_DFLG,3) THEN      ; Check for alarms (must be red or yellow)
182      BEGIN
183 0049      PTYPE = 2      ; Default to yellow alarm
184 0051      IF AND(PDIN:DIN_DFLG,1) THEN PREV = 1      ; Reverse acting
185 0053      IF AND(PDIN:DIN_DFLG,4) THEN PTYPE = 4      ; Set to red critical
186      END
187 0054      STEP PPASS BY 1
188 0055      STEP PREV BY 1
189 0056      STEP PSTAT BY 1
190 0057      STEP PTYPE BY 1
191 0058      STEP PSKID BY 1
192 0059      NEXT ILOOP
193
194      ;
194 0060      ASSIGN PALARM = FALM8
195 0061      ASSIGN PSKID = SKID8
196 0062      FOR JLOOP = 0 TO 32      ; For AIs
197 0063      PALARM = 0
198 0064      PSKID = 0
199 0065      PTYPE = 0
200 0066      STEP PALARM BY 1
201 0067      STEP PSKID BY 1
202 0068      STEP PTYPE BY 1
203 0069      NEXT JLOOP
204
205      ;
205 0070      TYPE0 = 1
206 0071      TYPE1 = 1
207 0072      TYPE2 = 2
208 0073      TYPE3 = 2
209 0074      TYPE4 = 4
210 0075      TYPE6 = 1
211 0076      TYPE7 = 1
212 0077      TYPE12 = 6
213 0078      TYPE14 = 6
214 0079      TYPE17 = 6
215 0080      TYPE18 = 6
216 0081      TYPE23 = 6
217 0082      TYPE28 = 4
218 0083      TYPE33 = 2
219
220 0084      ;      FRV0 = 1

```

```

221 0085      FRV2 = 1
222 0086      FRV6 = 0
223 0087      FRV7 = 0
224 0088      FRV33 = 1      ;:-mct1+
225      ;
226      ;      END      INITIALIZING
227      ;
228      ;      SEND      PROGRAM      START      MESSAGE
229      ;
230 0089      ANALOG = 0
231 0090      ILOOP = STAR
232 0091      FSTART = 1
233 0092      FPASS = 1
234 0093      GOSUB ACKK
235 0094      ON ERROR GOSUB ERRSUB
236      ;
237      ;      BEGIN      PROGRAM      LOOP:
238      ;
239 0095 AGAIN: MARK STEP "WTUNTL",0,0
240      ;      IF SELECT() THEN FWING = 1
241      ;      ELSE FWING = 0
242      ;      WAITUNTIL SELECT() ! FABORT      ; Wait until micro is selected or program aborted
243      ;      IF (FABORT) THEN GOTO TERM      ; Terminate program if abort flag set
244      ;      MARK STEP "RNNING",0,0
245 0096      WAIT 1
246 0097      FTRIP4 = 0
247 0098      IF (FSNO) THEN      ; If alarm flag is set
248 0100      IF STIME0 > 60 THEN      ; If alarm flag active for 1 minute
249      ;      BEGIN      ; Reset timer and turn off flag
250 0101      TIMEROFF STIME0
251 0102      TIMERCLR STIME0
252 0103      LET FSNO = 0
253 0104      LET ERRN = 0
254 0105      LET ERRL = 0
255      END
256      ;
257      ;      If IX-ALARM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
258      ;
259 0106      IF AND('IX-ALARM':DEV_ALMCT,32) & IX_ACK = 2 THEN
260      BEGIN
261 0107      IX_ACK = 1      ; Put IX-ALARM in ack state
262 0108      FTRIP4 = 1      ; Set for unit alarm ack
263 0109      ACK = 1      ; Set for unit alarm ack
264      ;
265 0110      CRIT
266      LAMP ADI,SKDKEY,1      ; Turn lamp on
267      END
268      ;
269 0111      LET LAST = 0
270 0112      LET NXT = 0
271 0113      BITVAL = 0
272      ;
273      ;      BEGIN DINs:
274 0114 LOP:  ASSIGN PDIN = 'ZS-IDX-2'
275 0115      ASSIGN PTYPE = TYPE0

```

```

276 0116      ASSIGN PREV = FRV0
277 0117      ASSIGN PSTAT = INTO
278 0118      ASSIGN PPASS = FPASS
279 0119      FOR ILOOP = 0 TO 7
280          ; IF ISELECT() THEN GOTO AGAIN
281 0120      IF ( PTYPE > 0 ) THEN                                ; If some type of alarm
282          BEGIN                                                ;be1
283 0122          IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3      ; If EPN is inhibited
284 0124          IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5      ; If EPN is bad input
285 0126          IF IAND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4      ; If EPN is disabled
286 0127          IF ( PSTAT < 3 ) THEN                            ; If not inhibited
287          BEGIN                                                ;be2
288 0128          IF (PSTAT = 1) THEN
289          BEGIN                                                ;be3
290 0129          IF (PTYPE != 1) THEN
291 0131          IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
292 0133          IF (FTRIP4) THEN GOSUB ACK1
293          END                                                  ;en3
294 0136          IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
295 0137          IF (PSTAT) THEN
296          BEGIN                                                ;be4
297 0139          IF (PDIN:DIN_VAL != PREV) THEN GOSUB CLR
298 0140          NXT = PSTAT
299 0142          IF (PPASS) & (PSTAT = 2) THEN NXT = 0
300 0143          LAST = OR(LAST,NXT)
301          END                                                  ;en4
302          END                                                  ;en2
303 0145          ELSE IF IAND(PDIN:DIN_ALMCT,16)&IAND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0
304          END                                                  ;en1
305 0146          STEP PDIN BY 1
306 0147          STEP PTYPE BY 1
307 0148          STEP PREV BY 1
308 0149          STEP PSTAT BY 1
309 0150          STEP PPASS BY 1
310 0151          NEXT ILOOP
311          ;
312          ;
313          ;
314 0152          LET STRNG0 = "RUN"
315          ;
316          ;
317          ;
318 0153      ALOG: LET ANALOG = 0
319 0154          ASSIGN PAI = 'PI-IDX-13'
320 0155          ASSIGN PSKID = SKID8
321 0156          ASSIGN PALARM = FALM8
322 0157          FOR JLOOP = 8 TO 28
323          ; IF ISELECT() THEN GOTO AGAIN
324 0158          IF I(PALARM) THEN
325          BEGIN                                                ;be1
326 0159          BITVAL = PAI:AI_ALMCT
327 0160          IF IAND(BITVAL,16) THEN                            ; If not inhibited
328          BEGIN                                                ;be2
329 0161          IF AND(BITVAL,15) THEN                            ; If alarm exists and is current
330          BEGIN                                                ;be3

```

```

331 0163      IF AND(BITVAL,32) THEN NXT = 2      ; If alarm is ack
332      ELSE      ; If alarm is unack
333      BEGIN      ;be4
334 0164      IF (PSKID !=BITVAL) THEN
335      BEGIN      ;be5
336 0166      IF BITVAL < 3 THEN FALMON = 1
337 0167      LET PSKID = BITVAL
338 0168      IX ACK = 2
339 0169      LAMP ADI,SKDKEY,2
340 0170      NXT = 1
341      END      ;en5
342 0171      ELSE NXT = 1
343      END      ;en4
344      END      ;en3
345      ELSE      ; Else there is no alarm
346      BEGIN
347 0172      NXT = 0
348 0173      LET PSKID = 0
349      END
350      END      ;en2
351      ELSE      ; Else alarm is inhibited
352      BEGIN
353 0174      NXT = 0
354 0175      LET PSKID = 0
355      END
356 0176      ANALOG = OR(ANALOG,NXT)
357      END      ;en1
358 0177      STEP PSKID BY 1
359 0178      STEP PAI BY 1
360 0179      STEP PALARM BY 1
361 0180      NEXT JLOOP
362      ;
363      ;
364      ;
365      ;
366      ;
367 0181      RC3 = 0
368 0182      ASSIGN PTYPE = TYPE33
369 0183      ASSIGN PREV = FRV33
370 0184      ASSIGN PSTAT = INT33
371 0185      ASSIGN PPASS = FPAS33
372      ;
373 0186      IF !SELECT() THEN GOTO AGAIN
374      IF ( PTYPE > 0 ) THEN      ; If some type of alarm
375      BEGIN      ;be1
376      IF AND('RC3-PIG':DEV_ALMCT,16) THEN LET PSTAT = 3      ; If EPN is inhibited
377      IF AND('RC3-PIG':DEV_ALMCT,4) THEN BLET PSTAT = 5      ; No IBAD on devices
378 0190      IF !AND('RC3-PIG':DEV_FLG,8) THEN LET PSTAT = 4      ; If EPN is disabled
379 0191      IF ( PSTAT < 3 ) THEN      ; If not inhibited
380      BEGIN      ;be2
381      IF PSTAT = 1 THEN
382      BEGIN      ;be3
383 0194      IF AND('RC3-PIG':DEV_ALMCT,32) THEN GOSUB ACK1
384 0196      IF (FTRIP4) THEN GOSUB ACK1
385      END
386      IF !PSTAT THEN IF AND('RC3-PIG':DEV_STAT,64) THEN

```

```

386          BEGIN                                     ;be4
387          CRIT
388          0199      LET PSTAT = 1
389          ;        IF AND('RC3-PIG':DEV_ALMCT,32) THEN 'RC3-PIG':DEV_ALMCT = XOR('RC3-PIG':DEV_ALMCT,32)
390          0200      IX_ACK = 2
391          0201      FALMON = 1
392          0202      LAMP ADI,SKDKEY,2
393          0203      GOSUB ACKK
394          ;        NOCRIT
395          ;        END                                     ;en4
396          0204      IF PSTAT THEN
397          ;        BEGIN                                     ;be5
398          0205      IF !AND('RC3-PIG':DEV_STAT,64) THEN
399          ;        BEGIN                                     ;be6
400          0206      LET PSTAT = 0
401          0207      LET FCLEAR = 1
402          0208      GOSUB ACKK
403          ;        END                                     ;en6
404          ;        END                                     ;en5
405          0209      NXT = PSTAT
406          0211      IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
407          0212      RC3 = OR(RC3,NXT)
408          ;        END                                     ;en2
409          0214      ELSE IF !AND('RC3-PIG':DEV_ALMCT,16)&AND('RC3-PIG':DEV_FLG,8) THEN LET PSTAT = 0
410          0216      IF PSTAT = 5 THEN PSTAT = 0
411          ;        END                                     ;en1
412          ;
413          ;        END DEVICE ALARM
414          ;
415          ;        BEGIN LOGIC FOR GRAPHICS:
416          ;
417          0217      AT0 = AND(RC3_PIG,1)      ; TIMER ON
418          0218      AT1 = AND(RC3_PIG,2)      ; TIME OUT
419          0219      AT2 = AND(RC3_PIG,4)
420          0220      AT3 = AND(RC3_PIG,8)
421          0221      AT4 = AND(RC3_PIG,16)
422          0222      AT5 = AND('RC3-SAMP':DEV_TMP0,32)
423          0223      AT6 = AND(RC3_PIG,64)
424          0224      AT9 = XOR(AT0,AT1)
425          0225      AT10= AND(RC3_PIG,1024)
426          0226      AT11= AND('RSH-RC3-1':DIN_VAL,AT21)
427          0227      AT17= AND('HV-RC3-3':DEV_TMP1,2)
428          0228      AT20= AND('HV-RC3-3':DEV_CMD,2)
429          0229      AT22= AND('HV-RC3-3':DEV_TMP1,32)
430          0230      AT21= AND(AT20,AT22)
431          0231      AT27= AND('HV-RC3-3':DEV_TMP1,2048)      ; JFB01 $T27
432          0232      AT28= AND('HV-RC3-3':DEV_TMP1,4096)      ; JFB01 $T28
433          0233      AT29= AND('HV-RC3-3':DEV_TMP1,8192)      ; JFB01 $T29
434          0234      AT31= AND('HV-RC3-3':DEV_TMP1,32768)      ; JFB01 $T31
435          ;
436          ;
437          0235      GT13 = AND(8192,PB2GLOBAL:DEV_TMP0)
438          ;
439          0236      SA0 = OR('RSH-RC3-1':DIN_VAL,'RSH-CA1-1':DIN_VAL)
440          0237      SA1 = 'YS-RC3-6':DIN_VAL      ; JFB01

```

```

441 0238      SA2 = AND('RC3-PIG':DEV_STAT,4)          ; PIG FLSH
442 0239      SA3 = AND('RC3-PIG':DEV_STAT,8)          ; DI/DRAIN
443 0240      SA4 = AND('RC3-PIG':DEV_STAT,32) ; INTERLOCK
444 0241      SA5 = AND('RC3-SAMP':DEV_STAT,8) ; RC3-SAMP SAMPLE
445 0242      SB0 = AND('HV-RC3-3':DEV_STAT,4) ; RESET
446 0243      SB1 = AND('HV-RC3-3':DEV_STAT,16) ; CHGSTATE
447 0244      SB2 = AND('HV-RC3-3':DEV_STAT,32) ; INTERLOCK
448 0245      LET FTRIP4 = 0
449      ;
450      ;
451      ;
452 0247      IF LAST > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
453 0248      CURENT = OR(LAST,RC3)
454 0249      CURENT = OR(CURENT,ANALOG)
455 0251      IF AND(CURENT,1) THEN LMPVAR = 2
456 0253          ELSE IF AND(CURENT,2) THEN LMPVAR = 1
457 0254          ELSE LMPVAR = 0
458 0255      LAMP ADI,SKDKEY,LMPVAR
459      ;
460 0256      GOTO AGAIN
461      ;
462      ;
463      ;
464      ;
465      ;
466 0257      CLR: LET PSTAT = 0
467 0260      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
468 0261      LET FCLEAR = 1
469 0262      GOSUB ACKK
470 0263      RETURN
471      ;
472 0264      ALM: LET PSTAT = 1
473 0267      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
474 0268      IX_ACK = 2
475 0270      IF AND(PTYPE,3) THEN FALMON = 1
476 0271      LAMP ADI,SKDKEY,2
477      ;
478 0272      GOSUB ACKK
479 0273      RETURN
480      ;
481 0274      ACK1: LET PSTAT = 2
482 0275      LET FACK = 1
483      ;
484      ;
485      ;
486 0277      ACKK: IF ILOOP = 0 THEN STRNGO = "ZS-IDX-2 VALVE 1-29 CLOSED"
487 0279      IF ILOOP = 1 THEN STRNGO = "PSH-ILF-1 FC3 INLET PRES HIGH"
488 0281      IF ILOOP = 2 THEN STRNGO = "YS-RC3-6 RADIATION BYPASS ON"
489 0283      IF ILOOP = 3 THEN STRNGO = "RXS-RC3-1 PC RAD SYSTEM FAIL "
490 0285      IF ILOOP = 4 THEN STRNGO = "RSN-RC3-1 PC SAMPLER RAD HIGH"
491 0287      IF ILOOP = 5 THEN STRNGO = "FSL-RC3-1 PC SAMPLER FLOW LOW"
492 0289      IF ILOOP = 6 THEN STRNGO = "VALVE 1-16 TO TK-C100 OPEN"
493 0291      IF ILOOP = 7 THEN STRNGO = "VALVE 1-17 TO TK-102AW OPEN"
494 0293      IF ILOOP = 33 THEN STRNGO = "RC3-PIG LOW FLOW "
495 0295      IF ILOOP = 24 THEN STRNGO = "START PCBATCH PGM "

```


WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

09:12:51 01-Apr-96

Page 10

```

496 0297      IF ILOOP = 25 THEN STRNGO = "PCBATCH WRONG UNIT"
497 0299      IF ILOOP = 26 THEN STRNGO = "TERMINATE PCBATCH"
498
499      ;
500 0300  AMSG:  IF PTYPE = 1 THEN
501      BEGIN
502      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNGO
503      IF (FACK) THEN
504      BEGIN
505      PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
506      LET FACK = 0
507      END
508      IF (FCLEAR) THEN
509      BEGIN
510      PRINT ADI,"%p ALARM CLEAR",STRNGO
511      LET FCLEAR = 0
512      END
513      IF (FSTART) THEN
514      BEGIN
515      LET FSTART = 0
516      PRINT ADI,"%p",STRNGO
517      END
518      FACK = 0
519      FCLEAR = 0
520      RETURN
521
522 0315  ERRSUB: IF ERRN != ERRNUM() THEN
523      BEGIN
524      TIMEROW STIMEO
525      TIMERCLR STIMEO
526      LET FSN0 = 1
527      LET ERRN = ERRNUM()
528      LET ERRL = ERRLIN()
529      PRINT ADI,"ERROR # %i AT LINE %i",errn,errl
530      END
531      ERSCLR
532 0322      GOTO AGAIN
533
534      ;
535      ; TERM: LET FABORT = 0
536 0324  TERM:  LET STRNGO = "IXBATCH TERMINATED  "
537 0325      LET ILOOP = STAR + 2
538 0326      LET FSTART = 1
539 0327      LET STRNGO = "TERM"
540 0328      GOSUB ACKK
541      END

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 6464 Bytes, 404 Pages
14 Forward Reference(s)

SCBATCH

```

1      ;      S C B A T C H . S E Q                      VERSION 2
2      ;
3      ;      Function:      Sequence program for Lamp function,
4      ;                      message display on Graphic number 014,
5      ;                      and printer messages.
6      ;
7      ;      Written By      : R. M. Kersey
8      ;      Date           : 05-31-92
9      ;      Facility       : 242-A Evaporator Bldg.
10     ;      Machine        : VAX 4104-A
11     ;      Op. System     : VAX/VMS 5.5-2AH
12     ;      D/3 Version    : 6.3-2 BASELINE
13     ;      Compiler       : SIC
14     ;      Build command  : @[320,20]SCBATCH
15     ;
16     ;      UNIT = 16, SKDKEY = 29
17     ;
18     ;      Assoc. UPF:      [300,1] PRCESS,___      PRCESS.LST
19     ;      Assoc. RPF:      [300,1] 242A,___      242A.LST
20     ;      SDB:            [300,1]PCMOA.SDB      PCMOASDB.CMD
21     ;
22     ;      POINTER      SDB VAR.      FUNCTION
23     ;      -----
24     ;      PSTAT      INTn      Current EPN alarm state is an integer tied to
25     ;                      graphic flashing string of value:
26     ;                      0 = previously not alarmed
27     ;                      1 = previously alarmed, not ack
28     ;                      2 = previously alarmed, is ack
29     ;                      3 = alarm is inhibited
30     ;                      4 = input is off scan
31     ;                      5 = input is bad
32     ;
33     ;      PDIN      LEAKn      Special INT variable for multiple white alarms on an EPN
34     ;      PTYPEn      'EPN'
35     ;                      alarm type 0 = none, 1 = white, 2 = yellow,
36     ;                      3 = yellow and red, 4 = red,
37     ;                      5 = yellow and red
38     ;      PREV      FRVn      Reverse acting (not fail safe) flag (1=set)
39     ;      PSKID      SKIDn      Used for analog alarms
40     ;      PPASS      FPASn      Turn lamp off when alarm is ACKnowledged
41     ;      PALARM      FALMn      Analog alarm flag
42     ;      ---      FNING      Unused error flag
43     ;
44     ;      MODIFICATION LOG:
45     ;
46     ;      30-DEC-92      Fixed white alarm print message.      M. Teats
47     ;
48     ;      18-FEB-93      Added BT13 SAMPLE status for RC1-SAMP.      M. Teats
49     ;
50     ;      08-JUN-93      Added comments. CCN# 93JUN01-01.      M. Teats
51     ;
52     ;      21-JUN-93      Removed 3 comment lines. CCN# 93JUN21-02.      M. Teats
53     ;
54     ;      -mct1+ 29-JUN-93      Remove IBAD on RC3-PIG, and baselined for      M. Teats
55     ;                      D/3 version 3.3. CCN# 93JUN29-01.

```

```

56      ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
57      ;
58      ;
59      ;
60      ;      31-JUN-95      Changed ZS-EA1-2D and ZS-RC1-3D to yellow      J. Berger
61      ;
62      ;
63      ;
64      ;      END MOD LOG
65      ;
66      UNIT      SC
67      ;
68      ;      COMMON PB2,WHOLE0
69      DEVON == 'RC1-SAMP':DEV_STAT      ; For FQIC-RESET device on/off check
70      CALOUT == 'FQIC-RC1':CB_OTVL:3      ; For FQIC-RESET calc block outvalue
71      HVEA1 == 'HV-EA1-2':DEV_TMP1
72      PIG0 == 'RC1-PIG':DEV_TMP0
73      PIG1 == 'RC1-PIG':DEV_TMP1
74      ;
75      0000      NOP
76      0001      LET STAR = 25      ; Start message JFB01
77      0002      LET SKDKEY = 29      ; SKID Key
78      0003      LET UNITT = 16      ; SEQ Unit number
79      0004      ASSIGN PDEV = 'SC-ALARM' ; Alarm device
80      0005      ASSIGN PTYPE = TYPE25      ; JFB01
81      0006      LET ADI = 0      ; Alarm destination index
82      0007      LET FSN0 = 0      ; No alarm yet
83      0008      TIMEROFF STIME0      ; Alarm timer off
84      0009      TIMERCLR STIME0      ; Alarm timer clear
85      0010      LET STRNG0 = " "      ; Blank 30 character string
86      0011      LET ERRN = 0      ; Set error number to zero
87      0012      LET ERRL = 0      ; Set error line to zero
88      0013      LET FTRIP4 = 0      ; FTRIP4 = 1 when audible horn is to be turned on
89      0014      GETUNIT TRIP      ; Which sequence unit is program running in
90      0015      IF TRIP != UNITT THEN      ; Abort if running in wrong seq. unit
91      ;
92      0016      BEGIN
93      0017      FSTART = 1
94      0018      LOOP = STAR + 1
95      0019      PTYPE = 0
96      0020      GOSUB ACKK
97      ;      END
98      ;      IF !SELECT() THEN GOTO AGAIN
99      ;
100     ;      B E G I N      I N I T I A L I Z I N G :
101     ;
102     ;
103     ;
104     ;      EPN      VAR      LOOP#      COLOR
105     ;      -----      ----      ----      ----
106     ;      DEVs:      SC-ALARM
107     ;                  HV-RC1-3
108     ;                  RC1-SAMP
109     ;                  RC1-PIG      24      Y
110     ;                  HV-EA1-2
111     ;
112     ;      DINs:      RSH-EA1-1      INTO      0      R

```

111	:	LSL-RC1-1	INT1	1	Y			
112	:	LSL-C1031	INT2	2	Y			
113	:	YS-P-RC1	INT3	3	-			
114	:	YS-RC1-6	INT4	4	Y			
115	:	FSL-RC1-1	INT5	5	-			
116	:	RSR-RC1-1	INT6	6	R			
117	:	RXS-RC1-1	INT7	7	Y			
118	:	LSL-C1032	INT8	8	Y			
119	:	ZS-EA12NM	INT9	9	-			
120	:	ZS-EA1-2D	INT10	10	Y			
121	:	ZS-RC13NM	INT11	11	-			
122	:	ZS-RC1-3D	INT12	12	Y			
123	:					VAR	LOOP#	ALARM SETPT
124	:					---	-----	-----
125	:	AINs:	F1-EA1-3	SKID13	13	-		
126	:		FQ1-EA1-D	SKID14	14	-		
127	:		FQ1-EA1NM	SKID15	15	-		
128	:		R1-EA1-1	SKID16	16	-		
129	:		KY-RC1-1	SKID17	17	-		
130	:		R1-RC1-1	SKID18	18	-		
131	:		F1-RC1-1	SKID19	19	-		
132	:		FQ1-RC1-D	SKID20	20	-		
133	:		FQ1-RC1NM	SKID21	21	-		
134	:		FQIC-RC1	SKID22	22	-		
135	:		C1-EA1-SE	SKID23	23	-		
136	:							
137	:							
138	0021	LET	STRNG0 = "INIT"					
139	0022	INIT:	LET FABORT = 0			; If FABORT = 1 then abort program		
140	0023		LET FSTART = 0			; Program start flag = 0		
141	0024		LET S00 = 0			; For FQIC-RESET RC1-SAMP device status flag		
142	0025		LAMP ADI,SKDKEY,0			; Turn off the annunciator key		
143	0026		LET LSG0 = " "			; Clear the string variables		
144	0027		LET LSG1 = " "					
145	0028		LET LSG2 = " "					
146	0029		LET LSG3 = " "					
147	0030		LET LSG4 = " "					
148	0031		PDEV:DEV_TMP0 = 1			; Set alarm device to non-alarm state		
149	0032		IF AND(PDEV:DEV_ALMCT,16) THEN			; If alarm device is inha, remove the inha		
150	0033		PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)					
151	0034		IF AND(PDEV:DEV_ALMCT,32) THEN			; If alarm device is ack, clear it		
152	0035		PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)					
153	:							
154	:		First, set all flags and integers:					
155	:							
156	0036		ASSIGN PDIM = 'RSR-EA1-1'					
157	0037		ASSIGN PSTAT = INTO					
158	0038		ASSIGN PTYPE = TYPE0					
159	0039		ASSIGN PREV = FRVO					
160	0040		ASSIGN PSKID = SKID0					
161	0041		ASSIGN PPASS = FPAS0					
162	:							
163	0042		FOR ILOOP = 0 TO 23			;JFB01		
164	0043		LET PPASS = 0					
165	0044		LET PREV = 0					

```

166 0046      IF PSTAT > 5 THEN PSTAT = 0      ; Reset current status if undefined
167 0047      LET PSKID = 0                    ; Default previous status to no alarms
168 0048      LET PTYPE = 0                      ; Default to no alarm
169 0049      IF ILOOP < 13 THEN                ; For Dins      JFB01
170          BEGIN
171 0050          LET PTYPE = 2                  ; Default to yellow alarm
172 0051          IF AND(PDIN:DIN_DFLG,3) THEN  ; Check for alarms if defined (must be red or yellow)
173              BEGIN
174 0052              LET PTYPE = 2              ; Default to yellow alarms
175 0054              IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1 ; Reverse acting
176 0056              IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4 ; Set to red critical
177              END
178 0057              STEP PDIN BY 1              ; Point to next DIN
179              END
180 0058          ELSE PTYPE = 0
181 0059          STEP PPASS BY 1
182 0060          STEP PREV BY 1
183 0061          STEP PSTAT BY 1
184 0062          STEP PSKID BY 1
185 0063          STEP PTYPE BY 1
186 0064          NEXT ILOOP
187      ;
188 0065      TYPE3 = 0      ; YS-RC1-1      JFB01
189 0066      TYPE5 = 0      ; FSL-RC1-1 LOW FLOW ALARM REPLACED BY RC1-PIG ALARM
190      ;
191      TYPE12 = 4      ; JFB01
192 0067      TYPE14 = 4      ; JFB01
193      ;
194      TYPE24 = 2      ; RC1-PIG      JFB01
195      ;
196 0068      FRV0 = 0
197 0069      FRV1 = 0
198 0070      FRV2 = 0
199      ;
200 0071      FRV3 = 0      JFB01
201 0072      FRV4 = 1
202 0073      FRV6 = 0
203 0074      FRV7 = 0
204 0075      FRV8 = 0
205 0076      FRV10 = 1      ; JFB01
206 0077      FRV12 = 1      ; JFB01
207 0078      FRV24 = 1      ; JFB01
208      ;
209      FPAS24 = 0      ; JFB01
210      ;
211      END INITIALIZING
212      ;
213      SEND PROGRAM START MESSAGE
214      ;
215 0079      ANALOG = 0
216 0080      ILOOP = STAR
217 0081      FSTART = 1
218 0082      GOSUB ACKK
219 0083      ON ERROR GOSUB ERRSUB
220      ;
221      BEGIN PROGRAM LOOP:
222 0084      AGAIN: MARK STEP "WTUNTLL",0,0

```

```

221      ; JH01 WAITUNTIL SELECT() ! FABORT      ; Wait until micro is selected or program aborted
222      ;      IF (FABORT) THEN GOTO TERM      ; Terminate program if abort flag set
223      0085      MARK STEP "RUNNING",0,0
224      0086      WAIT 1
225      0087      IF (FSNO) THEN                ; If alarm flag is set
226      0088      IF STIMEO > 60 THEN            ; If alarm flag active for 1 minute
227      BEGIN                                        ; Reset timer and turn off flag
228      0089      TIMEROFF STIMEO
229      0090      TIMERCLR STIMEO
230      0091      FSNO = 0
231      0092      ERRN = 0
232      0093      ERRL = 0
233      END
234      ;
235      ;      If SC-ALARM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
236      ;
237      0094      IF (PDEV:DEV_TMP0 = 2) & AND(PDEV:DEV_ALMCT,32) THEN
238      BEGIN
239      0095      PDEV:DEV_TMP0 = 1                ; Put SC-ALARM in ack state
240      0096      FTRIP4 = 1                      ; Set for unit alarm ack
241      0097      ACK = 1                        ; Set for unit alarm ack
242      0098      CRIT
243      0099      LAMP ADI,SKDKEY,1                ; Turn lamp on
244      END
245      ;
246      ;      BEGIN DINS:
247      ;
248      0100      ASSIGN PDIN = 'RSH-EA1-1'
249      0101      ASSIGN PTYPE = TYPEO
250      0102      ASSIGN PREV = FRVO
251      0103      ASSIGN PSTAT = INTO
252      0104      ASSIGN PPASS = FPASO
253      0105      LET CURENT = 0                  ; Used to set lamp status
254      0106      LET LAST = 0
255      0107      LET ANALOG = 0
256      0108      LET NXT = 0
257      0109      LMPVAR = 0
258      0110      BITVAL = 0
259      ;
260      0111      LOP:  FOR ILOOP = 0 TO 12        ; JF801
261      ;      IF !SELECT() THEN GOTO AGAIN
262      0112      IF ( PTYPE > 0 ) THEN                ; If some type of alarm
263      BEGIN
264      0114      IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3      ;be1 ; If EPN is inhibited
265      0116      IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5      ; If EPN is bad input
266      0118      IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4      ; If EPN is disabled
267      0119      IF ( PSTAT < 3 ) THEN                ; If not inhibited
268      BEGIN
269      0120      IF (PSTAT = 1) THEN
270      BEGIN
271      0121      IF (PTYPE != 1) THEN
272      0123      IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
273      0125      IF (FTRIP4) THEN GOSUB ACK1
274      END
275      0128      IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM

```

```

276 0129          IF (PSTAT) THEN
277              BEGIN
278                  IF PDIN:DIN_VAL != PREV THEN GOSUB CLR ;be4
279                  NXT = PSTAT
280                  IF (PPASS) & (PSTAT = 2) THEN NXT = 0
281                  LAST = OR(LAST,NXT)
282                  END
283              ;en4
284 0137          ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8)THEN LET PSTAT = 0
285              END
286              ;en1
287          STEP PDIN BY 1
288          STEP PTYPE BY 1
289          STEP PSTAT BY 1
290          STEP PPASS BY 1
291          NEXT ILOOP
292      ;
293      ;
294      ;
295      ;
296      ;
297      0144      ASSIGN PSKID = SKID13
298      0145      ASSIGN PA1 = 'F1-EA1-3' ; JFB01
299      0146      FOR JLOOP = 13 TO 23 ; JFB01
300      ;
301      0147      IF !SELECT() THEN GOTO AGAIN
302      BEGIN
303          BITVAL = PA1:AI_ALMCT ;be1
304          0149      IF !AND(BITVAL,16) THEN ; If not inhibited
305              BEGIN
306          0150          IF AND(BITVAL,15) THEN ; If alarm exists and is current
307              BEGIN
308          0152          IF AND(BITVAL,32) THEN NXT = 2 ; If alarm is ack
309              ELSE ; If alarm is unack
310              BEGIN ;be4
311          0153          IF PSKID !=BITVAL THEN
312              BEGIN ;be5
313          0155          IF BITVAL < 3 THEN FALMON = 1
314          0156          LET PSKID = BITVAL
315          0157          PDEV:DEV_TMPO = 2
316          0158          NXT = 1
317              END ;en5
318          0159          ELSE NXT = 1
319              END ;en4
320          END ;en3
321      ELSE ; Else there is no alarm
322          BEGIN
323          0160          NXT = 0
324          0161          LET PSKID = 0
325          END
326      END ;en2
327      ELSE ; Else alarm is inhibited
328          BEGIN
329          0162          NXT = 0
330          0163          LET PSKID = 0

```



```

331          END
332 0164      ANALOG = OR(ANALOG,NXT)
333          END                                ;en1
334 0165      STEP PREV BY 1
335 0166      STEP PSTAT BY 1
336 0167      STEP PSKID BY 1
337 0168      STEP PTYPE BY 1
338 0169      STEP PAI BY 1
339 0170      NEXT JLOOP
340          ;
341          ;
342          ;
343          ;
344          ;
345 0171      RC1:  ASSIGN PTYPE = TYPE24                ; JFB01
346 0172          ASSIGN PREV = FRV24                    ; JFB01
347 0173          ASSIGN PSTAT = INT24                   ; JFB01
348 0174          ASSIGN PPASS = FPAS24                  ; JFB01
349          ;
350 0175          IF ( PTYPE > 0 ) THEN
351              IF ( PTYPE > 0 ) THEN
352                  BEGIN                                ;be1
353                      IF AND('RC1-PIG':DEV_ALMCT,16) THEN LET PSTAT = 3      ; EPN is inha
354                      IF AND('RC1-PIG':DEV_ALMCT,4) THEN BLET PSTAT = 5      ; No IBAD on devices
355                      IF IAND('RC1-PIG':DEV_FLG,8) THEN LET PSTAT = 4        ; EPN is dis
356                      IF ( PSTAT < 3 ) THEN
357                          BEGIN                                ;be2
358                              IF PSTAT = 1 THEN
359                                  BEGIN                                ;be3
360                                      IF AND('RC1-PIG':DEV_ALMCT,32) THEN GOSUB ACK1
361                                      IF (FTRIP4) THEN GOSUB ACK1
362                                      END
363                                      IF PSTAT = 0 THEN IF 'RC1-PIG':DEV_ALMCT = 7 THEN
364                                          BEGIN                                ;be4
365                                              CRIT
366                                              LET PSTAT = 1
367                                              'SC-ALARM':DEV_TMPO = 2
368                                              FALMON = 1
369                                              LAMP AD1,SKDKEY,2
370                                              GOSUB ACKK
371                                              NOCRIT
372                                              END                                ;en4
373                                          IF (PSTAT) THEN
374                                              BEGIN                                ;be5
375                                                  IF IAND('RC1-PIG':DEV_ALMCT,7) THEN
376                                                      BEGIN                                ;be6
377                                                          LET PSTAT = 0
378                                                          LET FCLEAR = 1
379                                                          GOSUB ACKK
380                                                          END                                ;en6
381                                                          NXT = PSTAT
382                                                          IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
383                                                          LAST = OR(LAST,NXT)
384                                                          END
385                                  END                                ;en5
386                              END                                ;en2

```

```

386 0206      ELSE IF !AND('RC1-PIG':DEV_ALMCT,16)&AND('RC1-PIG':DEV_FLG,8)THEN LET PSTAT = 0
387 0208      IF PSTAT = 5 THEN PSTAT = 0
388      END                                ;en1
389
390      END DEVICE ALARM (RC1-PIG LOW FLOW ALARM)
391
392 0209      WHITE = 0
393 0210      LET FTRIP4 = 0
394 0212      IF LAST > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
395 0213      CURENT = OR(LAST,WHITE)
396 0214      CURENT = OR(CURENT,ANALOG)
397 0216      IF AND(CURENT,1) THEN LMPVAR = 2
398 0218      ELSE IF AND(CURENT,2) THEN LMPVAR = 1
399 0219      ELSE LMPVAR = 0
400 0220      LAMP ADI,SKDKEY,LMPVAR
401 0221      MARK STEP "BOTTOM",0,0
402 0222      NOCRIT
403
404      ;
405      ;
406      ;
407 0223      AT27 = AND(HVEA1,2048)                ; $T27 = RSH-RC1-1 OFF
408 0224      AT28 = AND(HVEA1,4096)                ; $T28 = RI-RC1-1 OFF
409 0225      AT29 = AND(HVEA1,8192)                ; $T29 = RSH-EA1-1 OFF
410 0226      AT30 = AND(HVEA1,16384)               ; $T30 = RI-EA1-1 OFF
411 0227      AT31 = AND(HVEA1,32768)               ; $T31 = HV-EA1-2 INTERLOCK OFF
412 0228      SA0 = AND(HVEA1,2)                   ; $T17 = HV-EA1-2 INTERLOCK ON
413      ; SA1 = 'ZS-EA1-2D':DIN_VAL              ; HV-EA1-2 DIVERTED = 1
414      ; SA2 = 'ZS-EA12NM':DIN_VAL              ; HV-EA1-2 TO WEIR = 1
415 0229      SA3 = AND('HV-RC1-3':DEV_TMP1,2)     ; $T17 = HV-RC1-3 INTERLOCK ON
416      ; SA4 = 'ZS-RC1-3D':DIN_VAL              ; HV-RC1-3 DIVERTED = 1
417 0230      SA5 = 'ZS-RC13NM':DIN_VAL            ; HV-RC1-3 TO BASIN = 1
418      ; AT0 = AND('HV-RC1-3':DEV_TMP1,32768)    ; $T31 = HV-RC1-3 INTERLOCK OFF
419 0231      GT15 = AND('PB2GLOBAL':DEV_TMP0,32768) ; RC1-SAMP TO RC1-PIG "SAMPLE"
420 0232      SA7 = 'YS-P-RC1':DIN_VAL              ; PUMP P-RC1 SC SAMPL PUMP ON
421 0233      SB0 = AND(P1G1,4)                    ; $T20 = YS-RC1-6 BYPASS ON
422 0234      SB1 = AND(P1G0,1)                    ; $T0 = BTIMER KY-RC1-1 ON
423 0235      SB2 = AND(P1G0,2)                    ; $T1 = BTIMER KY-RC1-1 TIMEOUT
424 0236      AT2 = AND(P1G0,4)
425 0237      AT3 = AND(P1G0,8)
426 0238      AT4 = AND(P1G0,16)                  ; $T4 = RC1-PIG INTERLOCK OFF
427 0239      AT8 = AND(P1G0,256)                 ; $T8 = RC1-PIG DIVERT
428 0240      AT10 = AND(P1G0,1024)               ; $T10 = RC1-PIG FLUSH
429 0241      AT15 = AND('RC1-SAMP':DEV_TMP0,32768) ; $T15 = RC1-SAMP FAULT
430 0242      AT17 = AND(P1G1,2)
431 0243      AT20 = AND(P1G1,16)                   ; $T20 = RC1-PIG BYPASS ON
432 0244      AT21 = AND('RC1-SAMP':DEV_TMP1,32)   ; $T21 = PULSE SAMPLER RC1-SAMP
433 0245      AT23 = AND('RC1-SAMP':DEV_TMP1,128)   ; $T23 = SAMPLE RC1-SAMP
434 0246      BT5 = AND(P1G0,1)
435 0247      BT1 = AND(P1G0,2)                   ; $T1 = BTIMER KY-RC1-1 TIMEOUT
436 0248      BT0 = XOR(BT1,BT5)                  ; $BT0 = BTIMER KY-RC1-1 ON
437 0249      BT2 = AND('RC1-PIG':DEV_STAT,8)
438 0250      BT3 = AND('RC1-PIG':DEV_STAT,16)
439 0251      BT4 = AND('RC1-PIG':DEV_STAT,32)
440 0252      BT5 = AND('RC1-PIG':DEV_STAT,64)

```

```

441 0253      BT6 = AND('RC1-PIG':DEV_STAT,128)
442 0254      BT7 = AND('HV-EA1-2':DEV_STAT,4)
443 0255      BT8 = AND('HV-EA1-2':DEV_STAT,16)
444 0256      BT9 = AND('HV-EA1-2':DEV_STAT,32)
445 0257      BT10 = AND('HV-RC1-3':DEV_STAT,4)
446 0258      BT11 = AND('HV-RC1-3':DEV_STAT,16)
447 0259      BT12 = AND('HV-RC1-3':DEV_STAT,32)
448 0260      BT13 = AND('RC1-SAMP':DEV_STAT,8) ; SAMPLE
449 0261      BT14 = AND('RC1-SAMP':DEV_STAT,16) ; SHUTDOWN
450 0262      BT19 = AND('RC1-SAMP':DEV_STAT,32) ; INTERLOCK
451 0263      BT31 = AND('RC1-SAMP':DEV_TNP1,32768) ; $T31 = RC1-SAMP INTERLOCK OFF
452 0265      IF AND('SC-ALARM':DEV_FLG,64) THEN PUTMANL 'SC-ALARM'
453 0267      IF AND('HV-RC1-3':DEV_FLG,64) THEN PUTMANL 'HV-RC1-3'
454 0269      IF AND('RC1-SAMP':DEV_FLG,64) THEN PUTMANL 'RC1-SAMP'
455 0271      IF AND('RC1-PIG':DEV_FLG,64) THEN PUTMANL 'RC1-PIG'
456 0273      IF AND('HV-EA1-2':DEV_FLG,64) THEN PUTMANL 'HV-EA1-2'
457
458      ;
459      ; BEGIN LOGIC FOR GRAPHICS
460
461      ; BEGIN FQIC - RESET :
462
463      ; Function: For FQIC loops that have a CALC block which performs the
464      ; integration, set the output of the CALC block to zero
465      ; after the RC1-SAMP device is turned off.
466
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473      ;
474      ;
475      ;
476      ;
477      ;
478      ;
479      ;
480      ;
481      ;
482      ;
483      ;
484      ;
485      ;
486      ;
487      ;
488      ;
489      ;
490      ;
491      ;
492      ;
493      ;
494      ;
495      ;

```

```

496      ;
497      ; Turn audible alarm on if white or yellow:
498      ;
499      0292      PDEV:DEV.TMPO = 2
500      0294      IF AND(PTYPE,3)THEN FALMON = 1
501      0295      LAMP ADI,SKDKEY,2
502      0296      CRIT
503      0297      GOSUB ACKK
504      0298      RETURN
505      ;
506      0299      ACK1: LET PSTAT = 2                      ; Ack the alarm
507      0300      LET FACK = 1                          ; Set ACK flag for C program
508      ;
509      ; ALARM TYPER STRINGS:
510      ;
511      0302      ACKK: IF ILOOP = 0 THEN STRNGO = "REBOILER SC RADIATION HI/FAIL"
512      0304      IF ILOOP = 1 THEN STRNGO = "WEIR BOX LEVEL LOW"
513      0306      IF ILOOP = 2 THEN STRNGO = "WEIR BOX OVERFLOW TRAP LVL LO"
514      0308      IF ILOOP = 3 THEN STRNGO = "PUMP RC1 SC SAMPLE PUMP ON "
515      0310      IF ILOOP = 4 THEN STRNGO = "RC1 RADIATION BYPASS SWITCH ON"
516      0312      IF ILOOP = 5 THEN STRNGO = "SC SAMPLER FLOW LOW"
517      0314      IF ILOOP = 6 THEN STRNGO = "SC SAMPLER RADIATION HIGH"
518      0316      IF ILOOP = 7 THEN STRNGO = "SC RADIATION SYSTEM FAIL"
519      0318      IF ILOOP = 8 THEN STRNGO = "WEIR BOX SC-LINE TRAP LEVEL LO"
520      0320      IF ILOOP = 10 THEN STRNGO = "STEAM CND58 DIVERTED TO 102AW" ; JFB01
521      0322      IF ILOOP = 12 THEN STRNGO = "STEAM CND58 DIVERTED TO 102AW" ; JFB01
522      0324      IF ILOOP = 24 THEN STRNGO = "RC1-PIG LOW FLOW" ; JFB01
523      0326      IF ILOOP = 25 THEN STRNGO = "START SCBATCH PGM " ; JFB01
524      0328      IF ILOOP = 26 THEN STRNGO = "SCBATCH WRONG UNIT" ; JFB01
525      0330      IF ILOOP = 27 THEN STRNGO = "TERMINATE SCBATCH " ; JFB01
526      ;
527      ; If acking alarm, then
528      ;
529      0331      AMSG: IF PTYPE = 1 THEN
530      BEGIN
531      ;
532      ; If white alarm, print:
533      ;
534      0333      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM Xp",STRNGO
535      0334      IF (FACK) THEN
536      BEGIN
537      ;
538      ; If white, print it is ack and reset the flag:
539      ;
540      0335      PRINT ADI,"Xp ALARM ACKNOWLEDGE",STRNGO
541      0336      LET FACK = 0
542      END
543      END
544      ;
545      ; If CLEAR, print it is clear and reset flag:
546      ;
547      0337      IF (FCLEAR) THEN
548      BEGIN
549      0338      PRINT ADI,"Xp ALARM CLEAR",STRNGO
550      0339      LET FCLEAR = 0

```

```

551      END
552      ;
553      ; If STARTUP, print startup message:
554      ;
555      0340      IF (START) THEN
556      BEGIN
557      LET FSTART = 0
558      PRINT AD1,"%p",STRNGO
559      END
560      RETURN
561      ;
562      ; If ERROR - print message, start timer, and set flag:
563      ;
564      ERRSUB:  IF ERRN != ERRNUM() THEN
565      BEGIN
566      TIMERON STIMEO
567      TIMERCLR STIMEO
568      LET FENO = 1
569      LET ERRN = ERRNUM()
570      LET ERRL = ERRLIN()
571      PRINT AD1,"ERROR # %i AT LINE %i",errn,errl
572      END
573      ERSCLR
574      GOTO AGAIN
575      ;
576      ; If FABORT = 1, then abort program:
577      ;
578      ; TERM: LET STRNGO = "SCRATCH TERMINATED"
579      ;
580      0354      LET ILOOP = STAR + 2
581      0355      LET FSTART = 1
582      0356      LET STRNGO = "TERM"
583      0357      GOSUB ACKK
584      END

```

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 7216 Bytes, 451 Pages
14 Forward Reference(s)

```

BASINBAT

```

1      :      B A S I N B A T . S E Q                      VERSION 2
2      :
3      :      Function:      Sequence program for Lamp function,
4      :                      message display on Graphic number 022,
5      :                      and printer messages.
6      :
7      :      Written By      : R. W. Kersey
8      :      Date           : 05-31-92
9      :      Facility       : 242-A Evaporator Bldg.
10     :      Machine        : VAX 4104-A
11     :      Op. System     : VAX/VMS 5.5-2AH
12     :      D/3 Version    : 6.3-2 BASELINE
13     :      Compiler       : SIC
14     :      Build command  : @[320,20]BASINBAT
15     :
16     :      UNIT = 17, SKDKEY = 28
17     :
18     :      Assoc. UPF:      [300,1] _PROCESS_      PROCESS.LST
19     :      Assoc. RPF:      [300,1] _242A_        242A.LST
20     :      SDB:           [300,1] PCMOA.SDB      PCMOASDB.CMD
21     :
22     :      POINTER  SDB VAR.      FUNCTION
23     :      -----
24     :      PSTAT    INTn          Current EPN alarm state is an integer tied to
25     :                      graphic flashing string of value:
26     :                      0 = previously not alarmed
27     :                      1 = previously alarmed, not ack
28     :                      2 = previously alarmed, is ack
29     :                      3 = alarm is inhibited
30     :                      4 = input is off scan
31     :                      5 = input is bad
32     :
33     :      PDIN      LEAKn          Special INT variable for multiple white alarms on an EPN
34     :      PTYPE     'EPN'          ---
35     :                      alarm type 0 = none, 1 = white, 2 = yellow,
36     :                      3 = yellow and yellow, 4 = red,
37     :                      6 = yellow and red
38     :      PREV      FRVn          Reverse acting (not fail safe) flag (1=set)
39     :      PSKID     SKIDn         Used for analog alarms
40     :      PPASS      FPASn         Turn lamp off when alarm is ACKnowledged
41     :      PALARM     FALMn         Analog alarm flag
42     :      ---          FNING       Unused error flag
43     :
44     :      MODIFICATION LOG:
45     :
46     :      30-DEC-92      Fixed white alarm print message.      M. Teats
47     :
48     :      -mct1+ 17-MAY-93      Added P-207A-S yellow device alarm and      M. Teats
49     :                      removed white analog pump amp alarms to make
50     :                      consistent with documentation.
51     :
52     :      09-JUN-93      Added comments and baselined for      M. Teats
53     :                      D/3 version 3.3. CCN# 93JUN01-01.
54     :
55     :      15-DEC-94      Revised header to indicate the new VMS      J. Berger
56     :                      system. Rebaselined document. Removed

```

```

56      ;                               obsolete Select() statements.
57      ;
58      ;   END OF MOD LOG
59      ;
60      UNIT      BASINS
61              COMMON A350
62      ;
63      0000      NOP
64      0001      LET STAR = 14                ; Start message
65      0002      LET SKDKEY = 28              ; SKID Key
66      0003      LET UNITT = 17              ; SEQ Unit number
67      0004      ASSIGN PDEV = 'BASIN-ALM'    ; Alarm device
68      0005      ASSIGN PTYPE = TYPE14
69      0006      LET ADI = 0                  ; Alarm destination index
70      0007      LET FSNO = 0                ; No alarm yet
71      0008      TIMEROFF STIMEO              ; Alarm timer off
72      0009      TIMERCLR STIMEO             ; Alarm timer clear
73      0010      LET STRNGO = " "            ; Blank 30 character string
74      0011      LET ERRN = 0                ; Set error number to zero
75      0012      LET ERRL = 0                ; Set error line to zero
76      0013      LET FTRIP4 = 0              ; FTRIP4 = 1 when audible horn is to be turned on
77      0014      GETUNIT TRIP                ; Which sequence unit is program running in
78      0015      IF TRIP != UNITT THEN       ; Abort if running in wrong seq. unit
79      BEGIN
80      0016      FSTART = 1
81      0017      ILOOP = STAR + 1
82      0018      PTYPE = 0
83      0019      GOSUB ACKK
84      0020      GOTO TERM
85      END
86      ;   IF !SELECT() THEN GOTO AGAIN
87      ;
88      ;   B E G I N   I N I T I A L I Z I N G :
89      ;
90      0021      STRNGO = "INIT"
91      0022      INIT:      LET FABORT = 0      ; If FABORT = 1 then abort program
92      0023      LET FSTART = 0                ; Program start flag = 0
93      0024      LAMP ADI,SKDKEY,0            ; Turn off the annunciator key
94      0025      LET LSGO = " "              ; Clear the string variables
95      0026      LET LSG1 = " "
96      0027      LET LSG2 = " "
97      0028      LET LSG3 = " "
98      0029      LET LSG4 = " "
99      0030      PDEV:DEV TMOPO = 1            ; Set alarm device to non-alarm state
100     0031      IF AND(PDEV:DEV ALMCT,16) THEN ; If alarm device is inha, remove the inha
101     0032      PDEV:DEV ALMCT=XOR(PDEV:DEV ALMCT,16)
102     0033      IF AND(PDEV:DEV ALMCT,32) THEN ; If alarm device is ack, clear it
103     0034      PDEV:DEV ALMCT = XOR(PDEV:DEV ALMCT,32)
104     ;
105     ;   First, set all flags and integers:
106     ;
107     0035      ASSIGN PDIN = 'YS-SCP3IL'
108     0036      ASSIGN PSTAT = INTO
109     0037      ASSIGN PTYPE = TYPEO
110     0038      ASSIGN PREV = FRVO

```



```

111 0039      ASSIGN PPASS = FPASO
112      ;
113 0040      FOR ILOOP = 0 TO 10
114 0041      LET PSTAT = 0
115 0042      LET PTYPE = 2
116 0043      LET PREV = 0
117 0044      LET PPASS = 0
118 0045      IF ILOOP < 10 THEN                                ; For digitals. -mct1+
119      BEGIN                                                ; -mct1+
120 0046      IF AND(PDIN:DIN_DFLG,3) THEN                    ; Check for alarms (must be red or yellow)
121      BEGIN
122 0047      PTYPE = 2                                          ; Default to yellow alarm
123 0049      IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1      ; Reverse acting
124 0051      IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4      ; Set to red critical
125      END
126 0052      STEP PDIN BY 1                                    ; -mct1+
127      END                                                  ; -mct1+
128 0053      STEP PSTAT BY 1
129 0054      STEP PTYPE BY 1
130 0055      STEP PREV BY 1
131 0056      STEP PPASS BY 1
132 0057      NEXT ILOOP
133      ;
134 0058      FRV0 = 0
135 0059      FRV1 = 0
136 0060      FRV2 = 1
137 0061      FRV3 = 0
138 0062      FRV4 = 0
139 0063      FRV5 = 0
140 0064      FRV6 = 0
141 0065      FRV7 = 0
142 0066      FRV8 = 0
143 0067      FRV9 = 0
144      ;
145 0068      TYPE2 = 0
146 0069      TYPE5 = 1
147 0070      TYPE7 = 1
148 0071      TYPE9 = 1
149 0072      TYPE10 = 2                                       ; -mct1+
150      ;
151 0073      INT10 = 0                                         ; -mct1+
152      ;
153      ;
154      ;
155      SEND PROGRAM START MESSAGE
156      ;
157 0074      ILOOP = STAR
158 0075      FSTART = 1
159 0076      GOSUB ACKK
160 0077      ON ERROR GOSUB ERRSUB
161      ;
162      ;
163      ;
164 0078  AGAIN:  MARK STEP "WTUNT",0,0
165      ;          WAITUNTIL SELECT() | FABORT                ; Wait until micro selected or program aborted

```

```

166      ;      IF (FABORT) THEN GOTO TERM      ; Terminate program if abort flag set
167 0079      MARK STEP "RNNING",0,0
168 0080      WAIT 1
169 0081      IF (FSMO) THEN      ; If alarm flag is set
170 0082      IF STIME0 > 60 THEN      ; If alarm flag active for 1 minute
171      BEGIN      ; Reset timer and turn off flag
172 0083      TIMEROFF STIME0
173 0084      TIMERCLR STIME0
174 0085      LET FSMO = 0
175 0086      LET ERRN = 0
176 0087      LET ERRL = 0
177      END
178      ;
179      ;      IF BASIN-ALM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
180      ;
181 0088      IF (PDEV:DEV_TMP0 = 2) & AND(PDEV:DEV_ALMCT,32) THEN
182      BEGIN
183 0089      PDEV:DEV_TMP0 = 1      ; Put BASIN-ALM in ack state
184 0090      LET FTRIP4 = 1      ; Set for unit alarm ack
185 0091      LET ACK = 1      ; Set for unit alarm ack
186 0092      CRIT
187 0093      LAMP ADI,SKDKEY,1      ; Turn lamp on
188      END
189      ;
190      ;      BEGIN DIMs:
191      ;
192 0094      ASSIGN PDIN = 'YS-SCP3IL'
193 0095      ASSIGN PSTAT = INTO
194 0096      ASSIGN PTYPE = TYPED
195 0097      ASSIGN PREV = FRVO
196 0098      ASSIGN PPASS = FPASO
197 0099      LET CURENT = 0      ; Used to set lamp status
198 0100      LET LAST = 0
199 0101      LET ANALOG = 0
200 0102      LET NXT = 0
201 0103      LMPVAR = 0
202 0104      BITVAL = 0
203      ;
204 0105      LOP:      FOR ILOOP = 0 TO 9
205 0106      IF ( PTYPE > 0 ) THEN      ; If some type of alarm
206      BEGIN
207 0108      IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3      ;be1      ; If EPN is inhibited
208 0110      IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5      ; If EPN is bad input
209 0112      IF IAND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4      ; If EPN is disabled
210 0113      IF ( PSTAT < 3 ) THEN      ; If not inhibited
211      BEGIN      ;be2
212 0114      IF PSTAT = 1 THEN
213      BEGIN      ;be3
214 0115      IF PTYPE != 1 THEN
215 0117      IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
216 0119      IF (FTRIP4) THEN GOSUB ACK1
217      END      ;en3
218 0122      IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
219 0123      IF PSTAT THEN
220      BEGIN      ;be4

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:33:59 24-Feb-95

Page 5

```

221 0125          IF PDIN:DIN_VAL != PREV THEN GOSUB CLR
222 0126          NXT = PSTAT
223 0128          IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
224 0129          LAST = OR(LAST,NXT)
225          END                                ;en4
226          END                                ;en2
227 0131          ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8)THEN LET PSTAT = 0
228          END                                ;en1
229 0132          STEP PDIN BY 1
230 0133          STEP PSTAT BY 1
231 0134          STEP PTYPE BY 1
232 0135          STEP PREV BY 1
233 0136          STEP PPASS BY 1
234 0137          NEXT ILOOP
235          ;
236          ;
237          ;
238          ;+mct1- BEGIN P-207A-S DEVICE ALARM:
239          ;
240 0138 SMP:      ILOOP = 10                                ; P-207A-S
241 0139          ASSIGN PSTAT = INT10
242 0140          ASSIGN PTYPE = TYPE10
243          ;
244 0141          IF !SELECT() THEN GOTO AGAIN
245          IF (TYPE10) THEN                                ; If some type of alarm
246          BEGIN                                          ;be1
247 0143          IF AND('P-207A-S':DEV_ALMCT,16) THEN PSTAT = 3      ; If inhibited
248          IF ( PSTAT < 3 ) THEN                          ; If not inhibited
249          BEGIN                                          ;be2
250 0145          IF PSTAT = 1 THEN                                ; If previously not alarmed
251          BEGIN                                          ;be3
252 0146          IF PTYPE != 1 THEN                                ; If not white
253 0148          IF AND('P-207A-S':DEV_ALMCT,32) THEN GOSUB ACK1    ; If acknowledged
254 0150          IF (FTRIP4) THEN GOSUB ACK1                    ; If system acknowledged
255          END                                          ;en3
256          ;
257          ; If not alarmed but device status is alarming:
258          ;
259 0152          IF PSTAT = 0 THEN IF AND('P-207A-S':DEV_STAT,64) THEN
260          BEGIN                                          ;be4
261 0153          LET PSTAT = 1                                ; Update the alarm status
262 0156          IF PTYPE != 1 THEN IF AND('P-207A-S':DEV_ALMCT,32)THEN 'P-207A-S':DEV_ALMCT = XOR('P-207A-S':DEV_ALMCT,32)
263 0157          LET PDEV:DEV_TMP0 = 2                    ; Set BASIN-ALM to alarm status
264 0159          IF AND(PTYPE,3) THEN FALMON = 1          ; Turn on audible horn
265 0160          CRIT
266 0161          LAMP ADI,SKDKEY,2                        ; Blink the SKID key
267 0162          GOSUB ACKK
268          END                                          ;en4
269          ;
270 0163          IF (PSTAT) THEN                                ; If previously alarmed
271          BEGIN                                          ;be5
272 0164          IF !AND('P-207A-S':DEV_STAT,64) THEN          ; If device status is not alarming
273          BEGIN                                          ;be6
274 0165          PSTAT = 0                                ; Update the alarm status
275 0168          IF PTYPE = 1 THEN IF AND('P-207A-S':DEV_ALMCT,32)THEN 'P-207A-S':DEV_ALMCT = XOR('P-207A-S':DEV_ALMCT,32)
276 0169          LET FCLEAR = 1

```

```

276 0170          GOSUB ACKK
277          END                                ;en6
278          ;
279          ; Manage The Lamp:
280          ;
281 0171          NXT = PSTAT
282 0173          IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
283 0174          LAST = OR(LAST,NXT)
284          END                                ;en5
285          END                                ;en2
286 0176          ELSE IF !AND('P-207A-S':DEV_ALMCT,16) THEN PSTAT = 0
287          END                                ;en1
288          ;
289          ;--mct1+ END P-207A-S DEVICE ALARM
290          ;
291 0177          MARK STEP "BELOW",0,0
292 0178          FTRIP4 = 0
293 0179          FTRIP3 = 0
294 0181          IF LAST > 3 THEN PRINT 0," LAMP VARIABLE IS GREATER THAN 3!"
295 0182          CURENT = OR(LAST,WHITE)
296 0184          IF AND(CURENT,1) THEN LMPVAR = 2
297 0186          ELSE IF AND(CURENT,2) THEN LMPVAR = 1
298 0187          ELSE LMPVAR = 0
299 0188          LAMP ADI,SKDKEY,LMPVAR
300 0189          MARK STEP "BOTTOM",0,0
301 0190          NOCRIT
302          ;
303          ;
304          ;
305 0191          SA0 = AND(16,'207-A-P3':DEV_STAT)                ; SHUTDOWN
306 0192          SA1 = AND(32,'207-A-P3':DEV_STAT)                ; INTERLOCK
307 0193          AT24 = OR('LDS-350-2':DIN_VAL,'LSH-350-2':DIN_VAL) ;
308 0194          AT25 = AND('207-A-P3':DEV_TMP1,512)
309 0195          AT26 = AND('207-A-P3':DEV_TMP1,1024)
310 0196          AT27 = AND('207-A-P3':DEV_TMP1,2048)
311 0197          AT28 = AND('207-A-P3':DEV_TMP1,4096)
312 0198          AT30 = AND('207-A-P3':DEV_TMP1,16384)
313 0199          AT31 = AND('207-A-P3':DEV_TMP1,32768)            ; INTERLOCK
314          ;
315 0200          IF AT28 THEN
316 0201          IF FLGO THEN
317          BEGIN
318 0202          TIMERON OSTMR
319 0203          TIMERCLR OSTMR
320 0204          FLGO = 1
321          END
322          ;
323 0205          IF FLGO THEN
324          BEGIN
325 0206          IF OSTMR > 60 THEN
326          BEGIN
327 0207          TIMEROFF OSTMR
328 0208          TIMERCLR OSTMR
329          END
330 0210          IF AT28 = 0 THEN FLGO = 0

```

```

331      END
332
333      BEGIN LOGIC FOR DEVICES:
334
335      DEVICES TO LOCK IN MANUAL:
336
337 0212      IF AND('BASIN-ALM':DEV_FLG,64) THEN PUTMANL 'BASIN-ALM'
338 0214      IF AND('P-207A-S':DEV_FLG,64) THEN PUTMANL 'P-207A-S'
339 0216      IF AND('207-A-P3':DEV_FLG,64) THEN PUTMANL '207-A-P3'
340 0218      IF AND('HV-SC-1A':DEV_FLG,64) THEN PUTMANL 'HV-SC-1A'
341 0220      IF AND('HV-SC-2A':DEV_FLG,64) THEN PUTMANL 'HV-SC-2A'
342 0222      IF AND('HV-SC-3A':DEV_FLG,64) THEN PUTMANL 'HV-SC-3A'
343 0224      IF AND('HV-SC-2B':DEV_FLG,64) THEN PUTMANL 'HV-SC-2B'
344 0226      IF AND('HV-SC-1B':DEV_FLG,64) THEN PUTMANL 'HV-SC-1B'
345 0228      IF AND('HV-SC-3B':DEV_FLG,64) THEN PUTMANL 'HV-SC-3B'
346
347 0229      GOTO AGAIN
348
349      END PROGRAM LOOP
350
351      BEGIN CLEAR, ALARM, ACKNOWLEDGE, MESSAGE, ERROR, AND TERMINATE SUBROUTINES:
352
353 0230 CLR: LET PSTAT = 0
354
355      ; If not white, then unack:
356
357 0233      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
358 0234      LET FCLEAR = 1
359 0235      GOSUB ACKK
360 0236      RETURN
361
362 0237 ALM: LET PSTAT = 1
363
364      ; If not white, then unack:
365
366 0240      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
367
368      ; Turn audible ;alarm on if white or yellow:
369
370 0241      PDEV:DEV_TMP0 = 2
371 0242      LAMP ADI,SKDKEY,2
372 0244      IF !AND(PTYPE,4) THEN FALMON = 1
373 0245      CRIT
374 0246      LET FTRIP4 = 0
375 0247      GOSUB ACKK
376 0248      RETURN
377
378 0249 ACK1: LET PSTAT = 2 ; Ack the alarm
379 0250      LET FACK = 1 ; Set ACK flag for C program
380
381      ALARM TYPER STRINGS:
382
383 0252 ACKK: IF ILOOP = 0 THEN STRNG0 = "PUMP P3 INTERLOCK "
384 0254      IF ILOOP = 1 THEN STRNG0 = "207-A SUMP LEVEL HIGH "
385 0256      IF ILOOP = 3 THEN STRNG0 = "SC BASINS LEVEL HIGH "

```

```

Sequence Instruction Compiler Listing      15:33:59  24-Feb-95      Page 8

396 0258 IF ILOOP = 4 THEN STRINGO = "SC BASIN-1 LEVEL HIGH "
397 0260 IF ILOOP = 5 THEN STRINGO = "SC BASIN-1 LEVEL LOW "
398 0262 IF ILOOP = 6 THEN STRINGO = "SC BASIN-2 LEVEL HIGH "
399 0264 IF ILOOP = 7 THEN STRINGO = "SC BASIN-2 LEVEL LOW "
400 0266 IF ILOOP = 8 THEN STRINGO = "SC BASIN-3 LEVEL HIGH "
401 0268 IF ILOOP = 9 THEN STRINGO = "SC BASIN-3 LEVEL LOW "
402 0270 IF ILOOP = 10 THEN STRINGO = "P207A SUMP PMP ALARM "
403 0272 IF ILOOP = 14 THEN STRINGO = "BASINBAT-SEQ START "
404 0274 IF ILOOP = 15 THEN STRINGO = "BASINBAT IN WRONG UNIT"
405 0276 IF ILOOP = 16 THEN STRINGO = "BASINBAT TERMINATING "
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440

; MSG:
397 BEGIN
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440

; If white alarm, print:
; IF (PACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRINGO
; IF (PACK) THEN
; BEGIN
;
; If white, print it is ack and reset the flag:
; PRINT ADI,"%p ALARM ACKNOWLEDGE",STRINGO
; LET FACK = 0
; END
;
; If CLEAR, print it is clear and reset flag:
; IF (FCLEAR) THEN
; BEGIN
; PRINT ADI,"%p ALARM CLEAR",STRINGO
; LET FCLEAR = 0
; END
;
; If STARTUP, print startup message:
; IF (FSTART) THEN
; BEGIN
; LET FSTART = 0
; PRINT ADI,"%p",STRINGO
; END
; RETURN
;
; If ERROR - print message, start timer, and set flag:
; ERRSUB: IF ERRL = ERRNUM() THEN
; BEGIN
; TIMERON STIMEO
; TIMERCLR STIMEO
; LET FSNO = 1
; LET ERRL = ERRNUM()
; LET ERRL = ERRNUM()
; PRINT ADI,"ERROR # %i AT LINE %i",errn,errl
; END

```

```

441 0297      ERSCLR
442 0298      GOTO AGAIN
443
444      : if FABORT = 1 then abort program:
445      :
446      : TERM: LET FABORT = 0
447      : TERM: LET STRINGO = "BASINBAT TERMINATED "
448      : TERM: LET ILOOP = STAR + 2
449      : TERM: LET FSTART = 1
450      : TERM: LET STRINGO = "TERM"
451      : TERM: GOSUB ACKK
452      : TERM: END

```

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 5936 Bytes, 371 Pages
14 Forward Reference(s)

```

UTILBATCH

MHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:37:09 24-Feb-95

Page 1

```

1      ;      U T I L B A T C H . S E Q                      VERSION 2
2      ;
3      ;      Function:      Sequence program for Lamp function (SKDKEY 46),
4      ;                      message display on Graphic number 021, and
5      ;                      printer messages. Sequence Unit 18.
6      ;
7      ;      Written By      : R. N. Kersey
8      ;      Date            : 05-31-92
9      ;      Facility        : 242-A Evaporator Bldg.
10     ;      Machine         : VAX 4104-A
11     ;      Op. System      : VAX/VMS 5.5-2AH
12     ;      D/3 Version     : 6.3-2 BASELINE
13     ;      Compiler        : SIC
14     ;      Build command   : @ [320,20]UTILBATCH
15     ;
16     ;      UNIT = 18, SKDKEY = 46
17     ;
18     ;      Assoc. UPF:      [300,1]_PROCESS.____ PROCESS.LST
19     ;      Assoc. RPF:      [300,1]_242A.____ 242A.LST
20     ;      SDB:            [300,1]PCMDA.SDB PCMDASDB.CMD
21     ;
22     ;      POINTER      SDB VAR.      FUNCTION
23     ;      -----
24     ;      PSTAT      INTn      Current EPN alarm state is an integer tied to
25     ;                      flashing string of value:
26     ;                      0 = previously not alarmed
27     ;                      1 = previously alarmed, not ack
28     ;                      2 = previously alarmed, is ack
29     ;                      3 = alarm is inhibited
30     ;                      4 = input is off scan
31     ;                      5 = input is bad
32     ;      LEAKn      Special INT variable for multiple white alarms on an EPN
33     ;      PDIN      'EPN'
34     ;      PTYPEn      alarm type 0 = none, 1 = white, 2 = yellow,
35     ;                      3 = yellow and yellow, 4 = red,
36     ;                      6 = yellow and red
37     ;      PREV      FRVn      Reverse acting (not fail safe) flag (1=set)
38     ;      PSKID      SKIDn      Used for analog alarms
39     ;      PPASS      FPASn      Turn lamp off when alarm is Acknowledged
40     ;      PALARM      FALMn      Analog alarm flag
41     ;      ---      FNING      Unused error flag
42     ;
43     ;      MODIFICATION LOG:
44     ;
45     ;      11-AUG-92      - Changed PI-IA-1 white low alarm 15 to 18      M. Teats
46     ;                      - Changed PI-PA-1 white low alarm 80 to
47     ;                      yellow low alarm 95. Added section 'JOP:'
48     ;                      - Fixed FTRIP4 to prevent auto-acknowledge
49     ;                      - Fixed print message error
50     ;                      - Improved to cover Red alarms
51     ;                      - Added comments
52     ;
53     ;      30-NOV-92      - Added 3 new UPS DINS per ECN 168693      M. Teats
54     ;                      - Fixed SKDKEY reflashing when acknowledge
55     ;

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:37:09 24-Feb-95

Page 2

```

56      ;      30-DEC-92      Fixed white alarm print message.      M. Teats
57      ;
58      ;      09-JUN-93      Added comments and baselined for      M. Teats
59      ;      D/3 version 3.3.  CCN# 93JUN01-01.
60      ;
61      ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
62      ;      system. Rebaselined document. Remove
63      ;      obsolete Select() statements.
64      ;
65      ;      END MOD LOG
66      ;
67      UNIT      UTIL
68      ;
69      0000      NOP
70      0001      STAR = 26      ; Start message
71      0002      LET SKDKEY = 46      ; SKID Key
72      0003      LET UNITT = 18      ; SEQ Unit number
73      0004      ASSIGN PDEV = 'UTILITIES'      ; Alarm device
74      0005      ASSIGN PTYPE = TYPE26
75      0006      LET ADI = 0      ; Alarm destination index
76      0007      LET FSN0 = 0      ; No alarm yet
77      0008      TIMEROFF STIMEO      ; Alarm timer off
78      0009      TIMERCLR STIMEO      ; Alarm timer clear
79      0010      STRNG0 = " "      ; Blank 30 character string
80      0011      LET ERRN = 0      ; Set error number to zero
81      0012      LET ERRL = 0      ; Set error line to zero
82      ;old      DLEAK0 = 0      ; Unused flag for inhibited alarms
83      0013      FTRIP4 = 0      ; FTRIP4 = 1 when audible horn is to be turned on
84      0014      GETUNIT TRIP      ; Which sequence unit is program running in
85      0015      IF TRIP != UNITT THEN      ; Abort if running in wrong seq. unit
86      BEGIN
87      0016      FSTART = 1
88      0017      ILOOP = STAR + 1
89      0018      PTYPE = 0
90      0019      GOSUB ACKK
91      0020      GOTO TERM
92      END
93      ;      IF ISELECT() THEN GOTO AGAIN
94      ;
95      ;
96      ;      BEGIN INITIALIZING :
97      ;
98      ;      ILOOP 0-10 are DINs, 11-25 are AINs
99      ;
100     0021      STRNG0 = "INIT"
101     0022      INIT:      LET FABORT = 0      ; If FABORT = 1 then abort program
102     0023      LET FSTART = 0      ; Program start flag = 0
103     0024      LAMP ADI,SKDKEY,0      ; Turn off the annunciator key
104     0025      LSG0 = " "      ; Clear the string variables
105     0026      LSG1 = " "
106     0027      LSG2 = " "
107     0028      LSG3 = " "
108     0029      LSG4 = " "
109     0030      PDEV:DEV_TMPO = 1      ; Set alarm device to non-alarm state
110     0031      IF AND(PDEV:DEV_ALMCT,16) THEN      ; if alarm device is inha, remove the inha
111     0032      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:37:09 24-Feb-95

Page 3

```

111 0033      IF AND(PDEV:DEV_ALMCT,32) THEN      ; If alarm device is ack, clear it
112 0034      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
113      ;
114      ;
115      ; First, set all flags and integers:
116 0035      ASSIGN PDIN = 'PSLCPE1/2'
117 0036      ASSIGN PSTAT = INTO
118 0037      ASSIGN PTYPE = TYPE0
119 0038      ASSIGN PREV = FRV0
120 0039      ASSIGN PPASS = FPAS0
121 0040      ASSIGN PALARM = FALMO
122 0041      ASSIGN PSKID = SKID0
123      ;
124 0042      FOR ILOOP = 0 TO 25
125 0043      LET PSKID = 0
126 0044      LET PPASS = 0
127 0045      LET PREV = 0
128 0046      LET PTYPE = 2      ; Default to Yellow alarm
129 0048      IF PSTAT > 5 THEN PSTAT = 0      ; Reset current status if undefined
130 0049      PALARM = 0      ; Default previous status to no alarms
131 0050      IF ILOOP <= 10 THEN      ; For DINs
132      BEGIN
133 0051      IF AND(PDIN:DIN_DFLG,3) THEN      ; Check for alarms (must be red or yellow)
134      BEGIN
135 0052      LET PTYPE = 2      ; Default to Yellow alarm
136 0054      IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1      ; Reverse acting
137 0056      IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4      ; Set to red critical alarm
138      END
139 0058      IF ILOOP < 10 THEN STEP PDIN BY 1      ; Point to next DIN
140      END
141 0059      ELSE PTYPE = 1      ; For AINs, set to White alarm
142 0060      STEP PSTAT BY 1
143 0061      STEP PTYPE BY 1
144 0062      STEP PREV BY 1
145 0063      STEP PPASS BY 1
146 0064      STEP PALARM BY 1
147 0065      STEP PSKID BY 1
148 0066      NEXT ILOOP
149      ;
150 0067      TYPE2 = 1      ; YS-CPE1-1
151 0068      TYPE3 = 1      ; YS-CPE2-1
152 0069      TYPE6 = 4      ; YS-GEN-1
153 0070      TYPE7 = 4      ; ZS-PWR-1
154 0071      TYPE8 = 4      ; JXS-PWR-3
155 0072      TYPE12 = 2      ; P1-PA-1 LO
156      ;
157      ;
158      ; For White LOLO and LO Alarms:
159 0073      FRV15 = 1      ; P1-1A-1 LO
160 0074      FRV18 = 1      ; T1-AMBS-1 LOLO
161 0075      FRV19 = 1      ; T1-AMBS-1 LO
162 0076      FRV22 = 1      ; T1-AMBS-2 LOLO
163 0077      FRV23 = 1      ; T1-AMBS-2 LO
164      ;
165 0078      AIO = 18.0      ; P1-1A-1 LO      ILOOP = INT #:
                                         15

```

```

166 0079      A11 = 25.0      ; P1-IA-1 HI      LEAK4 = 4+      16
167 0080      A12 = 120.0     ; P1-PA-1 HI      LEAK1 = 4+      17
168 0081      A13 = 48.0      ; T1-AMBS-1 LOLO    LEAK2 = +      18
169 0082      A14 = 55.0      ; T1-AMBS-1 LO      LEAK2 = 2+     19
170 0083      A15 = 80.0      ; T1-AMBS-1 HI      LEAK2 = 4+     20
171 0084      A16 = 90.0      ; T1-AMBS-1 HIHI    LEAK2 = 6+     21
172 0085      A17 = 48.0      ; T1-AMBS-2 LOLO    LEAK3 = +      22
173 0086      A18 = 55.0      ; T1-AMBS-2 LO      LEAK3 = 2      23
174 0087      A19 = 80.0      ; T1-AMBS-2 HI      LEAK3 = 4+     24
175 0088      A110 = 90.0     ; T1-AMBS-2 HIHI    LEAK3 = 6+     25
176          ;
177          ;      END INITIALIZING
178          ;
179          ;      SEND PROGRAM START MESSAGE
180          ;
181 0089      ANALOG = 0
182 0090      ILOOP = STAR
183 0091      FSTART = 1
184 0092      GOSUB ACKK
185 0093      ON ERROR GOSUB ERRSUB
186          ;
187          ;      BEGIN PROGRAM LOOP:
188          ;
189 0094      AGAIN:  MARK STEP "WTUNTIL",0,0
190          ;      WAITUNTIL SELECT() ; FABORT      ; Wait until micro is selected or program aborted
191          ;      IF (FABORT) THEN GOTO TERM      ; Terminate program if FABORT flag set
192 0095      MARK STEP "RNNING",0,0
193 0096      WAIT 1
194 0097      IF (FSNO) THEN      ; If alarm flag is set
195 0098      IF STIME0 > 60 THEN      ; If alarm flag active for 1 minute
196          BEGIN      ; Reset timer and turn off flag
197 0099      TIMEROFF STIME0
198 0100      TIMERCLR STIME0
199 0101      FSNO = 0
200 0102      ERRN = 0
201 0103      ERRL = 0
202          END
203          ;
204          ;
205          ;
206 0104      IF (PDEV:DEV_TMPO = 2) & AND(PDEV:DEV_ALMCT,32) THEN
207          BEGIN
208 0105      PDEV:DEV_TMPO = 1      ; Put UTILITIES in ACKMWLDG state
209 0106      FTRIP4 = 1      ; Set for Unit alarm ack
210 0107      ACK = 1      ; Set for Unit alarm ack
211 0108      CRIT
212 0109      LAMP AD1,SKDKEY,1      ; Turn LAMP on
213          END
214          ;
215          ;      BEGIN DINS:
216          ;
217 0110      ASSIGN PDIN = 'PSLCPE1/2'
218 0111      ASSIGN PTYPE = TYPE0
219 0112      ASSIGN PREV = FRVO
220 0113      ASSIGN PSTAT = INTO

```

```

221 0114      ASSIGN PPASS = FPASO
222 0115      ASSIGN PSKID = SKID0
223 0116      LET CURENT = 0          ; Used to set lamp status
224 0117      LET LAST = 0
225 0118      LET ANALOG = 0
226 0119      LET NXT = 0
227 0120      LMPVAR = 0
228 0121      BITVAL = 0
229
230 0122      LOP:   FOR ILOOP = 0 TO 10
231                IF !SELECT() THEN GOTO AGAIN
232 0123      ;
233                IF ( PTYPE > 0 ) THEN          ; If some type of alarm
234                BEGIN                          ;be1
235                IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3 ; If EPN inhibited
236                IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5 ; If EPN is bad input
237                IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4 ; If EPN is disabled
238                IF (PSTAT < 3) THEN              ; If Not Inhibited
239                BEGIN                          ;be2
240                IF (PSTAT = 1) THEN
241                BEGIN                          ;be3
242                IF (PTYPE != 1) THEN
243                IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
244                IF (FTRIP4) THEN GOSUB ACK1
245                END                          ;en3
246                IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
247                IF (PSTAT) THEN
248                BEGIN                          ;be4
249                IF (PDIN:DIN_VAL != PREV) THEN GOSUB CLR
250                NXT = PSTAT
251                IF (PPASS) & (PSTAT = 2) THEN NXT = 0
252                LAST = OR(LAST,NXT)
253                END                          ;en4
254                ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0
255                END                          ;en1
256                STEP PDIN BY 1
257                STEP PTYPE BY 1
258                STEP PREV BY 1
259                STEP PSTAT BY 1
260                STEP PPASS BY 1
261                STEP PSKID BY 1
262 0155      NEXT ILOOP
263
264      ;
265      ;
266      ;
267      ;
268 0156      ASSIGN PAI = 'PI-1A-1'
269 0157      ASSIGN PSKID = SKID11
270 0158      ASSIGN PREV = FRV11
271 0159      ASSIGN PSTAT = INT11
272 0160      ASSIGN PTYPE = TYPE11
273
274 0161      JOP:   FOR JLOOP = 11 TO 14
275                IF !SELECT() THEN GOTO AGAIN

```

15:37:09 24-Feb-95

Sequence Instruction Compiler Listing

```

276 0162      IF (PTYPE) THEN
277      BEGIN
278      BITVAL = PAI:AI_ALMCT
279 0163      IF AND(BITVAL,16) THEN
280      BEGIN
281 0164      IF AND(BITVAL,15) THEN
282      BEGIN
283 0165      IF AND(BITVAL,32) THEN NXT = 2
284      ELSE
285      BEGIN
286      IF PSKID != BITVAL THEN
287      BEGIN
288      IF BITVAL < 3 THEN FALMON = 1
289      LET PSKID = BITVAL
290      PDEV_DEV_TMR0 = 2
291      NXT = 1
292      ELSE NXT = 1
293      END
294 0174      END
295      ELSE
296      BEGIN
297      LET = 0
298      LET PSKID = 0
299      END
300      ELSE
301      BEGIN
302      NXT = 0
303      LET PSKID = 0
304 0177      END
305 0178      ANALOG = OR(ANALOG,NXT)
306      END
307 0179      STEP PSKID BY 1
308      STEP PAI BY 1
309 0180      STEP PREV BY 1
310 0181      STEP PSAT BY 1
311 0182      STEP PTYP BY 1
312 0183      STEP PTYP BY 1
313 0184      NEXT JLOOP
314 0185      END ANALOG RED AND YELLOWS
315      ;
316      ;
317      ;
318      ;
319      ;
320      ;
321      ;
322      ;
323      ;
324      ;
325      ;
326      ;
327      ;
328      ;
329      ;
330      ;
331      ;
332      ;
333      ;
334      ;
335      ;
336      ;
337      ;
338      ;
339      ;
340      ;
341      ;
342      ;
343      ;
344      ;
345      ;
346      ;
347      ;
348      ;
349      ;
350      ;
351      ;
352      ;
353      ;
354      ;
355      ;
356      ;
357      ;
358      ;
359      ;
360      ;
361      ;
362      ;
363      ;
364      ;
365      ;
366      ;
367      ;
368      ;
369      ;
370      ;
371      ;
372      ;
373      ;
374      ;
375      ;
376      ;
377      ;
378      ;
379      ;
380      ;
381      ;
382      ;
383      ;
384      ;
385      ;
386      ;
387      ;
388      ;
389      ;
390      ;
391      ;
392      ;
393      ;
394      ;
395      ;
396      ;
397      ;
398      ;
399      ;
400      ;
401      ;
402      ;
403      ;
404      ;
405      ;
406      ;
407      ;
408      ;
409      ;
410      ;
411      ;
412      ;
413      ;
414      ;
415      ;
416      ;
417      ;
418      ;
419      ;
420      ;
421      ;
422      ;
423      ;
424      ;
425      ;
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432      ;
433      ;
434      ;
435      ;
436      ;
437      ;
438      ;
439      ;
440      ;
441      ;
442      ;
443      ;
444      ;
445      ;
446      ;
447      ;
448      ;
449      ;
450      ;
451      ;
452      ;
453      ;
454      ;
455      ;
456      ;
457      ;
458      ;
459      ;
460      ;
461      ;
462      ;
463      ;
464      ;
465      ;
466      ;
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473      ;
474      ;
475      ;
476      ;
477      ;
478      ;
479      ;
480      ;
481      ;
482      ;
483      ;
484      ;
485      ;
486      ;
487      ;
488      ;
489      ;
490      ;
491      ;
492      ;
493      ;
494      ;
495      ;
496      ;
497      ;
498      ;
499      ;
500      ;
501      ;
502      ;
503      ;
504      ;
505      ;
506      ;
507      ;
508      ;
509      ;
510      ;
511      ;
512      ;
513      ;
514      ;
515      ;
516      ;
517      ;
518      ;
519      ;
520      ;
521      ;
522      ;
523      ;
524      ;
525      ;
526      ;
527      ;
528      ;
529      ;
530      ;
531      ;
532      ;
533      ;
534      ;
535      ;
536      ;
537      ;
538      ;
539      ;
540      ;
541      ;
542      ;
543      ;
544      ;
545      ;
546      ;
547      ;
548      ;
549      ;
550      ;
551      ;
552      ;
553      ;
554      ;
555      ;
556      ;
557      ;
558      ;
559      ;
560      ;
561      ;
562      ;
563      ;
564      ;
565      ;
566      ;
567      ;
568      ;
569      ;
570      ;
571      ;
572      ;
573      ;
574      ;
575      ;
576      ;
577      ;
578      ;
579      ;
580      ;
581      ;
582      ;
583      ;
584      ;
585      ;
586      ;
587      ;
588      ;
589      ;
590      ;
591      ;
592      ;
593      ;
594      ;
595      ;
596      ;
597      ;
598      ;
599      ;
600      ;
601      ;
602      ;
603      ;
604      ;
605      ;
606      ;
607      ;
608      ;
609      ;
610      ;
611      ;
612      ;
613      ;
614      ;
615      ;
616      ;
617      ;
618      ;
619      ;
620      ;
621      ;
622      ;
623      ;
624      ;
625      ;
626      ;
627      ;
628      ;
629      ;
630      ;
631      ;
632      ;
633      ;
634      ;
635      ;
636      ;
637      ;
638      ;
639      ;
640      ;
641      ;
642      ;
643      ;
644      ;
645      ;
646      ;
647      ;
648      ;
649      ;
650      ;
651      ;
652      ;
653      ;
654      ;
655      ;
656      ;
657      ;
658      ;
659      ;
660      ;
661      ;
662      ;
663      ;
664      ;
665      ;
666      ;
667      ;
668      ;
669      ;
670      ;
671      ;
672      ;
673      ;
674      ;
675      ;
676      ;
677      ;
678      ;
679      ;
680      ;
681      ;
682      ;
683      ;
684      ;
685      ;
686      ;
687      ;
688      ;
689      ;
690      ;
691      ;
692      ;
693      ;
694      ;
695      ;
696      ;
697      ;
698      ;
699      ;
700      ;
701      ;
702      ;
703      ;
704      ;
705      ;
706      ;
707      ;
708      ;
709      ;
710      ;
711      ;
712      ;
713      ;
714      ;
715      ;
716      ;
717      ;
718      ;
719      ;
720      ;
721      ;
722      ;
723      ;
724      ;
725      ;
726      ;
727      ;
728      ;
729      ;
730      ;
731      ;
732      ;
733      ;
734      ;
735      ;
736      ;
737      ;
738      ;
739      ;
740      ;
741      ;
742      ;
743      ;
744      ;
745      ;
746      ;
747      ;
748      ;
749      ;
750      ;
751      ;
752      ;
753      ;
754      ;
755      ;
756      ;
757      ;
758      ;
759      ;
760      ;
761      ;
762      ;
763      ;
764      ;
765      ;
766      ;
767      ;
768      ;
769      ;
770      ;
771      ;
772      ;
773      ;
774      ;
775      ;
776      ;
777      ;
778      ;
779      ;
780      ;
781      ;
782      ;
783      ;
784      ;
785      ;
786      ;
787      ;
788      ;
789      ;
790      ;
791      ;
792      ;
793      ;
794      ;
795      ;
796      ;
797      ;
798      ;
799      ;
800      ;
801      ;
802      ;
803      ;
804      ;
805      ;
806      ;
807      ;
808      ;
809      ;
810      ;
811      ;
812      ;
813      ;
814      ;
815      ;
816      ;
817      ;
818      ;
819      ;
820      ;
821      ;
822      ;
823      ;
824      ;
825      ;
826      ;
827      ;
828      ;
829      ;
830      ;
831      ;
832      ;
833      ;
834      ;
835      ;
836      ;
837      ;
838      ;
839      ;
840      ;
841      ;
842      ;
843      ;
844      ;
845      ;
846      ;
847      ;
848      ;
849      ;
850      ;
851      ;
852      ;
853      ;
854      ;
855      ;
856      ;
857      ;
858      ;
859      ;
860      ;
861      ;
862      ;
863      ;
864      ;
865      ;
866      ;
867      ;
868      ;
869      ;
870      ;
871      ;
872      ;
873      ;
874      ;
875      ;
876      ;
877      ;
878      ;
879      ;
880      ;
881      ;
882      ;
883      ;
884      ;
885      ;
886      ;
887      ;
888      ;
889      ;
890      ;
891      ;
892      ;
893      ;
894      ;
895      ;
896      ;
897      ;
898      ;
899      ;
900      ;
901      ;
902      ;
903      ;
904      ;
905      ;
906      ;
907      ;
908      ;
909      ;
910      ;
911      ;
912      ;
913      ;
914      ;
915      ;
916      ;
917      ;
918      ;
919      ;
920      ;
921      ;
922      ;
923      ;
924      ;
925      ;
926      ;
927      ;
928      ;
929      ;
930      ;
931      ;
932      ;
933      ;
934      ;
935      ;
936      ;
937      ;
938      ;
939      ;
940      ;
941      ;
942      ;
943      ;
944      ;
945      ;
946      ;
947      ;
948      ;
949      ;
950      ;
951      ;
952      ;
953      ;
954      ;
955      ;
956      ;
957      ;
958      ;
959      ;
960      ;
961      ;
962      ;
963      ;
964      ;
965      ;
966      ;
967      ;
968      ;
969      ;
970      ;
971      ;
972      ;
973      ;
974      ;
975      ;
976      ;
977      ;
978      ;
979      ;
980      ;
981      ;
982      ;
983      ;
984      ;
985      ;
986      ;
987      ;
988      ;
989      ;
990      ;
991      ;
992      ;
993      ;
994      ;
995      ;
996      ;
997      ;
998      ;
999      ;
1000      ;

```

```

331 0197      GOSUB ANLG
332 0198      ASSIGN PAI = 'TI-AMBS-1'
333 0199      GOSUB ANLG
334 0200      GOSUB ANLG
335 0201      GOSUB ANLG
336 0202      GOSUB ANLG
337 0203      ASSIGN PAI = 'TI-AMBS-2'
338 0204      GOSUB ANLG
339 0205      GOSUB ANLG
340 0206      GOSUB ANLG
341 0207      GOSUB ANLG
342          ;
343          ;
344          ;
345 0208      MARK STEP "BELOW",0,0
346 0209      SORT = 0
347 0210      FTRIP4 = 0
348 0212      IF LAST > 3 THEN PRINT ADI," LAMP VARIABLE IS GREATER THAN 3!"
349 0213      CURENT = OR(LAST,WHITE)
350 0214      CURENT = OR(CURENT,ANALOG)
351 0216      IF AND(CURENT,1) THEN LMPVAR = 2
352 0218          ELSE IF AND(CURENT,2) THEN LMPVAR = 1
353 0219          ELSE LMPVAR = 0
354 0220      LAMP ADI,SKDKEY,LMPVAR
355 0221      MARK STEP "BOTTOM",0,0
356 0222      NOCRIT
357 0224      IF AND(UTILITIES:DEV_FLG,64) THEN PUTMANL UTILITIES
358 0225      GOTO AGAIN
359          ;
360          ;
361          ;
362          ;
363          ;
364          ; ANLG: IF !SELECT() THEN GOTO AGAIN
365 0226  ANLG: IF (PTYPE = 1) THEN                                ; If some alarm
366          BEGIN                                                    ;be1
367 0227  BITVAL = PAI:AI_ALMCT
368          IF AND(BITVAL,16) THEN                                    ; If inhibited
369          ;old BEGIN
370          ;old BLET PSTAT = 3
371          ;old DLEAKO = 1
372          ;old GOSUB ACKK
373          ;old END
374 0228  IF !AND(BITVAL,16)) THEN                                     ; If not inhibited
375          BEGIN                                                    ;be2
376 0230      IF PSTAT = 3 THEN LET PSTAT = 0
377 0231      IF (PREV) THEN                                           ; If rev acting, LOLO, or LO
378          BEGIN                                                    ;be3
379 0232      IF (PSTAT = 0) THEN                                       ; If not previously alarmed
380          BEGIN                                                    ;be4
381 0234      IF (PAI:AI_INVL <= PALARM) THEN GOSUB ALM
382          END                                                        ;en4
383          ELSE
384          BEGIN                                                    ;be5
385 0235      IF (PSTAT = 1) THEN                                       ; If previously alarmed

```

```

386          BEGIN                                ;be6
387          IF (FTRIP4) THEN GOSUB ACK1            ; If ACK flag set, then ack it
388          IF PTYPE !=1 THEN IF AND(PAI:A1_ALMCT,32) THEN GOSUB ACK1
389          END                                     ;en6
390          IF (PAI:A1_INVL >= PALARM + PAI:A1_DEAD) THEN GOSUB CLR
391          END                                     ;en5
392          ELSE                                   ;en3
393          BEGIN                                ;be10 ; If fail-safe, HIRI, or HI
394          IF (PSTAT = 0) THEN
395          BEGIN                                ;be11
396          IF (PAI:A1_INVL >= PALARM) THEN GOSUB ALM
397          END                                     ;en11
398          ELSE
399          BEGIN                                ;be12
400          IF (PSTAT = 1) THEN
401          BEGIN                                ;be13
402          IF (FTRIP4) THEN GOSUB ACK1
403          IF PTYPE != 1 THEN IF AND(PAI:A1_ALMCT,32) THEN GOSUB ACK1
404          END                                     ;en13
405          IF (PAI:A1_INVL <= PALARM - PAI:A1_DEAD) THEN GOSUB CLR
406          END                                     ;en12
407          ELSE
408          NEXT = PSTAT
409          IF (PPASS) & (PSTAT = 2) THEN NEXT = 0
410          END                                     ;en2
411          ELSE
412          BEGIN
413          LET PSTAT = 3
414          NEXT = 0
415          END
416          WHITE = OR(WHITE,NXT)
417          END                                     ;en1
418          ELSE LET PSTAT = 0
419          STEP PALARM BY 1
420          STEP PSTAT BY 1
421          STEP PREV BY 1
422          STEP PPASS BY 1
423          STEP PTYPE BY 1
424          ILOOP = ILOOP + 1
425          RETURN
426          0267
427          ;
428          ;
429          ;
430          ;
431          BEGIN CLEAR, ALARM, ACKNOWLEDGE, MESSAGE, ERROR, AND TERMINATE SUBROUTINES:
432          0268 CLR: LET PSTAT = 0
433          ;
434          ; If not white, then unack:
435          ;
436          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
437          LET FCLEAR = 1
438          ;
439          ; To Set-Up For LOLO to LO or HIRI to HI Transition:
440          ;

```



```

441 0274      IF ILOOP = 18 THEN LET INT19 = 0      ; If TI-AMBS-1 LOLO, Reset LO
442 0276      IF ILOOP = 21 THEN LET INT20 = 0      ; If TI-AMBS-1 HIHI, Reset HI
443 0278      IF ILOOP = 22 THEN LET INT23 = 0      ; If TI-AMBS-2 LOLO, Reset LO
444 0280      IF ILOOP = 25 THEN LET INT24 = 0      ; If TI-AMBS-2 HIHI, Reset HI
445 0281      GOSUB ACKK
446 0282      RETURN
447      ;
448 0283  ALM:  LET PSTAT = 1
449      ;
450      ; If not white, then unack:
451      ;
452 0286      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
453      ;
454      ; Turn audible alarm on if white or yellow:
455      ;
456 0287      FALMON = 1
457 0289      IF AND(PTYPE,3) THEN FACK = 1
458 0290      CRIT
459 0291      PDEV:DEV_TMO = 2
460 0292      GOSUB ACKK
461 0293      RETURN
462      ;
463 0294  ACK1:  LET PSTAT = 2      ; Ack the alarm
464 0295      FACK = 1      ; Set ACK flag for C program
465      ;
466      ; Begin Multiple AI White Alarms on One EPN:
467      ; PI-IA-1:
468 0297  ACKK:  IF(INT15 & INT15 < 3)THEN LET LEAK4 = INT15 + 2      ; LO
469 0299      ELSE IF(INT16 & INT16 < 3)THEN LET LEAK4 = INT16 + 4      ; HI
470 0301      ELSE IF INT15 = 3 THEN LET LEAK4 = 9
471 0302      ELSE LET LEAK4 = 0
472      ;
473      ; PI-PA-1:
474 0304      IF(INT17 & INT17 < 3)THEN LET LEAK1 = INT17 * 4      ; HI
475 0306      ELSE IF INT17 = 3 THEN LET LEAK1 = 9
476 0307      ELSE LET LEAK1 = 0
477      ;
478      ; TI-AMBS-1:
479 0309      IF(INT18 & INT18 < 3)THEN LET LEAK2 = INT18      ; LOLO
480 0311      ELSE IF(INT19 & INT19 < 3)THEN LET LEAK2 = INT19 + 2      ; LO
481 0313      ELSE IF(INT21 & INT21 < 3)THEN LET LEAK2 = INT21 + 6      ; HIHI
482 0315      ELSE IF(INT20 & INT20 < 3)THEN LET LEAK2 = INT20 + 4      ; HI
483 0317      ELSE IF INT18 = 3 THEN LET LEAK2 = 9
484 0318      ELSE LET LEAK2 = 0
485      ;
486      ; TI-AMBS-2:
487 0320      IF(INT22 & INT22 < 3) THEN LET LEAK3 = INT22      ; LOLO
488 0322      ELSE IF (INT23 & INT23 < 3) THEN LET LEAK3 = INT23 + 2      ; LO
489 0324      ELSE IF(INT25 & INT25 < 3) THEN LET LEAK3 = INT25 + 6      ; HIHI
490 0326      ELSE IF( INT24 & INT24 < 3) THEN LET LEAK3 = INT24 + 4      ; HI
491 0328      ELSE IF INT22 = 3 THEN LET LEAK3 = 9
492 0329      ELSE LET LEAK3 = 0
493      ;
494      ; End Multiple AI White Alarms
495      ;

```

```

496      ;old      IF DLEAKO = 1 THEN                                ; If Inhibited
497      ;old      BEGIN
498      ;old      BLET DLEAKO = 0
499      ;old      RETURN                                           ; Skip message routine
500      ;old      END
501      ;
502      ;
503      ;
504      0331      ALARM TYPER STRNGS:
505      0333      IF ILOOP = 0 THEN STRNGO = "PROCESS AIR PRSS LOW " ; PSLCPE1/2
506      0335      IF ILOOP = 1 THEN STRNGO = "INSTRUMENT AIR PRES LO" ; PSL-IA-2
507      0337      IF ILOOP = 2 THEN STRNGO = "AIR COMPRESSOR 1 OFF " ; YS-CPE1-1
508      0339      IF ILOOP = 3 THEN STRNGO = "AIR COMPRESSOR 2 OFF " ; YS-CPE2-1
509      0341      IF ILOOP = 4 THEN STRNGO = "24 VDC POWER FAIL " ; JXS-PWR-1
510      0343      IF ILOOP = 5 THEN STRNGO = "DIESEL GENRATR TROUBL" ; YS-DIEGEN
511      0345      IF ILOOP = 6 THEN STRNGO = "EMRGNCY DIESELGEN FAIL" ; YS-GEN-1
512      0347      IF ILOOP = 7 THEN STRNGO = "EMERG PWR XFR SWITCH ON" ; ZS-PWR-1
513      0349      IF ILOOP = 8 THEN STRNGO = "UPS AC INPUT FAIL" ; JXS-PWR-3
514      0351      IF ILOOP = 9 THEN STRNGO = "UPS BATTERY LOW" ; YSL-UPS-1
515      0353      IF ILOOP = 10 THEN STRNGO = "UPS MBS BYPASS POSITION" ; ZS-UPS-1
516      0355      IF ILOOP = 15 THEN STRNGO = "INSTRUMENT AIR HDR PRESS LOW" ; PI-IA-1
517      0357      IF ILOOP = 16 THEN STRNGO = "INSTRUMENT AIR HDR PRESS HI" ; PI-IA-1
518      0359      IF ILOOP = 17 THEN STRNGO = "PROCESS AIR HDR PRESS HIGH" ; PI-PA-1
519      0361      IF ILOOP = 18 THEN STRNGO = "COLD JUNCTION SENSOR 1 LOLO" ; T1-AMBS-1
520      0363      IF ILOOP = 19 THEN STRNGO = "COLD JUNCTION SENSOR 1 LO " ; T1-AMBS-1
521      0365      IF ILOOP = 20 THEN STRNGO = "COLD JUNCTION SENSOR 1 HI " ; T1-AMBS-1
522      0367      IF ILOOP = 21 THEN STRNGO = "COLD JUNCTION SENSOR 1 HIHI" ; T1-AMBS-1
523      0369      IF ILOOP = 22 THEN STRNGO = "COLD JUNCTION SENSOR 2 LOLO" ; T1-AMBS-2
524      0371      IF ILOOP = 23 THEN STRNGO = "COLD JUNCTION SENSOR 2 LO " ; T1-AMBS-2
525      0373      IF ILOOP = 24 THEN STRNGO = "COLD JUNCTION SENSOR 2 HI " ; T1-AMBS-2
526      0375      IF ILOOP = 25 THEN STRNGO = "COLD JUNCTION SENSOR 2 HIHI" ; T1-AMBS-2
527      0377      IF ILOOP = 26 THEN STRNGO = "UTILBATCH.SEQ START " ; STAR
528      0379      IF ILOOP = 27 THEN STRNGO = "UTILBATCH IN WRONG UNIT "
529      0380      IF ILOOP = 28 THEN STRNGO = "UTILBATCH TERMINATING "
530      0380      ;
531      0380      AMGS: IF PTYPE = 1 THEN
532      0380      BEGIN
533      0380      ;
534      0380      ; If white alarm, print:
535      0382      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT AD1,"WHITE ALARM %p",STRNGO
536      0383      IF(FACK) THEN
537      0383      BEGIN
538      0383      ;
539      0383      ; If white, print it is ack and reset the flag:
540      0383      ;
541      0384      PRINT AD1,"%p ALARM ACKNOWLEDGE",STRNGO
542      0385      LET FACK = 0
543      0385      END
544      0385      END
545      0385      ;
546      0385      ; If CLEAR, print it is clear and reset flag:
547      0385      ;
548      0386      IF (FCLEAR) THEN
549      0386      BEGIN
550      0387      PRINT AD1,"%p ALARM CLEAR",STRNGO

```

```

551 0388      LET FCLEAR = 0
552          END
553          ;
554          ; If STARTUP, print startup message:
555          ;
556 0389      IF (FSTART) THEN
557          BEGIN
558 0390      LET FSTART = 0
559 0391      PRINT ADI,"%p",STRNGO
560          END
561 0392      RETURN
562          ;
563          ; If ERROR - print message, start timer, and set flag:
564          ;
565 0393  ERRSUB: IF ERRN != ERRNUM() THEN
566          BEGIN
567 0394      TIMEROM STIMEO
568 0395      TIMERCLR STIMEO
569 0396      LET FSWO = 1
570 0397      LET ERRN = ERRNUM()
571 0398      LET ERRL = ERRLIN()
572 0399      PRINT ADI,"ERROR # %i AT LINE %i",errn,errl
573          END
574 0400      ERSCLR
575 0401      GOTO AGAIN
576          ;
577          ; If FABORT = 1, then abort program:
578          ;
579          ; TERM: LET FABORT = 0
580 0402  TERM: LET STRNGO = "UTILBATCH TERMINATED"
581 0403      LET ILOOP = STAR + 2
582 0404      LET FSTART = 1
583 0405      LET STRNGO = "TERM"
584 0406      GOSUB ACKK
585          END

```

Total Errors Detected 0   Total Warnings 0
Sequence Program Size = 8032 Bytes, 502 Pages
29 Forward Reference(s)

AMUBATCH

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:32:28 24-Feb-95

Page 1

```

1      ;      A M U B A T C H . S E Q      VERSION 2
2      ;
3      ;      Function:      Sequence program for Lamp function,
4      ;                      message display on Graphic number 019,
5      ;                      and printer messages.
6      ;
7      ;      Written By      : R. W. Kersey
8      ;      Date           : 05-31-92
9      ;      Facility       : 242-A Evaporator Bldg.
10     ;      Machine        : VAX 4104-A
11     ;      Op. System     : VAX/VMS 5.5-2AH
12     ;      D/3 Version    : 6.3-2 BASELINE
13     ;      Compiler       : SIC
14     ;      Build command  : @([320,20]AMUBATCH
15     ;
16     ;      UNIT = 20, SKDKEY = 45
17     ;
18     ;      Assoc. UPF:      [300,1] _PROCESS.____ PRCESS.LST
19     ;      Assoc. RPF:      [300,1] _242A.____ 242A.LST
20     ;      SDB:            [300,1]PCMOA.SDB PCMOASDB.CMD
21     ;
22     ;      POINTER      SDB VAR.      FUNCTION
23     ;      -----
24     ;      PSTAT      INTn      Current EPN alarm state is an integer tied to
25     ;                      graphic flashing string of value:
26     ;                      .      (n=0-100)      0 = previously not alarmed
27     ;                      .                      1 = previously alarmed, not ack
28     ;                      .                      2 = previously alarmed, is ack
29     ;                      .                      3 = alarm is inhibited
30     ;                      .                      4 = input is off scan
31     ;                      .                      5 = input is bad
32     ;      .      LEAKn      Special INT variable for multiple white alarms on an EPN
33     ;      PDIN      'EPN'      ---
34     ;      PTYPE      TYPEn      alarm type 0 = none, 1 = white, 2 = yellow,
35     ;                      3 = yellow and yellow, 4 = red,
36     ;                      6 = yellow and red
37     ;      PREV      FRVn      Reverse acting (not fail safe) flag (1=set)
38     ;      PSKID      SKIDn      Used for analog alarms
39     ;      PPASS      FPASn      Turn lamp off when alarm is ACKnowledged
40     ;      PALARM      FALMn      Analog alarm flag
41     ;      ---      FNING      Unused error flag
42     ;
43     ;      MODIFICATION LOG:
44     ;
45     ;      30-DEC-92      Fixed white alarm print message.      M. Teats
46     ;
47     ;      08-JUN-93      Added comments and baselined for
48     ;                      D/3 version 3.3. CCN# 93JUN01-01.      R. Persin
49     ;
50     ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
51     ;                      system. Rebaselined document. Removed
52     ;                      obsolete Select() statment.
53     ;
54     ;      END MOD LOG
55     ;

```

```

56      UNIT      AMU
57      ;
58      0000      NOP
59      0001      STAR = 33
60      0002      ASSIGN PDEV = 'AMU-ALARM' ; Alarm device
61      ;
62      0003      LET FSN0 = 0 ; Error message repeater
63      0004      TIMEROFF STIME0 ; Alarm timer off
64      0005      TIMERCLR STIME0 ; Alarm timer clear
65      0006      LET STRNG0 = " " ; Blank 30 character string
66      0007      LET ERRN = 0 ; Set error number to zero
67      0008      LET ERRL = 0 ; Set error linbe to zero
68      ;
69      0009      LET FTRIP4 = 0 ; FTRIP = 1 when audible horn is to be turned on
70      0010      LET DLEAK3 = 0 ; Flag for inhibited alarms
71      ;
72      ;
73      ; SEQUENCE UNIT AND PCM CHECK
74      0011      GETUNIT TRIP ; Which sequence unit is program running in
75      0012      IF TRIP != 20 THEN ; Abort if runnin9 in wring seq. unit
76      BEGIN
77      LET FSTART = 1
78      LET ILOOP = STAR + 1
79      GOSUB ACKK
80      LET FSTART = 1
81      0017      PRINT ADI," >>>>> ATTEMPT TO RUN AMUBATCH IN WRONG UNIT"
82      0018      GOTO TERM
83      END
84      ; IF !SELECT() THEN GOTO AGAIN
85      ;
86      ;
87      ; END SEQUENCE UNIT AND PCM CHECK
88      ;
89      ; BEGIN INITIALIZATION :
90      0019      INIT: LET SKDKEY = 45 ; SKID Key
91      0020      LET ADI = 0 ; Alarm destination index
92      0021      LET FABORT = 0 ; If FABORT = 1 then abort program
93      0022      LET FSTART = 0 ; Program start flag = 0
94      0023      LAMP ADI,SKDKEY,0 ; Turn off the annunciator key
95      0024      LET PDEV:DEV_TMO0 = 1 ; Set alarm device to non-alarm state
96      0025      IF AND(PDEV:DEV_ALMCT,16) THEN ; If alarm device is inhibited , remove the inhibit
97      0026      LET PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
98      0027      IF AND(PDEV:DEV_ALMCT,32) THEN ; If alarm device is ack, clear it
99      0028      LET PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
100     ;
101     ; First, set all flags and integers
102     ;
103     0029      ASSIGN PDIN = 'YS-A-E101' ; Assign pointer to first DIN
104     0030      ASSIGN PPASS = FPAS0
105     0031      ASSIGN PREV = FRV0 ; Reverse acting flag
106     0032      ASSIGN PSTAT = INTO
107     0033      ASSIGN PTYPE = TYPE0
108     0034      ASSIGN PSKID = SKID0 ; stePer
109     ;
110     0035      FOR ILOOP = 0 TO 25

```

```

111 0036      LET PPASS = 0
112 0037      LET PREV = 0
113 0039      IF ILOOP < 6 THEN LET PREV = 1      ; For white DINs
114 0041      IF PSTAT > 5 THEN PSTAT = 0          ; Reset current status if undefined
115 0042      LET PSKID = 0                        ; Default previous status to no alarm
116 0043      LET PTYPE = 0                        ; Default to no alarm
117 0044      IF ILOOP < 6 THEN
118          BEGIN
119 0045          LET PTYPE = 2                    ; Default to yellow alarm
120 0046          IF AND(PDIN:DIN_DFLG,3) THEN    ; Check for alarms if defined (must be a red or yellow alarm)
121              BEGIN
122 0047              LET PTYPE = 2                ; Default to yellow alarm
123 0049              IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1 ; Reverse acting
124 0051              IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4 ; Set to critical red alarm
125                  END
126          END
127 0052      ELSE PTYPE = 6                    ; Red and Yellow, future use
128 0054      SKP:  IF ILOOP < 6 THEN STEP PDIN BY 1 ; Point to next DIN
129 0055          STEP PPASS BY 1
130 0056          STEP PREV BY 1
131 0057          STEP PSTAT BY 1
132 0058          STEP PSKID BY 1
133 0059          STEP PTYPE BY 1
134 0060      NEXT ILOOP
135          ;
136          ;      Set alarm type if not red or yellow
137          ;
138 0061      LET TYPE0 = 1                    ; YS-A-E101
139 0062      LET TYPE1 = 1                    ; YS-P-E101
140 0063      LET TYPE2 = 1                    ; YS-A-E102
141 0064      LET TYPE3 = 1                    ; YS-P-E102
142 0065      LET TYPE4 = 1                    ; YS-A-E104
143 0066      LET TYPE5 = 1                    ; YS-P-E104
144 0067      LET TYPE6 = 1                    ; TI-E101-2 Hi
145 0068      LET TYPE8 = 1                    ; WFI-E101 Lo-Lo
146 0069      LET TYPE9 = 1                    ; WFI-E101 Lo
147 0070      LET TYPE10 = 0                   ; DI-E101-2
148 0071      LET TYPE12 = 0                   ; FI-E102-1
149 0072      LET TYPE14 = 1                    ; WFI-E102 Lo-Lo
150 0073      LET TYPE15 = 1                    ; WFI-E102 Lo
151 0074      LET TYPE16 = 0                   ; FQI-E102
152 0075      LET TYPE17 = 1                    ; TI-E104-2 Hi
153 0076      LET TYPE19 = 1                    ; WFI-E104 Lo-Lo
154 0077      LET TYPE20 = 1                    ; WFI-E104 Lo
155 0078      LET TYPE21 = 0                   ; DI-E104-2
156 0079      LET TYPE23 = 0                   ; FQI-E104
157 0080      LET TYPE24 = 0                   ; FI-E104-2
158 0081      LET TYPE25 = 0                   ; FIC-E102
159          ;
160 0082      LET FRV8 = 1                    ; WFI-E101 Lo-Lo
161 0083      LET FRV9 = 1                    ; WFI-E101 Lo
162 0084      LET FRV14 = 1                    ; WFI-E102 Lo-Lo
163 0085      LET FRV15 = 1                    ; WFI-E102 Lo
164 0086      LET FRV19 = 1                    ; WFI-E104 Lo-Lo
165 0087      LET FRV20 = 1                    ; WFI-E104 Lo

```

```

166      ;
167      ;
168      ;           Analog white alarm limits
169      0088      LET AI0 = 150                      ; TI-E101-2 Hi
170      0089      LET AI2 = 15                      ; WFI-E101 Lo-Lo
171      0090      LET AI3 = 25                      ; WFI-E101 Lo
172      0091      LET AI8 = 5                       ; WFI-E102 Lo-Lo
173      0092      LET AI9 = 12                      ; WFI-E102 Lo
174      0093      LET AI11 = 170                   ; TI-E104-2 Hi
175      0094      LET AI13 = 5                      ; WFI-E104 Lo-Lo
176      0095      LET AI14 = 12                   ; WFI-E104 Lo
177      ;
178      ;           END INITIALIZING
179      ;
180      ;
181      ;           SEND PROGRAM START MESSAGE
182      ;
183      ;
184      0096      LET ANALOG = 0
185      0097      LET ILOOP = STAR
186      0098      LET FSTART = 1
187      0099      GOSUB ACKK
188      ;
189      ;
190      ;           BEGIN PROGRAM LOOP
191      ;
192      ;
193      0100  AGAIN: MARK STEP "AGAIN",0,0
194      0101      ON ERROR GOSUB ERRSUB
195      ;           ; Terminate program if abort flag is set
196      ;           IF (FABORT) THEN GOTO TERM
197      0102      WAITUNTIL SELECT()
198      ;           WAIT 1
199      ;
200      0103      IF (FSNO) THEN                      ; If alarm flag is set
201      0104      IF STIME0 > 60 THEN                ; If alarm flag active for 1 minute
202      ;           BEGIN                             ; Reset timer and turn off flag
203      ;           TIMEROFF STIME0
204      ;           TIMERCLR STIME0
205      ;           LET FSNO = 0
206      ;           LET ERRL = 0
207      ;           END
208      ;
209      ;
210      ;           If AMU ALARM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
211      ;
212      ;
213      0110      IF (PDEV:DEV_TMPO = 2) & AND(PDEV:DEV_ALMCT,32) THEN
214      ;           BEGIN
215      0111      PDEV:DEV_TMPO = 1                      ; Put AMU - ALARM is ack state
216      0112      LET FTRIP4 = 1                      ; Set for unit alarm ack, system to graphic
217      0113      LET ACK = 1                          ; Set for unit alarm ack, graphic to system
218      0114      CRIT
219      ;           END
220      ;

```



```

221      ;
222      ;
223      ;      B E G I N   D I G I T A L S
224      ;
225      ;      Function:      Recognize digital red, yellow, white alarms
226      ;                      and associate with graphic string.  Turn
227      ;                      on alarm ACK device, turn on Lamp, and
228      ;                      flash message on graphic.
229      0115      ASSIGN PDIN = 'YS-A-E101'
230      0116      ASSIGN PTYPE = TYPE0                      ; ALARM TYPE
231      0117      ASSIGN PREV = FRV0                        ; REVERSE ACTING FLAG
232      0118      ASSIGN PSTAT = INTO                      ; INTEGERS TIED TO GRAPHIC FLASHING STRINGS
233      0119      ASSIGN PPASS = FPASO
234      0120      LET LAST = 0
235      0121      LET NXT = 0
236      0122      LMPVAR = 0
237      0123      BITVAL = 0
238      ;
239      0124      LOP:  FOR ILOOP = 0 TO 5
240      ;
241      0125      IF ISELECT() THEN GOTO AGAIN
242      ;
243      0127      IF PTYPE > 0 THEN                          ; If some type of alarm
244      ;
245      0129      BEGIN                                     ; b1
246      0131      IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3
247      0132      IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5
248      0133      IF IAND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4
249      0134      IF PSTAT < 3 THEN                          ; If not inhibited
250      ;
251      0136      BEGIN                                     ; b2
252      0138      IF PSTAT = 1 THEN
253      0141      BEGIN                                     ; b3
254      0143      IF PTYPE != 1 THEN
255      0144      IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
256      0145      IF (FTRIP4) THEN GOSUB ACK1
257      0146      END                                         ; e3
258      0147      IF PSTAT = 0 THEN IF (PDIN:DIN_VAL = PREV) THEN GOSUB ALM
259      0148      IF (PSTAT) THEN
260      0150      BEGIN                                     ; b4
261      0152      IF (PDIN:DIN_VAL != PREV) THEN GOSUB CLR
262      0153      NXT = PSTAT
263      0154      IF (PPASS) & (PSTAT = 2) THEN NXT = 0
264      0155      LAST = OR(LAST,NXT)
265      0156      END                                         ; e4
266      0157      ELSE IF IAND(PDIN:DIN_ALMCT,16)&IAND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN
267      0158      LET PSTAT = 0
268      0159      END                                         ; e1
269      0160      STEP PDIN BY 1
270      0161      STEP PTYPE BY 1
271      0162      STEP PREV BY 1
272      0163      STEP PSTAT BY 1
273      0164      STEP PPASS BY 1
274      ;
275      ;
276      ;
277      ;
278      ;
279      ;
280      ;
281      ;
282      ;
283      ;
284      ;
285      ;
286      ;
287      ;
288      ;
289      ;
290      ;
291      ;
292      ;
293      ;
294      ;
295      ;
296      ;
297      ;
298      ;
299      ;
300      ;
301      ;
302      ;
303      ;
304      ;
305      ;
306      ;
307      ;
308      ;
309      ;
310      ;
311      ;
312      ;
313      ;
314      ;
315      ;
316      ;
317      ;
318      ;
319      ;
320      ;
321      ;
322      ;
323      ;
324      ;
325      ;
326      ;
327      ;
328      ;
329      ;
330      ;
331      ;
332      ;
333      ;
334      ;
335      ;
336      ;
337      ;
338      ;
339      ;
340      ;
341      ;
342      ;
343      ;
344      ;
345      ;
346      ;
347      ;
348      ;
349      ;
350      ;
351      ;
352      ;
353      ;
354      ;
355      ;
356      ;
357      ;
358      ;
359      ;
360      ;
361      ;
362      ;
363      ;
364      ;
365      ;
366      ;
367      ;
368      ;
369      ;
370      ;
371      ;
372      ;
373      ;
374      ;
375      ;
376      ;
377      ;
378      ;
379      ;
380      ;
381      ;
382      ;
383      ;
384      ;
385      ;
386      ;
387      ;
388      ;
389      ;
390      ;
391      ;
392      ;
393      ;
394      ;
395      ;
396      ;
397      ;
398      ;
399      ;
400      ;
401      ;
402      ;
403      ;
404      ;
405      ;
406      ;
407      ;
408      ;
409      ;
410      ;
411      ;
412      ;
413      ;
414      ;
415      ;
416      ;
417      ;
418      ;
419      ;
420      ;
421      ;
422      ;
423      ;
424      ;
425      ;
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432      ;
433      ;
434      ;
435      ;
436      ;
437      ;
438      ;
439      ;
440      ;
441      ;
442      ;
443      ;
444      ;
445      ;
446      ;
447      ;
448      ;
449      ;
450      ;
451      ;
452      ;
453      ;
454      ;
455      ;
456      ;
457      ;
458      ;
459      ;
460      ;
461      ;
462      ;
463      ;
464      ;
465      ;
466      ;
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473      ;
474      ;
475      ;
476      ;
477      ;
478      ;
479      ;
480      ;
481      ;
482      ;
483      ;
484      ;
485      ;
486      ;
487      ;
488      ;
489      ;
490      ;
491      ;
492      ;
493      ;
494      ;
495      ;
496      ;
497      ;
498      ;
499      ;
500      ;
501      ;
502      ;
503      ;
504      ;
505      ;
506      ;
507      ;
508      ;
509      ;
510      ;
511      ;
512      ;
513      ;
514      ;
515      ;
516      ;
517      ;
518      ;
519      ;
520      ;
521      ;
522      ;
523      ;
524      ;
525      ;
526      ;
527      ;
528      ;
529      ;
530      ;
531      ;
532      ;
533      ;
534      ;
535      ;
536      ;
537      ;
538      ;
539      ;
540      ;
541      ;
542      ;
543      ;
544      ;
545      ;
546      ;
547      ;
548      ;
549      ;
550      ;
551      ;
552      ;
553      ;
554      ;
555      ;
556      ;
557      ;
558      ;
559      ;
560      ;
561      ;
562      ;
563      ;
564      ;
565      ;
566      ;
567      ;
568      ;
569      ;
570      ;
571      ;
572      ;
573      ;
574      ;
575      ;
576      ;
577      ;
578      ;
579      ;
580      ;
581      ;
582      ;
583      ;
584      ;
585      ;
586      ;
587      ;
588      ;
589      ;
590      ;
591      ;
592      ;
593      ;
594      ;
595      ;
596      ;
597      ;
598      ;
599      ;
600      ;
601      ;
602      ;
603      ;
604      ;
605      ;
606      ;
607      ;
608      ;
609      ;
610      ;
611      ;
612      ;
613      ;
614      ;
615      ;
616      ;
617      ;
618      ;
619      ;
620      ;
621      ;
622      ;
623      ;
624      ;
625      ;
626      ;
627      ;
628      ;
629      ;
630      ;
631      ;
632      ;
633      ;
634      ;
635      ;
636      ;
637      ;
638      ;
639      ;
640      ;
641      ;
642      ;
643      ;
644      ;
645      ;
646      ;
647      ;
648      ;
649      ;
650      ;
651      ;
652      ;
653      ;
654      ;
655      ;
656      ;
657      ;
658      ;
659      ;
660      ;
661      ;
662      ;
663      ;
664      ;
665      ;
666      ;
667      ;
668      ;
669      ;
670      ;
671      ;
672      ;
673      ;
674      ;
675      ;
676      ;
677      ;
678      ;
679      ;
680      ;
681      ;
682      ;
683      ;
684      ;
685      ;
686      ;
687      ;
688      ;
689      ;
690      ;
691      ;
692      ;
693      ;
694      ;
695      ;
696      ;
697      ;
698      ;
699      ;
700      ;
701      ;
702      ;
703      ;
704      ;
705      ;
706      ;
707      ;
708      ;
709      ;
710      ;
711      ;
712      ;
713      ;
714      ;
715      ;
716      ;
717      ;
718      ;
719      ;
720      ;
721      ;
722      ;
723      ;
724      ;
725      ;
726      ;
727      ;
728      ;
729      ;
730      ;
731      ;
732      ;
733      ;
734      ;
735      ;
736      ;
737      ;
738      ;
739      ;
740      ;
741      ;
742      ;
743      ;
744      ;
745      ;
746      ;
747      ;
748      ;
749      ;
750      ;
751      ;
752      ;
753      ;
754      ;
755      ;
756      ;
757      ;
758      ;
759      ;
760      ;
761      ;
762      ;
763      ;
764      ;
765      ;
766      ;
767      ;
768      ;
769      ;
770      ;
771      ;
772      ;
773      ;
774      ;
775      ;
776      ;
777      ;
778      ;
779      ;
780      ;
781      ;
782      ;
783      ;
784      ;
785      ;
786      ;
787      ;
788      ;
789      ;
790      ;
791      ;
792      ;
793      ;
794      ;
795      ;
796      ;
797      ;
798      ;
799      ;
800      ;
801      ;
802      ;
803      ;
804      ;
805      ;
806      ;
807      ;
808      ;
809      ;
810      ;
811      ;
812      ;
813      ;
814      ;
815      ;
816      ;
817      ;
818      ;
819      ;
820      ;
821      ;
822      ;
823      ;
824      ;
825      ;
826      ;
827      ;
828      ;
829      ;
830      ;
831      ;
832      ;
833      ;
834      ;
835      ;
836      ;
837      ;
838      ;
839      ;
840      ;
841      ;
842      ;
843      ;
844      ;
845      ;
846      ;
847      ;
848      ;
849      ;
850      ;
851      ;
852      ;
853      ;
854      ;
855      ;
856      ;
857      ;
858      ;
859      ;
860      ;
861      ;
862      ;
863      ;
864      ;
865      ;
866      ;
867      ;
868      ;
869      ;
870      ;
871      ;
872      ;
873      ;
874      ;
875      ;
876      ;
877      ;
878      ;
879      ;
880      ;
881      ;
882      ;
883      ;
884      ;
885      ;
886      ;
887      ;
888      ;
889      ;
890      ;
891      ;
892      ;
893      ;
894      ;
895      ;
896      ;
897      ;
898      ;
899      ;
900      ;
901      ;
902      ;
903      ;
904      ;
905      ;
906      ;
907      ;
908      ;
909      ;
910      ;
911      ;
912      ;
913      ;
914      ;
915      ;
916      ;
917      ;
918      ;
919      ;
920      ;
921      ;
922      ;
923      ;
924      ;
925      ;
926      ;
927      ;
928      ;
929      ;
930      ;
931      ;
932      ;
933      ;
934      ;
935      ;
936      ;
937      ;
938      ;
939      ;
940      ;
941      ;
942      ;
943      ;
944      ;
945      ;
946      ;
947      ;
948      ;
949      ;
950      ;
951      ;
952      ;
953      ;
954      ;
955      ;
956      ;
957      ;
958      ;
959      ;
960      ;
961      ;
962      ;
963      ;
964      ;
965      ;
966      ;
967      ;
968      ;
969      ;
970      ;
971      ;
972      ;
973      ;
974      ;
975      ;
976      ;
977      ;
978      ;
979      ;
980      ;
981      ;
982      ;
983      ;
984      ;
985      ;
986      ;
987      ;
988      ;
989      ;
990      ;
991      ;
992      ;
993      ;
994      ;
995      ;
996      ;
997      ;
998      ;
999      ;
1000     ;

```

```

276      ;
277      ;
278      ;       BEGIN  ANALOGS RED AND YELLOW :
279      ;
280      ;       Function:      Recognize analog red and yellow alarms
281      ;                      and associate with graphic strings. Turn
282      ;                      on alarm ACK device, turn on lamp, and
283      ;                      flash message on graphic.
284      0157      ANALOG = 0
285      0158      ASSIGN PSKID = SKID6
286      0159      ASSIGN PAI = 'I1-E101-2'
287      0160      FOR JLOOP = 6 TO 25
288      ;          IF !SELECT() THEN GOTO AGAIN          ; If some alarm exists
289      0161      IF (PTYPE) THEN
290      ;          BEGIN                                  ; b1
291      0162      BITVAL = PAI:AI_ALMCT
292      0163      IF !AND(BITVAL,16) THEN                ; If not inhibited
293      ;          BEGIN                                  ; b2
294      0164      IF AND(BITVAL,15) THEN                ; If alarm exists and is current
295      ;          BEGIN                                  ; b3
296      0166      IF AND(BITVAL,32) THEN NXT = 2        ; If alarm is ack
297      ;          ELSE                                  ; If alarm is unack
298      ;          BEGIN                                  ; b4
299      0167      IF PSKID != BITVAL THEN
300      ;          BEGIN                                  ; b5
301      0168      LET PSKID = BITVAL
302      0170      IF BITVAL < 3 THEN FALMON = 1
303      0171      PDEV:DEV_TMPO = 2
304      0172      NXT = 1
305      ;          END                                  ; e5
306      0173      ELSE NXT = 1
307      ;          END                                  ; e4
308      ;          END                                  ; e3
309      ;          ELSE                                  ; Else there is no alarm
310      ;          BEGIN                                  ; b6
311      0174      NXT = 0
312      0175      LET PSKID = 0
313      ;          END                                  ; e6
314      ;          END                                  ; e2
315      ;          ELSE                                  ; Else alarm is inhibited
316      ;          BEGIN                                  ; b7
317      0176      NXT = 0
318      0177      LET PSKID = 0
319      ;          END                                  ; e7
320      0178      ANALOG = OR(ANALOG,NXT)
321      ;          END                                  ; e1
322      0179      STEP PREV BY 1
323      0180      STEP PSTAT BY 1
324      0181      STEP PSKID BY 1
325      0182      STEP PTYPE BY 1
326      0183      IF (JLOOP!=8)&(JLOOP!=9)&(JLOOP!=14)&(JLOOP!=15)&(JLOOP!=19)&(JLOOP!=20) THEN
327      ;          BEGIN
328      0184      STEP PAI BY 1                      ; Step PAI if JLOOP != 8, 9, 14, 15, 19, OR 20
329      ;          END
330      0185      NEXT JLOOP

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:32:28 24-Feb-95

Page 7

```

331      ;
332      ;
333      ;           E N D   A N A L O G S
334      ;
335      ;           B E G I N   A N A L O G   W H I T E   A L A R M S
336      ;
337      ;           Function:      Recognize AI White alarms, associate
338      ;                           with graphic string.  Turn on alarm
339      ;                           ACK device, turn on Lamp, and flash
340      ;                           message on graphic.
341      0186      WHITE = 0
342      0187      ASSIGN PALARM = A10
343      0188      ASSIGN PA1 = 'TI-E101-2'
344      0189      ASSIGN PREV = FRV6
345      0190      ASSIGN PSTAT = INT6
346      0191      ASSIGN PTYPE = TYPE6
347      0192      IN1 = 6
348      0193      IN2 = 6
349      0194      GOSUB ANLG
350      0195      ASSIGN PA1 = 'WFI-E101'
351      0196      IN1 = 7
352      0197      IN2 = 9
353      0198      GOSUB ANLG
354      0199      ASSIGN PA1 = 'DI-E101-2'
355      0200      IN1 = 10
356      0201      IN2 = 12
357      0202      GOSUB ANLG
358      0203      ASSIGN PA1 = 'WFI-E102'
359      0204      IN1 = 13
360      0205      IN2 = 15
361      0206      GOSUB ANLG
362      0207      ASSIGN PA1 = 'FQI-E102'
363      0208      IN1 = 16
364      0209      IN2 = 17
365      0210      GOSUB ANLG
366      0211      ASSIGN PA1 = 'WFI-E104'
367      0212      IN1 = 18
368      0213      IN2 = 20
369      0214      GOSUB ANLG
370      0215      ASSIGN PA1 = 'DI-E104-2'
371      0216      IN1 = 21
372      0217      IN2 = 25
373      0218      GOSUB ANLG
374      ;
375      ;
376      ;           E N D   A N A L O G   W H I T E   A L A R M S
377      0220      IF (INT6 & INT6 < 3) THEN LET LEAK3 = INT6 + 4      ; TI-E101-2 Hi
378      0222      ELSE IF INT6 = 3 THEN LET LEAK3 = 9      ; TI-E101-2 INHA
379      0223      ELSE LET LEAK3 = 0      ; No alarm
380
381      0225      IF (INT8 & INT8 < 3) THEN LET LEAK0 = INT8      ; WFI-E101 Lo-Lo
382      0227      ELSE IF (INT9 & INT9 < 3) THEN LET LEAK0 = INT9 + 2      ; WFI-E101 Lo
383      0229      ELSE IF INT8 = 3 THEN LET LEAK0 = 9      ; WFI-E101 INHA
384      0230      ELSE LET LEAK0 = 0      ; No alarm
385

```

```

386 0232      IF (INT14 & INT14 < 3) THEN LET LEAK1 = INT14          ; WFI-E102 Lo-Lo
387 0234      ELSE IF (INT15 & INT15 < 3) THEN LET LEAK1 = INT15 + 2 ; WFI-E102 Lo
388 0236      ELSE IF INT14 = 3 THEN LET LEAK1 = 9                ; WFI-E102 INHA
389 0237      ELSE LET LEAK1 = 0                                    ; No alarm
390
391 0239      IF (INT17 & INT17 < 3) THEN LET LEAK4 = INT17 + 4      ; TI-E104-2 Hi
392 0241      ELSE IF INT17 = 3 THEN LET LEAK4 = 9                ; TI-E104-2 INHA
393 0242      ELSE LET LEAK4 = 0                                    ; No alarm
394
395 0244      IF (INT19 & INT19 < 3) THEN LET LEAK2 = INT19          ; WFI-E104 Lo-Lo
396 0246      ELSE IF (INT20 & INT20 < 3) THEN LET LEAK2 = INT20 + 2 ; WFI-E104 Lo
397 0248      ELSE IF INT19 = 3 THEN LET LEAK2 = 9                ; WFI-E104 INHA
398 0249      ELSE LET LEAK2 = 0                                    ; No alarm
399
400 0250      MARK STEP "BELOW",0,0
401 0251      SORT = 0
402 0253      IF LAST > 3 THEN PRINT AD1," LAMP VARIABLE IS GREATER THAN 3!"
403 0254      CURENT = OR(LAST,WHITE)
404 0255      CURENT = OR(CURENT,ANALOG)
405 0257      IF AND(CURENT,1) THEN LMPVAR = 2
406 0259      ELSE IF AND(CURENT,2) THEN LMPVAR = 1
407 0260      ELSE LMPVAR = 0
408 0261      LAMP AD1,SKDKEY,LMPVAR
409 0262      MARK STEP "BOTTOM",0,0
410 0263      NOCRIT
411 0265      IF AND('AMU-ALARM':DEV_FLG,64)THEN PUTMANL 'AMU-ALARM'
412 0267      IF AND('P-E-101':DEV_FLG,64)THEN PUTMANL 'P-E-101'
413 0269      IF AND('A-E-101':DEV_FLG,64)THEN PUTMANL 'A-E-101'
414 0271      IF AND('P-E-102':DEV_FLG,64)THEN PUTMANL 'P-E-102'
415 0273      IF AND('A-E-102':DEV_FLG,64)THEN PUTMANL 'A-E-102'
416 0275      IF AND('P-E-104':DEV_FLG,64)THEN PUTMANL 'P-E-104'
417 0277      IF AND('A-E-104':DEV_FLG,64)THEN PUTMANL 'A-E-104'
418 0278      GOTO AGAIN
419
420      ;
421      ;
422      ;
423      ;
424      ;
425      ;
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432      ;
433 0279  ANLG:  FOR ILOOP = IN1 TO IN2
434      ;      IF ISELECT() THEN GOTO AGAIN
435 0280      IF PTYPE = 1 THEN
436      BEGIN
437 0281      BITVAL = PAI:AI_ALMCT
438      ;
439 0282      IF AND(BITVAL,16) THEN
440      BEGIN

```

BEGIN ANALOG WHITE SUBROUTINE

ILOOP	EPN	ALARM LEVEL
6	TI-E101-2	Hi
8	WFI-E101	Lo-Lo
9	WFI-E101	Lo
14	WFI-E102	Lo-Lo
15	WFI-E102	Lo
17	TI-E104-2	Hi
19	WFI-E104	Lo-Lo
20	WFI-E104	Lo

; b0

; If inhibited

```

441 0283      LET PSTAT = 3
442 0284      DLEAK3 = 1
443 0285      GOSUB ACKK
444      END
445
446 0286      IF (AND(BITVAL,16)) THEN          ; If not inhibited and alarm enabled
447      BEGIN                                ; b1
448 0287      IF (PREV) THEN                    ; If rev acting, Lo-Lo, or Lo
449      BEGIN                                ; b2
450 0288      IF (PSTAT = 0) THEN
451      BEGIN                                ; b3
452 0290      IF (PAI:AI_MEAS <= PALARM) THEN GOSUB ALM
453      END                                  ; e3
454      ELSE
455      BEGIN                                ; b4
456 0291      IF PSTAT = 1 THEN
457      BEGIN                                ; b5
458      IF (FTRIP4) THEN GOSUB ACK1
459 0296      IF PTYP1 = 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
460      END                                  ; e5
461 0298      IF (PAI:AI_MEAS >= PALARM + PAI:AI_DEAD) THEN GOSUB CLR
462      END                                  ; e4
463      END                                  ; e2
464      ELSE                                  ; If fail safe, Hi-Hi, or Hi
465      BEGIN                                ; b2.1
466 0299      IF PSTAT = 0 THEN
467      BEGIN                                ; b6
468 0301      IF (PAI:AI_MEAS >= PALARM) THEN GOSUB ALM
469      END                                  ; e6
470      ELSE
471      BEGIN                                ; b7
472 0302      IF PSTAT = 1 THEN
473      BEGIN                                ; b8
474 0304      IF (FTRIP4) THEN GOSUB ACK1
475 0307      IF PTYP1 != 1 THEN IF AND(PAI:AI_ALMCT,32) THEN GOSUB ACK1
476      END                                  ; e8
477 0309      IF (PAI:AI_MEAS <= PALARM - PAI:AI_DEAD) THEN GOSUB CLR
478      END                                  ; e7
479      END                                  ; e2.1
480 0310      NXT = PSTAT
481 0312      IF (PPASS) & (PSTAT = 2) THEN NXT = 0
482      END                                  ; e1
483      ELSE
484      BEGIN                                ; b9
485 0313      LET PSTAT = 3
486 0314      NXT = 0
487      END                                  ; e9
488 0315      WHITE = OR(WHITE,NXT)
489      END                                  ; e0
490 0316      ELSE LET PSTAT = 0
491 0317      STEP PALARM BY 1
492 0318      IF (ILOOP!=7)&(ILOOP!=8)&(ILOOP!=13)&(ILOOP!=14)&(ILOOP!=18)&(ILOOP!=19) THEN
493      BEGIN
494 0319      STEP PAI BY 1                      ; Step PAI if ILOOP != 7, 8, 13, 14, 18, OR 19
495      END

```

```

496 0320      STEP PSTAT BY 1
497 0321      STEP PTYPE BY 1
498 0322      STEP PREV BY 1
499 0323      STEP PPASS BY 1
500 0324      NEXT ILOOP
501 0325      RETURN
502      ;
503      ;
504      ;
505      ;
506      ;
507      ;
508 0326  CLR:  LET PSTAT = 0
509 0328      IF ILOOP = 8 THEN LET INT9 = 0          ; Clear Hi alarm for Hi-Hi to Hi transition
510 0330      IF ILOOP = 14 THEN LET INT15 = 0         ; Clear Hi alarm for Hi-Hi to Hi transition
511 0332      IF ILOOP = 19 THEN LET INT20 = 0         ; Clear Hi alarm for Hi-Hi to Hi transition
512      ;
513      ;
514      ;
515 0334      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN
516 0335          PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
517 0336          LET FCLEAR = 1                        ; Clear flag
518 0337          GOSUB ACKK                            ; Print message
519 0338          RETURN
520      ;
521 0339  ALM:  LET PSTAT = 1                          ; Int set to one
522      ;
523      ;
524      ;
525 0341      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN
526 0342          PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
527      ;
528      ;
529      ;
530 0343      PDEV:DEV_TMP0 = 2
531 0345      IF AND(PTYPE,3) THEN FALMON = 1
532 0346      LAMP ADI,SKDKEY,2
533 0347      CRIT
534 0348      GOSUB ACKK                                ; Print Message
535 0349      RETURN
536      ;
537 0350  ACK1: LET PSTAT = 2                          ; Ack the alarm
538 0351      LET FACK = 1                            ; Set ack flag for C program
539      ;
540      ;
541      ;
542      ;
543 0352  ACKK: IF DLEAK3 = 1 THEN                      ; If inhibited
544      BEGIN
545 0353      LET DLEAK3 = 0
546 0354      RETURN                                    ; Skip message routine
547      END
548      ;
549      ;
550      ;

```

```

551      ;          A L A R M   T Y P E R   S T R I N G S
552      ;
553      ;          For DIN white (there are no DIN R or Y), AI white,
554      ;          and program status.
555      ;
556 0356      IF ILOOP = 0 THEN STRNGO = "ELUANT AGITATOR ON"      " ; YS-A-E101
557 0358      IF ILOOP = 1 THEN STRNGO = "ELUANT PUMP ON"         " ; YS-P-E101
558 0360      IF ILOOP = 2 THEN STRNGO = "ANTIFOAM AGITATOR ON"   " ; YS-A-E102
559 0362      IF ILOOP = 3 THEN STRNGO = "ANTIFOAM PUMP ON"       " ; YS-P-E102
560 0364      IF ILOOP = 4 THEN STRNGO = "DECON AGITATOR ON"      " ; YS-A-E104
561 0366      IF ILOOP = 5 THEN STRNGO = "DECON PUMP ON"          " ; YS-P-E104
562 0368      IF ILOOP = 6 THEN STRNGO = "TK-E-101 LIQUID TEMP HI" " ; TI-E101-2 HI
563 0370      IF ILOOP = 8 THEN STRNGO = "TK-E-101 WT FACTOR LO-LO" " ; WFI-E101 Lo-Lo
564 0372      IF ILOOP = 9 THEN STRNGO = "TK-E-101 WT FACTOR LO"   " ; WFI-E101 Lo
565 0374      IF ILOOP = 14 THEN STRNGO = "TK-E-102 WT FACTOR LO-LO" " ; WFI-E102 Lo-Lo
566 0376      IF ILOOP = 15 THEN STRNGO = "TK-E-102 WT FACTOR LO"   " ; WFI-E102 Lo
567 0378      IF ILOOP = 17 THEN STRNGO = "TK-E-104 LIQUID TEMP HI" " ; TI-E104-2 HI
568 0380      IF ILOOP = 19 THEN STRNGO = "TK-E-104 WT FACTOR LO-LO" " ; WFI-E104 Lo-Lo
569 0382      IF ILOOP = 20 THEN STRNGO = "TK-E-104 WT FACTOR LO"   " ; WFI-E104 Lo
570 0384      IF ILOOP = 33 THEN STRNGO = "START AMUBATCH PGM"      "
571 0386      IF ILOOP = 34 THEN STRNGO = "AMUBATCH WRONG UNIT"     "
572 0388      IF ILOOP = 35 THEN STRNGO = "TERMINATE AMUBATCH"      "
573      ;
574      ; If acknowledging alarm, then:
575      ;
576 0389      MSG:      IF PTYPE = 1 THEN
577                      BEGIN
578                      ;
579                      ; If white alarm, print:
580                      ;
581 0391                      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNGO
582 0392                      IF (FACK) THEN
583                          BEGIN
584 0393                          PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
585 0394                          LET FACK = 0
586                          END
587                      END
588      ;
589      ; If CLEAR, print it is clear and reset flag:
590      ;
591 0395      IF (FCLEAR) THEN
592          BEGIN
593 0396          PRINT ADI,"%p ALARM CLEAR",STRNGO
594 0397          LET FCLEAR = 0
595          END
596      ;
597      ; If STARTUP, print startup message:
598      ;
599 0398      IF (FSTART) THEN
600          BEGIN
601 0399          LET FSTART = 0
602 0400          PRINT ADI,"%p",STRNGO
603          END
604 0401      RETURN
605      ;

```

```

606      ; if ERROR - print message, start timer, and set flag
607      ;
608      ERRSUB: IF ERRM (= ERRNUM()) THEN
609      BEGIN
610      TIMERON STIMEO
611      TIMERCLR STIMEO
612      LET FSHO = 1
613      LET ERRM = ERRNUM()
614      LET ERRL = ERRLIN()
615      PRINT ADI,"ERROR # %I AT LINE %I",errm,errl
616      END
617      ERSCLR
618      GOTO AGAIN
619
620      ; if FABORT = 1, then abort program
621      ;
622      ; TERM: LET FABORT = 0
623      ; TERM: LET LOOP = STAR + 2
624      ; TERM: LET FSTART = 1
625      ; GOSUB ACKK
626      ; END

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 8016 Bytes, 501 Pages
26 Forward Reference(s)

LERFBATCH

```

1      ; L E R F B A T C H . S E Q                                VERSION 4
2      ;
3      ; Function:      Sequence program for Lamp function,
4      ;               message display on Graphic number 024,
5      ;               and printer messages.
6      ;
7      ; Written By    : R. N. Kersey
8      ; Date         : 05-31-92
9      ; Facility     : 242-A Evaporator Bldg.
10     ; Machine      : VAX 4104-A
11     ; Op. System   : VAX/VMS 5.5-2AH
12     ; D/3 Version  : 6.3-2 BASELINE
13     ; Compiler     : SIC
14     ; Build command : @[320,20]LERFBATCH
15     ;
16     ; UNIT = 21, SKDKEY = 40
17     ;
18     ; Assoc. UPF:    [300,1]_PROCESS.____ PROCESS.LST
19     ; Assoc. RPF:    [300,1]_242A.      242A.LST
20     ; SDB:          [300,1]PCMOA.SDB    PCMOASDB.CMD
21     ;
22     ; POINTER      SDB VAR.      FUNCTION
23     ; -----
24     ; PSTAT      INTn      Current EPN alarm state is an integer tied to
25     ;               graphic flashing string of value:
26     ;               (n=0-100)  0 = previously not alarmed
27     ;               .          1 = previously alarmed, not ack
28     ;               .          2 = previously alarmed, is ack
29     ;               .          3 = alarm is inhibited
30     ;               .          4 = input is off scan
31     ;               .          5 = input is bad
32     ;               LEAKn      Special INT variable for multiple white alarms on an EPN
33     ;               'EPN'      ---
34     ; PTYPE      TYPEn      alarm type 0 = none, 1 = white, 2 = yellow,
35     ;               .          3 = yellow and yellow, 4 = red,
36     ;               .          6 = yellow and red
37     ; PREV      FRVn      Reverse acting (not fail safe) flag (1=set)
38     ; PSKID      SKIDn     Used for analog alarms
39     ; PPASS      FPASn     Turn lamp off when alarm is ACKnowledged
40     ; PALARM     FALMn     Analog alarm flag
41     ; ---          FNING    Unused error flag
42     ;
43     ; MODIFICATION LOG:
44     ;
45     ; 05-NOV-92      Deleted LERF valve MOV-43-1 associated          M. Teats
46     ;               logic and DINs (search for WAS).
47     ;
48     ; 30-NOV-92      Added SA5, SA6, SA7 for line leaks;          M. Teats
49     ;               renumbered SA0 to SA7; revised leak logic.
50     ;
51     ; 16-DEC-92      Added analog alarm code, and comments.      M. Teats
52     ;
53     ; 30-DEC-92      Fixed white alarm print message.          M. Teats
54     ;
55     ; 29-JAN-93      Added YS-PLC-1.                              M. Teats

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

12:24:06 03-Apr-96

Page 2

```

56      ;
57      ;
58      ; 01-MAR-93      Change LDS-BSW42, 43, 44 to fail-safe      M. Teats
59      ;      (alarms on voltage absent in MUX,
60      ;      change FRV_ from 1 to 0).
61      ;
62      ; 11-MAR-93      Modified SA_ (SA0...SAB) variable code for      M. Teats
63      ;      as-found.
64      ;
65      ; 08-JUN-93      Added comments and baselined for      R. Persin
66      ;      D/3 version 3.3. CCN# 93JUN01-01.
67      ;
68      ; 11-JUL-94      Added LERF DINs LSH60A12, LSH60A13      R. Kersey
69      ;      YSETFCOM, and YS60A56
70      ;
71      ; 15-DEC-94      Revised header to indicate the new VMS      J. Berger
72      ;      system. Rebaselined document. Removed
73      ;      obsolete Select() document.
74      ;
75      ; 10-APR-95      Deleted LERF gate alarms ZS-105PS1 and      J. Berger
76      ;      ZS-105PS2 per ECN 620391 and WP EE-95-
77      ;      0006. (JFB01)
78      ;
79      ; 03-APR-96      Deleted the ETF signals. ECN 627478      J. Berger
80      ;      MCS-96-010.
81      ;
82      ; END MOD LOG
83      ;
84      ; UNIT      LERF
85      ; 0000      NOP
86      ; 0001      STAR = 26      ; Start message
87      ; 0002      LET SKDKEY = 40      ; SKID Key
88      ; 0003      LET UNITT = 21      ; SEQ Unit
89      ; 0004      ASSIGN PDEV = 'LERF-ALM' ; Alarm device
90      ; 0005      ASSIGN PTYPE = TYPE25
91      ; 0006      LET ADI = 0      ; Alarm destination index
92      ; 0007      LET FSN0 = 0      ; No alarm yet
93      ; 0008      TIMEROFF STIME0      ; Alarm timer off
94      ; 0009      TIMERCLK STIME0      ; Alarm timer clear
95      ; 0010      STRNG0 = " "      ; Blank 30 character string
96      ; 0011      LET ERRN = 0      ; Set error number to zero
97      ; 0012      LET ERRL = 0      ; Set error line to zero
98      ; 0013      LET FTRIP4 = 0      ; FTRIP = 1 when audible horn is to be turned on
99      ; 0014      GETUNIT TRIP      ; Which sequence unit is program running in
100     ; 0015      IF TRIP != UNITT THEN      ; Abort if running in wrong seq. unit
101     ;      BEGIN
102     ;      FSTART = 1
103     ;      ILOOP = STAR + 1
104     ;      PTYPE = 0
105     ;      GOSUB ACKK
106     ;      GOTO TERM
107     ;      END
108     ;      IF !SELECT() THEN GOTO AGAIN
109     ;
110     ; BEGIN INITIALIZING

```

```

111      ;      DEVICES:
112      ;      LERF-ALM
113      ;      LERFILK
114      ;      DIGITALS:      ILOOP#
115      ;      YS-P-43      0
116      ;      LDS-A1      1
117      ;      Y-P-44      2
118      ;      ZS-105PS1    3      DELETED JFB01
119      ;      ZS-105PS2    4      DELETED JFB01
120      ;      YS-P-42      5
121      ;      LDS-BSN42    6
122      ;      LDS-BSN43    7
123      ;      LDS-BSN44    8
124      ;      LXS-LCH42    9
125      ;      LXS-BSN42   10
126      ;      LXS-LCH43   11
127      ;      LXS-BSN43   12
128      ;      LXS-LCH44   13
129      ;      LXS-BSN44   14
130      ;      YS-PLC-1    15
131      ;      LSH60A12     16      DELETED
132      ;      LSH60A13     17      DELETED
133      ;      YSETFCOM     18      DELETED
134      ;      YS60A56     19      DELETED
135      ;      ANALOGS:
136      ;      LDI-A1      20
137      ;      LI-LCH42     21
138      ;      LI-BSN42     22
139      ;      LI-LCH43     23
140      ;      LI-BSN43     24
141      ;      LI-LCH44     25
142      ;      LI-BSN44     26
143      ;
144 0021      STRNG0 = "INIT"
145      INIT:  LET FABORT = 0      ; if FABORT = 1 then abort program
146 0023      LET FSTART = 0      ; Program start flag = 0
147 0024      LAMP AD1,SKDKEY,0    ; Turn off the annunciator key
148 0025      LET LSG0 = " "      ; Clear the string variables
149 0026      LET LSG1 = " "
150 0027      LET LSG2 = " "
151 0028      LET LSG3 = " "
152 0029      LET LSG4 = " "
153 0030      LET PDEV:DEV_TMPO = 1      ; Set alarm device to non-alarm state
154 0031      IF AND(PDEV:DEV_ALMCT,16) THEN      ; If alarm device is inhibited, remove the inhibit
155 0032      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
156 0033      IF AND(PDEV:DEV_ALMCT,32) THEN      ; If alarm device is ack, clear it
157 0034      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
158      ;
159      ;      First, set all flags and integers:
160      ;
161 0035      ASSIGN PDIN = 'YS-P-43'      ; Assign pointer to first DIN
162 0036      ASSIGN PSTAT = INTO
163 0037      ASSIGN PTYPE = TYPE0
164 0038      ASSIGN PREV = FRV0
165 0039      ASSIGN PPASS = FPAS0

```

```

166 0040      ASSIGN PALARM = FALMO
167 0041      ASSIGN PSKID = SKID0
168
169 0042      ;      FOR ILOOP = 0 TO 26
170 0043      LET PSKID = 0
171 0044      LET PPASS = 0
172 0045      LET PREV = 0
173 0046      LET PTYPE = 0
174 0048      IF PSTAT > 5 THEN PSTAT = 0
175 0049      PALARM = 0
176 0050      IF ILOOP = 3 THEN
177          BEGIN
178          STEP PDIN BY 1
179 0052      GOTO SKP
180          END
181 0053      IF ILOOP = 4 THEN
182          BEGIN
183 0054      STEP PDIN BY 1
184 0055      GOTO SKP
185          END
186 0057      IF ILOOP = 16 THEN GOTO SKP
187 0059      IF ILOOP = 17 THEN GOTO SKP
188 0061      IF ILOOP = 18 THEN GOTO SKP
189 0063      IF ILOOP = 19 THEN GOTO SKP
190
191 0064      IF ILOOP <= 15 THEN
192          BEGIN
193 0065      IF AND(PDIN:DIN_DFLG,3) THEN
194          BEGIN
195 0066      LET PTYPE = 2
196 0068      IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1
197 0070      IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4
198          END
199 0072      SKP1:  IF ILOOP < 15 THEN STEP PDIN BY 1
200          END
201 0073      ELSE PTYPE = 0
202 0074      SKP:  STEP PSTAT BY 1
203 0075      STEP PTYPE BY 1
204 0076      STEP PREV BY 1
205 0077      STEP PPASS BY 1
206 0078      STEP PALARM BY 1
207 0079      STEP PSKID BY 1
208 0080      NEXT ILOOP
209
210 0081      ;      TYPE0 = 1      ; YS-P-43
211 0082      TYPE1 = 2      ; LDS-A1
212 0083      TYPE2 = 1      ; YS-P-44
213      ;      TYPE3 = 1      ; ZS-105PS1      JFB01
214      ;      TYPE4 = 1      ; ZS-105PS2      JFB01
215 0084      TYPE5 = 1      ; YS-P-42
216 0085      TYPE6 = 2      ; LDS-BSN42
217 0086      TYPE7 = 2      ; LDS-BSN43
218 0087      TYPE8 = 2      ; LDS-BSN44
219 0088      TYPE9 = 4      ; LXS-LCM42
220 0089      TYPE10 = 4     ; LXS-BSN42

```

```

221 0090      TYPE11 = 4      ; LXS-LCH43
222 0091      TYPE12 = 4      ; LXS-BSN43
223 0092      TYPE13 = 4      ; LXS-LCH44
224 0093      TYPE14 = 4      ; LXS-BSN44
225 0094      TYPE15 = 4      ; YS-PLC-1
226      ;      TYPE16 = 2      ; LSH60A12      DELETED
227      ;      TYPE17 = 4      ; LAH60A13      DELETED
228      ;      TYPE18 = 4      ; YSETFCOM      DELETED
229      ;      TYPE19 = 2      ; YS60A56      DELETED
230
231 0095      TYPE20 = 2      ; LDI-A1
232      ;
233 0096      FRV0 = 1      ; YS-P-43
234 0097      FRV2 = 1      ; YS-P-44
235 0098      FRV5 = 1      ; YS-P-42
236      ;      FRV19 = 0      ; YS60A56      DELETED
237      ;
238 0099      ANALOG = 0
239      ;
240      ;      END      INITIALIZING
241      ;
242      ;      SEND      PROGRAM      START      MESSAGE
243      ;
244 0100      ILOOP = STAR
245 0101      FSTART = 1
246 0102      PTYPE = 0
247 0103      GOSUB ACKK
248 0104      ON ERROR GOSUB ERRSUB
249      ;
250      ;      BEGIN      PROGRAM      LOOP
251      ;
252 0105 AGAIN: MARK STEP "WTUNTL",0,0
253 0106      STRNG0 = "WAIT"
254      ;      WAITUNTIL SELECT() ; FABORT      ; Wait until micro is selected or program aborted
255      ;      IF (FABORT) THEN GOTO TERM      ; Terminate program if FABORT flag is set
256 0107      MARK STEP "RNNING",0,0
257 0108      WAIT 1
258 0109      FTRIP4 = 0
259      ;
260 0110      IF (FSNO) THEN      ; If alarm flag is set
261 0111      IF STIME0 > 60 THEN      ; If alarm flag active for 1 minute
262      BEGIN      ; Reset timer and turn off flag
263      TIMEROFF STIME0
264 0113      TIMERCLR STIME0
265 0114      LET FSNO = 0
266 0115      LET ERRN = 0
267 0116      LET ERRL = 0
268      END
269      ;
270      ;      IF LERF-ALM $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
271      ;
272 0117      IF (PDEV:DEV_TMP0 = 2) & AND(PDEV:DEV_ALMCT,32) THEN
273      BEGIN
274 0118      'LERF-ALM':DEV_TMP0 = 1      ; Put LERF-ALM in ACKNWLDG state
275 0119      LET FTRIP4 = 1      ; Set for Unit alarm acknowledgement, system to graphic

```

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

12:24:06 03-Apr-96

Page 6

```

276 0120      ACK = 1                      ; Set for Unit alarm acknowledgement, graphic to system
277          ;
278 0121      LAMP ADJ,SKDKEY,1
279          END
280          ;
281          ;
282          BEGIN DINs:
283 0122      ASSIGN PDIN = 'YS-P-43'
284 0123      ASSIGN PTYPE = TYPE0
285 0124      ASSIGN PREV = FRV0
286 0125      ASSIGN PSTAT = INTO
287 0126      ASSIGN PPASS = FPAS0
288 0127      LET CURENT = 0                ; Used to set lamp status
289 0128      LET LAST = 0
290 0129      LET ANALOG = 0
291 0130      LET NXT = 0
292 0131      LMPVAR = 0
293 0132      BITVAL = 0
294          ;
295 0133      LOP:  FOR ILOOP = 0 TO 15
296          ;
297 0135      IF !SELECT() THEN GOTO AGAIN
298 0137      IF ILOOP = 3 THEN GOTO LOP1
299 0138      IF ILOOP = 4 THEN GOTO LOP1
300          IF ( PTYPE > 0 ) THEN
301              BEGIN
302                  IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3      ;be1
303                  IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5      ; If EPN inhibited
304                  IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4      ; If EPN is bad input
305                  IF ( PSTAT < 3 ) THEN
306                      BEGIN
307                          IF PSTAT = 1 THEN
308                              BEGIN
309                                  IF PTYPE != 1 THEN
310                                      IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
311                                      IF (FTRIP4) THEN GOSUB ACK1
312                                  END
313                                  IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
314                                  IF (PSTAT) THEN
315                                      BEGIN
316                                          IF PDIN:DIN_VAL != PREV THEN GOSUB CLR
317                                          NXT = PSTAT
318                                          IF (PPASS) & (PSTAT = 2) THEN NXT = 0
319                                          LAST = OR(LAST,NXT)
320                                          END
321                                          ;en4
322                                          ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 0
323                                          ;en2
324                                          END
325                                          ;en1
326          LOP1:  IF ILOOP < 15 THEN STEP PDIN BY 1
327          STEP PTYPE BY 1
328          STEP PREV BY 1
329          STEP PSTAT BY 1
330          STEP PPASS BY 1
331          NEXT ILOOP
332          ;
333          ;
334          END DINs:

```

```

331      ;
332      ;      BEGIN ANALOG RED AND YELLOWS:
333      ;
334      0171      ANALOG = 0
335      0172      LET STRNG0 = "RUN"
336      0173      ASSIGN PAI = 'LDI-A1'
337      0174      ASSIGN PSKID = SKID20
338      0175      ASSIGN PTYPE = TYPE20
339      ;
340      0176      FOR JLOOP = 20 TO 26
341      ;
342      0177      IF !SELECT() THEN GOTO AGAIN
343      ;
344      0178      IF (PTYPE) THEN
345      ;
346      0179      BEGIN
347      ;
348      0180      BITVAL = PAI:A1 ALMCT
349      ;
350      0182      IF !AND(BITVAL,16) THEN
351      ;
352      0183      BEGIN
353      ;
354      0185      IF PSKID !=BITVAL THEN
355      ;
356      0186      BEGIN
357      ;
358      0188      IF BITVAL < 3 THEN FALMON = 1
359      ;
360      0189      LET PSKID = BITVAL
361      ;
362      0191      LET PDEV:DEV_TMPO = 2
363      ;
364      0192      NXT = 1
365      ;
366      0193      END
367      ;
368      0194      ELSE NXT = 1
369      ;
370      0195      END
371      ;
372      0196      ELSE
373      ;
374      0197      BEGIN
375      ;
376      0198      NXT = 0
377      ;
378      0199      LET PSKID = 0
379      ;
380      0200      END
381      ;
382      0201      ANALOG = OR(ANALOG,NXT)
383      ;
384      0202      END
385      ;
386      0203      STEP PSKID BY 1
387      0204      STEP PAI BY 1
388      0205      STEP PTYPE BY 1
389      0206      NEXT JLOOP
390      ;
391      ;
392      ;
393      0207      END ANALOG RED AND YELLOWS
394      ;
395      0208      MARK STEP "BELOW",0,0
396      0209      LET FTRIP4 = 0
397      0210      IF LAST > 3 THEN PRINT AD1," LAMP VAR IS GREATER THAN 3"
398      0211      CURENT = OR(LAST,ANALOG)

```



```

386 0205      IF AND(CURRENT,1) THEN LMPVAR = 2
387 0207      ELSE IF AND(CURRENT,2) THEN LMPVAR = 1
388 0208      ELSE LMPVAR = 0
389 0209      LAMP AD1,SKDKEY,LMPVAR      ; LMPVAR 0=off, 1=on, 2=blink
390 0210      MARK STEP "BOTTOM",0,0
391      NOCRIT
392 0212      IF AND('LERF-ALM':DEV_FLG,64) THEN PUTMANL 'LERF-ALM'
393
394      BEGIN LERF LEAK
395
396      Each LDE represents 0.32 mA of the 4-20 mA signal, or 2% with the
397      following exception. During ATPs, the percent increment between
398      leak detector zones was found to be 2.25 to 2.5%. The following
399      code corresponds to the following ATP findings:
400
401      Sensor      % Reading      mA reading
402      -----
403      LDE-01      0.61      4.08
404      LDE-02      2.98      4.46
405      LDE-03      5.36      4.84
406      LDE-04      7.92      5.25
407      LDE-05      10.23      5.62
408      LDE-06      12.73      6.02
409      LDE-07      15.61      6.48
410      LDE-08      18.23      6.90
411      LDE-09      20.23      7.29
412      LDE-10      22.23      7.63
413
414
415
416      Allow 1 % error for LDE trip. The SA_ variables, where
417      the underscore is a number from 0-7, are tied to graphic
418      indications of the alarmed leak detector zone.
419
420 0213      AI10 = 'LDI-A1':AI_INVL
421 0214      IF AI10 >= 25 THEN
422      BEGIN
423 0215      SA0 = 1      ; LDE-01
424 0216      SA1 = 1      ; LDE-02
425 0217      SA2 = 1      ; LDE-03
426 0218      SA3 = 1      ; LDE-04
427 0219      SA4 = 1      ; LDE-05
428 0220      SA5 = 1      ; LDE-06
429 0221      SA6 = 1      ; LDE-07
430 0222      SA7 = 1      ; LDE-08
431 0223      SB0 = 1      ; LDE-09
432 0224      SB1 = 1      ; LDE-10
433      END
434      ELSE
435      BEGIN
436 0226      IF (AI10 >= 0.0) & (AI10 < 1.8) THEN SA0 = 1      ; LDE-01
437 0227      ELSE SA0 = 0
438 0229      IF (AI10 >= 1.8) & (AI10 < 4.2) THEN SA1 = 1      ; LDE-02
439 0230      ELSE SA1 = 0
440 0232      IF (AI10 >= 4.2) & (AI10 < 6.6) THEN SA2 = 1      ; LDE-03

```

```

441 0233          ELSE SA2 = 0
442 0235          IF (A110 >= 6.6) & (A110 < 9.1) THEN SA3 = 1      ; LDE-04
443 0236          ELSE SA3 = 0
444 0238          IF (A110 >= 9.1) & (A110 < 11.5) THEN SA4 = 1      ; LDE-05
445 0239          ELSE SA4 = 0
446 0241          IF (A110 >= 11.5) & (A110 < 14.2) THEN SA5 = 1      ; LDE-06
447 0242          ELSE SA5 = 0
448 0244          IF (A110 >= 14.2) & (A110 < 16.9) THEN SA6 = 1      ; LDE-07
449 0245          ELSE SA6 = 0
450 0247          IF (A110 >= 16.9) & (A110 < 19.2) THEN SA7 = 1      ; LDE-08
451 0248          ELSE SA7 = 0
452 0250          IF (A110 >= 19.2) & (A110 < 21.2) THEN S80 = 1      ; LDE-09
453 0251          ELSE S80 = 0
454 0253          IF (A110 >= 21.2) & (A110 < 25.0) THEN S81 = 1      ; LDE-10
455 0254          ELSE S81 = 0
456          END
457          ;
458          ;
459          ;
460 0255          GOTO AGAIN
461          ;
462          ;
463          ;
464          ;
465          ;
466 0256 CLR:      LET PSTAT = 0                                ; Int set to zero
467          ;
468          ; If not white, then unack:
469          ;
470 0259          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
471 0260          LET FCLEAR = 1                                ; Clear flag
472 0261          GOSUB ACKK                                    ; Print message
473 0262          RETURN
474          ;
475 0263 ALM:      LET PSTAT = 1                                ; Int set to one
476          ;
477          ; If not white, then unack:
478          ;
479 0266          IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
480 0267          LET PDEV:DEV_TMPO = 2
481 0269          IF AND(PTYPE,3) THEN FALMON = 1
482          ;
483 0270          CRIT
484 0271          LAMP ADI,SKDKEY,2
485 0272          GOSUB ACKK                                    ; Print message
486          ;
487 0273          LET PSTAT = 2                                ; Ack the alarm
488 0274          LET FACK = 1                                  ; Set ACK flag for C program
489          ;
490          ;
491          ;
492          ;
493 0276          ACKK:  IF ILOOP = 0 THEN STRNGO = "LERF LECHATE PUMP #43 ON"
494 0278          IF ILOOP = 1 THEN STRNGO = "EVAPORATOR TO LERF LEAK"
495 0280          IF ILOOP = 2 THEN STRNGO = "LERF LECHATE PUMP #44 ON"
496          ;
497          IF ILOOP = 3 THEN STRNGO = "LERF MAIN GATE (NORTH) OPEN" ; JF801

```

```

496      ;      IF ILOOP = 4 THEN STRNGO = "LERF REAR GATE (SOUTH) OPEN" ; JFB01
497 0282      IF ILOOP = 5 THEN STRNGO = "LERF LECHATE PUMP #42 ON"
498 0284      IF ILOOP = 6 THEN STRNGO = "CATCH BASIN 42 LEAK DETECT "
499 0286      IF ILOOP = 7 THEN STRNGO = "CATCH BASIN 43 LEAK DETECT "
500 0288      IF ILOOP = 8 THEN STRNGO = "CATCH BASIN 44 LEAK DETECT "
501 0290      IF ILOOP = 9 THEN STRNGO = "BASIN 42 LEACHATE DIAGNOSTIC"
502 0292      IF ILOOP = 10 THEN STRNGO = "BASIN 42 LIQUID DIAGNOSTIC "
503 0294      IF ILOOP = 11 THEN STRNGO = "BASIN 43 LEACHATE DIAGNOSTIC"
504 0296      IF ILOOP = 12 THEN STRNGO = "BASIN 43 LIQUID DIAGNOSTIC "
505 0298      IF ILOOP = 13 THEN STRNGO = "BASIN 44 LEACHATE DIAGNOSTIC"
506 0300      IF ILOOP = 14 THEN STRNGO = "BASIN 44 LIQUID DIAGNOSTIC "
507 0302      IF ILOOP = 15 THEN STRNGO = "LERF MODEM COMM STATUS "
508      ;      IF ILOOP = 16 THEN STRNGO = "ETF SURGE TANK HI" DELETED
509      ;      IF ILOOP = 17 THEN STRNGO = "ETF SURGE TANK HIHI" DELETED
510      ;      IF ILOOP = 18 THEN STRNGO = "ETF COMMUNICATION FAILURE" DELETED
511      ;      IF ILOOP = 19 THEN STRNGO = "ETF AOV CLOSED" DELETED
512 0304      IF ILOOP = 27 THEN STRNGO = "LERFBATCH STARTING "
513 0306      IF ILOOP = 28 THEN STRNGO = "LERFBATCH TERMINATING "
514      ;
515      ; If acknowledging alarms, then:
516      ;
517 0307  MSG:  IF PTYPE = 1 THEN
518      BEGIN
519      ;
520      ; If white alarm, print:
521      ;
522 0309      IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNGO
523 0310      IF (FACK) THEN
524      BEGIN
525      ;
526      ; If white alarm, print:
527      ;
528 0311      PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
529 0312      LET FACK = 0
530      END
531      END
532      ;
533      ; If CLEAR, print it is ack and reset the flag:
534      ;
535 0313      IF (FCLEAR) THEN
536      BEGIN
537 0314      PRINT ADI,"%p ALARM CLEAR",STRNGO
538 0315      LET FCLEAR = 0
539      END
540      ;
541      ; If STARTUP, print startup message:
542      ;
543 0316      IF (FSTART) THEN
544      BEGIN
545 0317      LET FSTART = 0
546 0318      PRINT ADI,"%p",STRNGO
547      END
548 0319      RETURN
549      ;
550      ; If ERROR - print message, start timer, and set flag:

```

```

551      ;
552 0320  ERRSUB: IF ERRN != ERRNUM() THEN
553          BEGIN
554 0321          TIMEROW STIMEO
555 0322          TIMERCLR STIMEO
556 0323          LET FSNO = 1
557 0324          LET ERRN = ERRNUM()
558 0325          LET ERRL = ERRLIN()
559 0326          PRINT AD1,"ERROR # %i AT LINE %i",errn,errl
560          END
561 0327          ERSCLR
562 0328          GOTO AGAIN
563      ;
564      ; If FABORT = 1, then abort program
565      ;
566      ; TERM: LET FABORT = 0
567 0329  TERM: LET STRNGO = "LERFBATCH TERMINATED  "
568 0330          LET ILOOP = STAR + 2
569 0331          LET FSTART = 1
570 0332          LET STRNGO = "TERM"
571 0333          GOSUB ACKK
572          END

```

```

Total Errors Detected      0      Total Warnings      0
Sequence Program Size = 6624 Bytes, 414 Pages
      18 Forward Reference(s)

```

CASSBATCH

```

1      ;      C A S S B A T C H . S E Q                      VERSION 2
2      ;
3      ;      Written By      : R. N. Kersey
4      ;      Date           : 05-31-92
5      ;      Facility       : 242-A Evaporator Bldg.
6      ;      Machine        : VAX 4104-A
7      ;      Op. System     : VAX/VMS 5.5-2AH
8      ;      D/3 Version    : 6.3-2 BASELINE
9      ;      Compiler       : SIC
10     ;      Build command  : @[320,20]CASSBATCH
11     ;
12     ;      UNIT = 22, SKDKEY = 44
13     ;
14     ;      Assoc. UPF:      [300,1] OVRVIEW.____      OVRVIEW.LST
15     ;      Assoc. RPF:      [300,1] 242A.____      242A.LST
16     ;      SDB:            [300,1]PCMOASDB.CMD      PCMOASDB.LST
17     ;
18     ;      MODIFICATION LOG:
19     ;
20     ;      03-AUG-93      Added the following:                      M. Teats
21     ;      - comments
22     ;      - error handling
23     ;      - unit checking
24     ;      - start message
25     ;      - micro select, abort
26     ;      - redundant SABL
27     ;      - ACKK, ERRSUB, TERM subroutines
28     ;      and baselined for D/3 version 3.3.
29     ;
30     ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
31     ;      system. Rebaselined document. Removed
32     ;      obsolete Select() statements.
33     ;
34     ;      END MOD LOG
35     ;
36     ;      UNIT      EVAP
37     ;      COMMON WHOLEO,WHOLE1
38     ;
39     0000      NOP
40     0001      STA100 = 20      ; Start message
41     0002      LET STA101 = 22      ; SEQ Unit number
42     0003      LET ADI = 0      ; Alarm Destination Index
43     0004      LET FSPARO = 0      ; No Error yet
44     0005      TIMEROFF STIMEO      ; Alarm timer off
45     0006      TIMERCLR STIMEO      ; Alarm timer clear
46     0007      MSSGO = " "      ; Blank 30 character string
47     0008      LET ERRN = 0      ; Set error number to zero
48     0009      LET ERRL = 0      ; Set error line to zero
49     0010      GETUNIT TRIP      ; Which sequence unit is program running in
50     0011      IF TRIP != STA101 THEN      ; Abort if running in wrong seq. unit
51     ;      BEGIN
52     0012      FSTART = 1
53     0013      ILOOP = STA100 + 1      ; Iloop = 21
54     0014      GOSUB ACKK
55     0015      GOTO TERM

```

```

56      END
57      ; IF !SELECT() THEN GOTO AGAIN
58      ;
59      ; SEND PROGRAM START MESSAGE
60      ;
61      0016      ILOOP = STA100          ; Iloop = 20
62      0017      FSTART = 1
63      0018      GOSUB ACKK
64      0019      ON ERROR GOSUB ERRSUB
65      ;
66      ; BEGIN PROGRAM LOOP
67      ;
68      0020      AGAIN: MARK STEP "WTUNTIL",0,0
69      0021      MSSGO = "WAIT"
70      ; WAITUNTIL SELECT() ; FABORT          ; Wait until micro is selected or program aborted
71      ; IF (FABORT) THEN GOTO TERM          ; Terminate program if FABORT flag is set
72      0022      MARK STEP "RNNING",0,0
73      0023      WAIT 1
74      0024      IF (FSPARO) THEN          ; If Error flag set
75      0025      IF STIMEO > 60 THEN          ; If Error flag active for 1 minute
76      ; BEGIN          ; Reset timer and turn off flag
77      0026      TIMEROFF STIMEO
78      0027      TIMERCLR STIMEO
79      0028      LET FSPARO = 0
80      0029      LET ERRL = 0
81      0030      LET ERRL = 0
82      0031      END
83      ;
84      0031      INDEX = CASS1AL1A:DEV_TMPD
85      0032      ASSIGN PDSABL = FALO
86      0033      GOSUB FETCH
87      ;
88      0034      INDEX = CASSOAL1A:DEV_TMPD
89      0035      ASSIGN PDSABL = FAL20
90      0036      GOSUB FETCH
91      ;
92      0037      INDEX = CASSOAL2A:DEV_TMPD
93      0038      ASSIGN PDSABL = FAL40
94      0039      GOSUB FETCH
95      ;
96      0040      INDEX = CASSOAL2B:DEV_TMPD
97      0041      ASSIGN PDSABL = FAL60
98      0042      GOSUB FETCH
99      ;
100     0043      INDEX = CASS1AL2A:DEV_TMPD
101     0044      ASSIGN PDSABL = FAL80
102     0045      GOSUB FETCH
103     ;
104     0046      WAIT 2
105     0047      GOTO AGAIN
106     ;
107     ; SUBROUTINES:
108     ;
109     0048      FETCH: BITVAL = 1
110     0049      FOR JLOOP = 0 TO 13

```


~~XXXX~~ RMIS View/Print Document Cover Sheet ~~XXXX~~

This document was retrieved from the Documentation and Records Management (DRM) ISEARCH System. It is intended for Information only and may not be the most recent or updated version. Contact a Document Service Center (see Hanford Info for locations) if you need additional retrieval information.

Accession #: D196079545

Document #: SD-534-CSWL-001

Title/Desc:

SEQUENCE & BATCH LANGUAGE PROGRAMS & ALARM RELATED
C PROGRAMS FOR THE 242A MCS [SEC 5 OF 6 PAGE 512
TO 636]

Pages: 127

This document was too large to scan as a whole document, therefore it required breaking into smaller sections.

Document number: SD-534-CSWL-001

Section 5 of 6

Title: Sequence and Batch Language programs
and Alarm Related "C" Programs for the
242-A MCS

Date: 4/15/96 Revision: 3

Originator: Bergin, J.F.

Co: _____

Recipient: _____

Co: _____

References: ECN- 627504

TAG

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:36:54 24-Feb-95

Page 1

```

1      :      T A G . S E Q                      VERSION 2
2      :
3      :
4      :      Function:      EPN ON/OFF for Graphics.
5      :
6      :      Written By      : R. N. Kersey
7      :      Date           : 05-31-92
8      :      Facility      : 242-A Evaporator Bldg.
9      :      Machine       : VAX 4104-A
10     :      Op. System     : BAX/VMS 5.5-2AH
11     :      D/3 Version    : 6.3-2 BASELINE
12     :      Compiler      : SIC
13     :      Build command  : @([320,20])TAG
14     :
15     :      UNIT = 0, SKDKEY = 62
16     :
17     :      Assoc. UPF:      [300,1]_TOTAL0.____ TOTAL0.LST
18     :      Assoc. RPF:      [300,1]_ALL0.____ ALL0.LST
19     :      SDB:            [300,1]PCMOASDB.CMD PCMOASDB.LST
20     :
21     :      MODIFICATION LOG:
22     :
23     :      29-JUL-93      Added comments and baselined for      M. Teats
24     :                      D/3 version 3.3.
25     :
26     :      15-DEC-94      Revised header to indicate the new VMS      J. Berger
27     :                      system. Rebase line document.
28     :
29     :      END MOD LOG
30     :
31     :      UNIT      WHOLED
32     0001      IF(FDISP)THEN FDISP = 0
33     0002      ELSE FDISP = 1
34     :      END

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 80 Bytes, 5 Pages
0 Forward Reference(s)

TIMEINHA

VERSION 2

TIMEINHASEQ

Function: Provide time for INHIBIT program.

Written By : R. N. Kersey
Date : 05-31-92
Facility : 242-A Evaporator Bldg.
Machine : VAX 4104-A
Op. System : VAX/VMS 5.5-2AH
D/3 Version : 6.3-2 BASELINE
Compiler : SIC
Build command : @LC20,20TIMEINHASEQ

UNIT = 0, SKDKEY = 53

Assoc. UPF: [300,1]_TOTAL0.____ TOTAL0.LST
Assoc. RPF: [300,1]_ALLO.____ ALLO.LST
SDB: [300,1]PCHASDB.CHD PCHASDB.LST

MODIFICATION LOG:

29-JUL-93 Added comments and baselined for M. Teats
D/3 version 3.3.

15-DEC-94 Revised header to indicate the new VMS system. Rebase (line document). J. Berger

END MOD LOG

UNIT

WHOLE0
FTRIP0 = 0
FTRIP1 = 0
FTRIP2 = 0
FTRIP3 = 0
FSTOP = 0
INT0 = TIMSTR(TIME0)
INT0 = DATSTR(DATE0)
END

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 144 Bytes, 9 Pages
0 Forward Reference(s)

PGMOA

P G M O A . S E Q

VERSION 2

Program to monitor sequence programs running in PCWG
 PGMSTAT sets bits in an integer to give the status of a program.
 It indicates if the program is running, is critical, trapped etc.
 This program fetches this value then, by using a bit mask, it
 checks specific statuses and converts these to a single integer value
 used by the string editor. SED translates the integer to strings
 in graphic 043, titled "PROGRAM STATUS". SED = String Editor for graphics

Written By : R. N. Kersey
 Date : 05-31-92
 Facility : 242-A Evaporator Bldg.
 Machine : VAX 4104-A
 Op. System : VAX/VMS 5.5-2AH
 D/3 Version : 6.3 BASELINE
 Compiler : SIC
 Build command : a[320,20]PGM0A
 Assoc. LPT: [300,1] BLANKK.LST
 Assoc. RPT: [300,1] PGMSTAT.LST
 SOB: [300,1]PCWGASOB.CHD
 [300,1]PCWGASOB.LST

MODIFICATION LOG:

29-JUL-93 Added comments and baselined for M. Teats
 D/3 version 3.3.
 15-DEC-94 Revised header to indicate the new VMS
 system. Rebaselined document. Removed
 obsolete Select() statements.

J. Berger

END MOD LOG

UNIT
COMMON ROOMS

```

PSTAT == PALARM
PGRAPHIC == PSPARO
FABORT = 0

ASSIGN PGRAPHIC = INTO
FOR ILOOP = 0 TO 39
  PGRAPHIC = 0
  NEXT ILOOP

TIMERON STM1
ON ERROR GOSUB ERRSUB

AGAIN:
MARK STEP "MTUNTIL",0,0
LS66 = "WAIT"
WAIT 1
IF FABORT THEN GOTO TERM
LS66 = "RUNNING"

```


PAGE 518

```

111 0045      FOR INDEX = 1 TO 4                      ; 2 = 2 = TRAPPED
112 0047      IF AND(BVAL,PSTAT)THEN REMAIN = INDEX    ; 4 = 3 = CRITICAL
113 0048      BVAL = BVAL < 1                          ; 8 = 4 = WAIT
114 0049      NEXT INDEX
115          ;
116 0050      IF PSTAT = -1 THEN                        ; -1 = 6 = NOTRUNNING
117          BEGIN
118 0051          FTRIP1 = 1
119 0052          PGRAPHIC = 6
120          END
121      ELSE
122          BEGIN
123 0053          PGRAPHIC = REMAIN                      ; set SED value
124 0055          IF AND(PSTAT,4096)THEN PGRAPHIC = 5    ; 4096 = 5 = WAITUNTIL
125          END
126 0056          TRIP = OR(TRIP,PSTAT)                  ; Accumulate for all pgms
127 0057          STEP PGRAPHIC BY 1
128 0058          STEP PSTAT BY 1
129 0059          NEXT ILOOP
130          ;
131          ;
132          ;      TRIP = TRAPPED
133          ;      FTRIP1 = NOTRUNNING
134          ;      FTRIP2 = FLAG FROM DOGO
135          ;      FTRIP4 = FAULT
136          ;
137          ;
138          BEGIN
139 0061          IF AND(AUDALARMO:DEV_DFLG,1)THEN FTRIP4 = 1 ; IF AUDALARMO IN FAULT
140 0062          ELSE FTRIP4 = 0
141 0063          FTRIP0 = OR(FTRIP1,FTRIP4)                ; IF FAULT OR NOTRUNNING
142 0064          FSPARO = AND(TRIP,2)                    ; IF ONE PGM TRAPPED
143 0065          FTRIP0 = OR(FTRIP0,FSPARO)              ; TOTAL ALL
144 0067          IF OR(FTRIP0,FTRIP2)THEN LAMP 0,48,3
145 0068          ELSE LAMP 0,48,0
146          END
147 0069          GOTO AGAIN
148          ;
149          ;
150          ;
151          ;=====
152          ;      SUBROUTINES
153          ;
154  ERRSUB:
155          ;
156          ; errors are handled differently depending on their severity
157          ;
158          ; 19  PROGRAM ALREADY IN KICK SUBROUTINE (rekick)
159          ; 32  UNDEFINED COMMUNICATIONS PACKET TYPE (ignore)
160          ; 41  PCM DB HEADER NOT LOADED (ignore)
161          ; 69  IPN NOT DEFINED (ignore)
162          ; 162 NO SUCH PROGRAM IN THIS PCM (ignore)
163          ; 165 SEQUENCE PROGRAM ALREADY RUNNING (ignore)
164          ; 170 SERIAL NUMBER MISMATCH (wait forever)
165          ; 179 ILLEGAL SUBROUTINE STACK CLEAR (ignore)

```

```

166      ; 187 BACKUP PCM IS DOWN (ignore)
167      ; 193 NO SUCH PROGRAM IN THIS PCM (ignore)
168      ; 198 ILLEGAL SUBROUTINE STACK NESTING ERROR (ignore)
169      ; 199 ILLEGAL RDF NAME (wait forever)
170      ; 200 RDF NOT LOADED IN MEMORY (wait forever)
171      ; 201 RECIPE LOAD NOT ENABLED (wait forever)
172      ; 222 BOARDS DEAD (ignore)
173      ; 226 BOARD ERROR E.G. broken skid panel etc... (ignore)
174      ; 245 BOTH PCMS DOWN, SENT REDUNDANT HIGHWAY ACCESS (ignore)
175      ;
176 0070      ERRN = ABS(ERRNUM())      ; get error number
177 0071      ERRL = ERRLIN()          ; get error line number
178 0072      TIMERON STMRO
179 0073      TIMERCLR STMRO
180 0074      FSPAR5 = 1
181      ;
182      ; if error is not serious, clear and return to program
183      ;
184 0075      IF (ERRN=187) | (ERRN=32) | (ERRN=69) | (ERRN=222) THEN
185 0076          ERSRET
186
187 0077      IF (ERRN=41) | (ERRN=165) | (ERRN=162) | (ERRN=179) THEN
188 0078          ERSRET
189
190 0079      IF (ERRN=226) | (ERRN=198) | (ERRN=245) THEN
191 0080          ERSRET
192
193 0081      IF (ERRN=193) | (ERRN=19) THEN
194 0082          ERSRET
195
196      ; else report the error, and wait for corrective action ....
197 0083      PRINT 0,"ERROR NUMBER %i AT LINE %i ****",ERRN,ERRL
198
199 0084      ERSCLR
200 0085      GOTO AGAIN                ; when corrective action taken,
                                   ; goto done and cleanup
201
202      ;
203      ; =====
204      ;
205      ;
206 0086      TERM:  ASSIGN PGRAPHIC = INT1
207 0087              FOR ILOOP = 1 TO 20
208 0088              PGRAPHIC = 7
209 0089              STEP PGRAPHIC BY 1
210 0090              NEXT ILOOP
211      END

```

```

Total Errors Detected      0      Total Warnings      0
Sequence Program Size = 2032 Bytes, 127 Pages
      1 Forward Reference(s)

```

WHITE1

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:04:09 08-Jan-96

Page 1

```

1      ;      W H I T E 1 . S E Q      VERSION 11
2      ;
3      ;      Function:      Drives WHITE ALARM SUMMARY PAGES,
4      ;      Graphic numbers 100 - 102.
5      ;
6      ;      Written By      : R. W. Kersey
7      ;      Date           : 05-31-92
8      ;      Facility       : 242-A Evaporator Bldg.
9      ;      Machine        : VAX 4104-A
10     ;      Op. System     : VAX/VMS 5.5-2AH
11     ;      D/3 Version    : 6.3-2 BASELINE
12     ;      Compiler       : SIC
13     ;      Build command  : @[320,20]WHITE1
14     ;
15     ;      UNIT = 0, SKDKEY = 60
16     ;
17     ;      Assoc. UPF:     [300,1]_TOTAL1.____ TOTAL1.LST
18     ;      Assoc. RPF:     [300,1]_ALL1.____ ALL1.LST
19     ;      SDB:           [300,1]PCM1ASDB.____ PCM1ASDB.CMD
20     ;
21     ;      POINTER      SDB VAR.      FUNCTION
22     ;      -----
23     ;      PTYPE        FDSn          Alarm inhibit status
24     ;      PREV         FRVn          Alarm condition or setpoint
25     ;      PSTAT        Fxn          Current signal status
26     ;      PALARM       FALn          Previous alarm condition
27     ;
28     ;      MODIFICATION LOG:
29     ;
30     ;      12-SEP-92      Changed LI-C100-2 to LI-C100-1 and alarm M. Tests
31     ;                      level from 0.95 to 0.98
32     ;
33     ;      16-SEP-92      Deleted DDI-CA1-1 alarms (96+97)          M. Tests
34     ;
35     ;      02-DEC-92      - Corrected AMU strings                      M. Tests
36     ;                      - Modified P1-RW-2 alarms
37     ;                      - Added comments
38     ;
39     ;      09-DEC-92      Changed FIC-EC3-1 LO alarm from 140 to 50    M. Tests
40     ;
41     ;      29-DEC-92      Corrected white alarm level on II-AS-1      M. Tests
42     ;
43     ;      13-FEB-93      Added LERF gate white alarms:              M. Tests
44     ;                      ZS-105PS1      (171)
45     ;                      ZS-105PS2      (172)
46     ;
47     ;      01-APR-93      Changed FRV128 (YS-A-C100) and FRV129      M. Tests
48     ;                      (YS-P-C100) from 1 to 0 so the alarm occurs
49     ;                      on shutdown. Also changed strings for labels
50     ;                      S128: and S129: from "...CONFIRM ON " to
51     ;                      "...SHUTDOWN ".
52     ;
53     ;      04-APR-93      Reinstated white alarms on AW and AN pump    M. Tests
54     ;                      pump shutdown (YS-) signals, and on AP pump
55     ;                      MCC status (YS-) signals.

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:04:09 08-Jan-96

Page 2

```

56      ;
57      ;mct1+ 13-MAY-93      Remove AN pump shutdown DEV white alarms,      M. Teats
58      ;                      which are based on the analog signals,
59      ;                      since they are not needed due to the 4/4/93
60      ;                      mods. See CCN# 93MAY13-01.
61      ;
62      ;mct2+ 13-MAY-93      Remove AN pump shutdown DEV white alarms,      M. Teats
63      ;                      which are based on the analog signals,
64      ;                      since they are not needed due to the 4/4/93
65      ;                      mods. See CCN# 93MAY13-01.
66      ;
67      ;mct3+ 14-MAY-93      Corrected typo on FDS36 and FX36 from      M. Teats
68      ;                      PSH-APFL1 to FXS-DR712. Added FRV11, 17,
69      ;                      and 18 = 1 to get "MCC OFF" alarm for AP
70      ;                      pumps 103, 108, and 102M. See CCN# 93MAY14-01.
71      ;
72      ;                      Summary To Date Of Deletions:
73      ;                      Deleted Points: 47-50, 80-83
74      ;                      Deleted alarms: 85-89, 93, 95, 96-97, 147, 118,
75      ;                      120, 122-125, 127, 134, 137, 141-142
76      ;                      Deleted DEV white alarms: 53-59 (AN pumps) -mct2-
77      ;                      Deleted DEV white alarms: 70-75 (AW pumps) -mct1-
78      ;
79      ;mct4+ 19-MAY-93      Added YS-P-42, YS-P-43, YS-P-44 (173-175)      M. Teats
80      ;                      per CCN# 93MAY19-02.
81      ;
82      ;                      10-JUN-93      Added comments and baselined for      M. Teats
83      ;                      D/3 version 3.3. CCN# 93JUN01-01.
84      ;
85      ;mct5+ 12-JAN-94      Deleted YS-AW102 per CCN# 94JAN12-01.      M. Teats
86      ;                      and deleted II-P-C100
87      ;
88      ;                      01-AUG-94      Added White Alarm for PDI-CA1-3 per      J.F. Berger
89      ;                      EE-94-00019.
90      ;
91      ;                      15-DEC-94      Revised header to indicate the new VMS      J. Berger
92      ;                      system. Rebaselined document. Removed
93      ;                      obsolete Select() statement.
94      ;
95      ;                      30-MAR-95      Revised White1 to incorporate EPNs for      J. Berger
96      ;                      the PC Recycle System. YS-P-C106, HV-CA1-10.
97      ;                      Per ECN 620390 and WP EE-95-00036. (JFB01)
98      ;
99      ;                      10-APR-95      Revised to remove LERF gate alarms, ZS-105PS1      J. Berger
100     ;                      and ZS-105PS2. See ECN 620391 and WP EE-95-
101     ;                      00006. (JFB02)
102     ;
103     ;                      27-APR-95      Deleted YS-P-C100 from white alarm list      J. Berger
104     ;                      See WP EE-95-00006 and ECN 622499.
105     ;
106     ;                      15-MAY-95      Deleted the GROUT signals from the White      J. Berger
107     ;                      alarms. See EE-95-00126 and ECN 620366. jfb03
108     ;
109     ;                      03-OCT-95      Deleted EPN PSL-EA1-1 per ECN 623939 &      J. Berger
110     ;                      EE-95-00006. JFB04

```

```

111 ;
112 ; 27-DEC-95 Deleted PSW-FP per ECH 627413 and J. Berger
113 ; package MCS-96-001. JF805
114 ;
115 ;
116 ; END MOD LOG
117 ;
118 UNIT WHOLE1
119 COMMON WHOLE0
120 ;
121 NXTGRA == PDATE
122 NXTMSG == PTIME
123 NXTINT == PEPN
124 NXTLSG == PALM
125 PLRGSG == PTRIP
126 STPPER == WHITE
127 ;
128 0000 ASSIGN PGRAFX = GRAFX0
129 0001 ASSIGN PMESSG = MSSG0
130 0002 ASSIGN POSABL = LSG0
131 0003 ASSIGN PLINE = LINE0
132 0004 FOR ILOOP = 0 TO 46
133 0005 PGRAFX = 0
134 0006 PMESSG = " "
135 0007 POSABL = " "
136 0008 PLINE = 0
137 0009 STEP PGRAFX BY 1
138 0010 STEP PMESSG BY 1
139 0011 STEP POSABL BY 1
140 0012 STEP PLINE BY 1
141 0013 NEXT ILOOP
142 ;
143 0014 ERRN = 0
144 0015 ERRL = 0
145 ;
146 0016 LSG47 = " "
147 0017 MSSG47 = " "
148 0018 INT47 = 0
149 0019 GRAF47 = 0
150 ;
151 ; BEGIN INITIALIZING:
152 ;
153 0020 ASSIGN PSTAT = FX1
154 0021 ASSIGN PALARM = FAL1
155 0022 ASSIGN PREV = FRV1
156 0023 ASSIGN PTYPE = FDS1
157 0024 ASSIGN PLINE = INT1
158 0025 FOR ILOOP = 1 TO 177 ; JF801
159 0026 PSTAT = 0
160 0027 PALARM = 0
161 0028 PREV = 0
162 0029 PTYPE = 1
163 0030 PLINE = 0
164 0031 STEP PSTAT BY 1
165 0032 STEP PALARM BY 1

```


[illegible]

```

276      ;
277 0112 AGAIN: WAIT 3 ; Go get alarm status for all
278      ; WAITUNTIL SELECT() ; Wait until micro selected
279 0113 ; IF (FSPARO) THEN ; If alarm flag is set
280 0114 IF STMRO > 60 THEN ; If alarm flag active for 1 minute
281      BEGIN ; Reset timer and turn off flag
282 0115 TIMEROFF STMRO
283 0116 TIMERCLR STMRO
284 0117 FSPARO = 0
285 0118 ERRN = 0
286 0119 ERRRL = 0
287      END
288 0120 SORT = PGMSTAT("PGM1A","BLANK1")
289 0121 REMAIN = 0
290 0122 BITVAL = 1
291 0123 FOR INDEX = 1 TO 4
292 0125 IF AND(BITVAL,SORT) THEN REMAIN = INDEX
293 0126 BITVAL = BITVAL < 1
294 0127 NEXT INDEX
295 0129 IF SORT = -1 THEN INO = 6
296      ELSE
297      BEGIN
298 0130 INO = REMAIN
299 0132 IF AND(SORT,4096) THEN INO = 5
300      END
301 0133 IF FTRIPO THEN
302      BEGIN
303 0134 LAMP ADI,60,0
304 0135 FTRIPO = 0
305      END
306      ;
307      ; DETERMINE INHIBITED STATUS, (PTYPE = FDSn = 1 IF INHA):
308      ;
309 0136 FDS1 = AND('YS-AN-101':DIN_ALMCT,16) ; 1
310 0137 FDS2 = AND('YS-AN-102':DIN_ALMCT,16) ; 2
311 0138 FDS3 = AND('YS-AN-103':DIN_ALMCT,16) ; 3
312 0139 FDS4 = AND('YS-AN-104':DIN_ALMCT,16) ; 4
313 0140 FDS5 = AND('YS-AN-105':DIN_ALMCT,16) ; 5
314 0141 FDS6 = AND('YS-AN-106':DIN_ALMCT,16) ; 6
315 0142 FDS7 = AND('YS-AN-107':DIN_ALMCT,16) ; 7
316 0143 FDS8 = AND('YS-AN-21':DIN_ALMCT,16) ; 8
317 0144 FDS9 = AND('YS-AP-101':DIN_ALMCT,16) ; 9
318 0145 FDS10 = AND('YS-AP-102':DIN_ALMCT,16) ; 10
319 0146 FDS11 = AND('YS-AP-103':DIN_ALMCT,16) ; 11
320 0147 FDS12 = AND('LDS-AP03D':DIN_ALMCT,16) ; 12
321 0148 FDS13 = AND('YS-AP-104':DIN_ALMCT,16) ; 13
322 0149 FDS14 = AND('YS-AP-105':DIN_ALMCT,16) ; 14
323 0150 FDS15 = AND('YS-AP-106':DIN_ALMCT,16) ; 15
324 0151 FDS16 = AND('YS-AP-107':DIN_ALMCT,16) ; 16
325 0152 FDS17 = AND('YS-AP-108':DIN_ALMCT,16) ; 17
326      ; FDS18 = AND('YS-AP102M':DIN_ALMCT,16) ; 18 jfb03
327      ; FDS19 = AND('PSHAP102M':DIN_ALMCT,16) ; 19 jfb03
328      ; FDS20 = AND('PSLAP102M':DIN_ALMCT,16) ; 20 jfb03
329      ; FDS21 = AND('YS-AP102R':DIN_ALMCT,16) ; 21 jfb03
330      ; FDS22 = AND('WSHAP102M':DIN_ALMCT,16) ; 22 jfb03

```

```

331      ;
332      ;
333      FDS23 = AND('TSMAP102M':DIN_ALMCT, 16)
334      FDS24 = AND('VSMAP102M':DIN_ALMCT, 16)
335      FDS25 = AND('TXS-AP-2':DIN_ALMCT, 16)
336      FDS26 = AND('FSL-AP2711':DIN_ALMCT, 16)
337      FDS27 = AND('PSH-APFP1':DIN_ALMCT, 16)
338      FDS28 = AND('2S-AP-FP1':DIN_ALMCT, 16)
339      FDS29 = AND('FSL-APK13':DIN_ALMCT, 16)
340      FDS30 = AND('FSL-APK12':DIN_ALMCT, 16)
341      FDS31 = AND('FSL-APK11':DIN_ALMCT, 16)
342      FDS32 = AND('FSL-APK23':DIN_ALMCT, 16)
343      FDS33 = AND('FSL-APK22':DIN_ALMCT, 16)
344      FDS34 = AND('FSL-APK21':DIN_ALMCT, 16)
345      FDS35 = AND('FSL-DR712':DIN_ALMCT, 16)
346      FDS36 = AND('FSL-DR712':DIN_ALMCT, 16)
347      FDS37 = AND('PSH-APFL1':DIN_ALMCT, 16)
348      FDS38 = AND('2S-AP-DR1':DIN_ALMCT, 16)
349      FDS39 = AND('2S-AP-FL1':DIN_ALMCT, 16)
350      FDS40 = AND('YS-AM-101':DIN_ALMCT, 16)
351      FDS41 = AND('YS-AM-102':DIN_ALMCT, 16)
352      FDS42 = AND('YS-AM-103':DIN_ALMCT, 16)
353      FDS43 = AND('YS-AM-104':DIN_ALMCT, 16)
354      FDS44 = AND('YS-AM-105':DIN_ALMCT, 16)
355      FDS45 = AND('YS-AM-106':DIN_ALMCT, 16)
356      FDS46 = AND('TXS-AM-2':DIN_ALMCT, 16)
357      ;
358      ; deleted YS-AY-101, -102, and YS-AZ-101, -102: 47-50
359      ;
360      FDS51 = AND('LSH-AX-1':DIN_ALMCT, 16)
361      FDS52 = AND('LSL-AX-1':DIN_ALMCT, 16)
362      FDS53 = AND('P-AM-101':DEV_ALMCT, 16)
363      FDS54 = AND('P-AM-102':DEV_ALMCT, 16)
364      FDS55 = AND('P-AM-103':DEV_ALMCT, 16)
365      FDS56 = AND('P-AM-104':DEV_ALMCT, 16)
366      FDS57 = AND('P-AM-105':DEV_ALMCT, 16)
367      FDS58 = AND('P-AM-106':DEV_ALMCT, 16)
368      FDS59 = AND('P-AM-107':DEV_ALMCT, 16)
369      FDS60 = AND('P-AM-101':DEV_ALMCT, 16)
370      FDS61 = AND('P-AP-102':DEV_ALMCT, 16)
371      FDS62 = AND('P-AP-103':DEV_ALMCT, 16)
372      FDS63 = AND('P-AP-104':DEV_ALMCT, 16)
373      FDS64 = AND('P-AP-105':DEV_ALMCT, 16)
374      FDS65 = AND('P-AP-106':DEV_ALMCT, 16)
375      FDS66 = AND('P-AP-107':DEV_ALMCT, 16)
376      FDS67 = AND('P-AP-108':DEV_ALMCT, 16)
377      FDS68 = AND('P-AP-102M':DEV_ALMCT, 16)
378      FDS69 = AND('P-AP-102M':DEV_ALMCT, 16)
379      FDS70 = AND('P-AM-101':DEV_ALMCT, 16)
380      FDS71 = AND('P-AM-102':DEV_ALMCT, 16)
381      FDS72 = AND('P-AM-103':DEV_ALMCT, 16)
382      FDS73 = AND('P-AM-104':DEV_ALMCT, 16)
383      FDS74 = AND('P-AM-105':DEV_ALMCT, 16)
384      FDS75 = AND('P-AM-106':DEV_ALMCT, 16)
385      FDS76 = AND('P-AM-107':DEV_ALMCT, 16)
386      FDS77 = AND('P-AY-101':DEV_ALMCT, 16)
387      FDS78 = AND('P-AZ-101':DEV_ALMCT, 16)
388      ;
389      ;
390      ;
391      ;
392      ;
393      ;
394      ;
395      ;
396      ;
397      ;
398      ;
399      ;
400      ;
401      ;
402      ;
403      ;
404      ;
405      ;
406      ;
407      ;
408      ;
409      ;
410      ;
411      ;
412      ;
413      ;
414      ;
415      ;
416      ;
417      ;
418      ;
419      ;
420      ;
421      ;
422      ;
423      ;
424      ;
425      ;
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432      ;
433      ;
434      ;
435      ;
436      ;
437      ;
438      ;
439      ;
440      ;
441      ;
442      ;
443      ;
444      ;
445      ;
446      ;
447      ;
448      ;
449      ;
450      ;
451      ;
452      ;
453      ;
454      ;
455      ;
456      ;
457      ;
458      ;
459      ;
460      ;
461      ;
462      ;
463      ;
464      ;
465      ;
466      ;
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473      ;
474      ;
475      ;
476      ;
477      ;
478      ;
479      ;
480      ;
481      ;
482      ;
483      ;
484      ;
485      ;
486      ;
487      ;
488      ;
489      ;
490      ;
491      ;
492      ;
493      ;
494      ;
495      ;
496      ;
497      ;
498      ;
499      ;
500      ;
501      ;
502      ;
503      ;
504      ;
505      ;
506      ;
507      ;
508      ;
509      ;
510      ;
511      ;
512      ;
513      ;
514      ;
515      ;
516      ;
517      ;
518      ;
519      ;
520      ;
521      ;
522      ;
523      ;
524      ;
525      ;
526      ;
527      ;
528      ;
529      ;
530      ;
531      ;
532      ;
533      ;
534      ;
535      ;
536      ;
537      ;
538      ;
539      ;
540      ;
541      ;
542      ;
543      ;
544      ;
545      ;
546      ;
547      ;
548      ;
549      ;
550      ;
551      ;
552      ;
553      ;
554      ;
555      ;
556      ;
557      ;
558      ;
559      ;
560      ;
561      ;
562      ;
563      ;
564      ;
565      ;
566      ;
567      ;
568      ;
569      ;
570      ;
571      ;
572      ;
573      ;
574      ;
575      ;
576      ;
577      ;
578      ;
579      ;
580      ;
581      ;
582      ;
583      ;
584      ;
585      ;
586      ;
587      ;
588      ;
589      ;
590      ;
591      ;
592      ;
593      ;
594      ;
595      ;
596      ;
597      ;
598      ;
599      ;
600      ;
601      ;
602      ;
603      ;
604      ;
605      ;
606      ;
607      ;
608      ;
609      ;
610      ;
611      ;
612      ;
613      ;
614      ;
615      ;
616      ;
617      ;
618      ;
619      ;
620      ;
621      ;
622      ;
623      ;
624      ;
625      ;
626      ;
627      ;
628      ;
629      ;
630      ;
631      ;
632      ;
633      ;
634      ;
635      ;
636      ;
637      ;
638      ;
639      ;
640      ;
641      ;
642      ;
643      ;
644      ;
645      ;
646      ;
647      ;
648      ;
649      ;
650      ;
651      ;
652      ;
653      ;
654      ;
655      ;
656      ;
657      ;
658      ;
659      ;
660      ;
661      ;
662      ;
663      ;
664      ;
665      ;
666      ;
667      ;
668      ;
669      ;
670      ;
671      ;
672      ;
673      ;
674      ;
675      ;
676      ;
677      ;
678      ;
679      ;
680      ;
681      ;
682      ;
683      ;
684      ;
685      ;
686      ;
687      ;
688      ;
689      ;
690      ;
691      ;
692      ;
693      ;
694      ;
695      ;
696      ;
697      ;
698      ;
699      ;
700      ;
701      ;
702      ;
703      ;
704      ;
705      ;
706      ;
707      ;
708      ;
709      ;
710      ;
711      ;
712      ;
713      ;
714      ;
715      ;
716      ;
717      ;
718      ;
719      ;
720      ;
721      ;
722      ;
723      ;
724      ;
725      ;
726      ;
727      ;
728      ;
729      ;
730      ;
731      ;
732      ;
733      ;
734      ;
735      ;
736      ;
737      ;
738      ;
739      ;
740      ;
741      ;
742      ;
743      ;
744      ;
745      ;
746      ;
747      ;
748      ;
749      ;
750      ;
751      ;
752      ;
753      ;
754      ;
755      ;
756      ;
757      ;
758      ;
759      ;
760      ;
761      ;
762      ;
763      ;
764      ;
765      ;
766      ;
767      ;
768      ;
769      ;
770      ;
771      ;
772      ;
773      ;
774      ;
775      ;
776      ;
777      ;
778      ;
779      ;
780      ;
781      ;
782      ;
783      ;
784      ;
785      ;
786      ;
787      ;
788      ;
789      ;
790      ;
791      ;
792      ;
793      ;
794      ;
795      ;
796      ;
797      ;
798      ;
799      ;
800      ;
801      ;
802      ;
803      ;
804      ;
805      ;
806      ;
807      ;
808      ;
809      ;
810      ;
811      ;
812      ;
813      ;
814      ;
815      ;
816      ;
817      ;
818      ;
819      ;
820      ;
821      ;
822      ;
823      ;
824      ;
825      ;
826      ;
827      ;
828      ;
829      ;
830      ;
831      ;
832      ;
833      ;
834      ;
835      ;
836      ;
837      ;
838      ;
839      ;
840      ;
841      ;
842      ;
843      ;
844      ;
845      ;
846      ;
847      ;
848      ;
849      ;
850      ;
851      ;
852      ;
853      ;
854      ;
855      ;
856      ;
857      ;
858      ;
859      ;
860      ;
861      ;
862      ;
863      ;
864      ;
865      ;
866      ;
867      ;
868      ;
869      ;
870      ;
871      ;
872      ;
873      ;
874      ;
875      ;
876      ;
877      ;
878      ;
879      ;
880      ;
881      ;
882      ;
883      ;
884      ;
885      ;
886      ;
887      ;
888      ;
889      ;
890      ;
891      ;
892      ;
893      ;
894      ;
895      ;
896      ;
897      ;
898      ;
899      ;
900      ;
901      ;
902      ;
903      ;
904      ;
905      ;
906      ;
907      ;
908      ;
909      ;
910      ;
911      ;
912      ;
913      ;
914      ;
915      ;
916      ;
917      ;
918      ;
919      ;
920      ;
921      ;
922      ;
923      ;
924      ;
925      ;
926      ;
927      ;
928      ;
929      ;
930      ;
931      ;
932      ;
933      ;
934      ;
935      ;
936      ;
937      ;
938      ;
939      ;
940      ;
941      ;
942      ;
943      ;
944      ;
945      ;
946      ;
947      ;
948      ;
949      ;
950      ;
951      ;
952      ;
953      ;
954      ;
955      ;
956      ;
957      ;
958      ;
959      ;
960      ;
961      ;
962      ;
963      ;
964      ;
965      ;
966      ;
967      ;
968      ;
969      ;
970      ;
971      ;
972      ;
973      ;
974      ;
975      ;
976      ;
977      ;
978      ;
979      ;
980      ;
981      ;
982      ;
983      ;
984      ;
985      ;
986      ;
987      ;
988      ;
989      ;
990      ;
991      ;
992      ;
993      ;
994      ;
995      ;
996      ;
997      ;
998      ;
999      ;
1000      ;

```

```

386 0188      FDS79 = AND('P-AZ-102':DEV_ALMCT,16)      ; 79      ;JFB05
387          ; FDS84 = AND('PSW-FP':DIN_ALMCT,16)      ; 84
388 0189      FDS90 = AND('TI-244-TK':AI_ALMCT,16)      ; 90
389 0190      FDS91 = AND('FI-244EXH':AI_ALMCT,16)      ; 91
390 0191      FDS92 = AND('FI-244EXH':AI_ALMCT,16)      ; 92
391 0192      FDS94 = AND('II-AW-102':AI_ALMCT,16)      ; 94
392          ; FDS96 = AND('DDI-CA1-1':AI_ALMCT,16)      ; 96
393          ; FDS97 = AND('DDI-CA1-1':AI_ALMCT,16)      ; 97
394 0193      FDS98 = AND('PDI-FH1-1':AI_ALMCT,16)      ; 98
395 0194      FDS99 = AND('PDI-FH2-1':AI_ALMCT,16)      ; 99
396 0195      FDS100 = AND('PDI-FH3-1':AI_ALMCT,16)      ; 100
397 0196      FDS101 = AND('SIC-PB2-1':AI_ALMCT,16)      ; 101
398          ; FDS102 = AND('PSL-EA1-1':DIN_ALMCT,16)      ; 102 JFB04
399 0197      FDS103 = AND('PI-EA1-13':AI_ALMCT,16)      ; 103
400 0198      FDS104 = AND('PI-EA1-13':AI_ALMCT,16)      ; 104
401 0199      FDS105 = AND('FIC-EA1-1':AI_ALMCT,16)      ; 105
402 0200      FDS106 = AND('PI-EA1-1':AI_ALMCT,16)      ; 106
403 0201      FDS107 = AND('PI-EA1-1':AI_ALMCT,16)      ; 107
404 0202      FDS108 = AND('TI-EA1-5':AI_ALMCT,16)      ; 108
405 0203      FDS109 = AND('PI-EC1-5':AI_ALMCT,16)      ; 109
406 0204      FDS110 = AND('FIC-EC3-1':AI_ALMCT,16)      ; 110
407 0205      FDS111 = AND('TIC-EC1-2':AI_ALMCT,16)      ; 111
408 0206      FDS112 = AND('LDS-BFP-1':DIN_ALMCT,16)      ; 112
409 0207      FDS113 = AND('LSL-RWSMP':DIN_ALMCT,16)      ; 113
410 0208      FDS114 = AND('LDS-BFP-2':DIN_ALMCT,16)      ; 114
411 0209      FDS115 = AND('PDI-RW-1':AI_ALMCT,16)      ; 115
412 0210      FDS116 = AND('PDI-RW-2':AI_ALMCT,16)      ; 116
413 0211      FDS117 = AND('PI-RW-2':AI_ALMCT,16)      ; 117
414          ; FDS118 = AND('PI-RW-2':AI_ALMCT,16)      ; 118
415 0212      FDS119 = AND('FIC-EC1-1':AI_ALMCT,16)      ; 119
416          ; FDS120 = AND('FIC-EC1-1':AI_ALMCT,16)      ; 120
417 0213      FDS121 = AND('II-EXC-1':AI_ALMCT,16)      ; 121
418 0214      FDS126 = AND('II-AS-1':AI_ALMCT,16)      ; 126
419 0215      FDS128 = AND('YS-A-C100':DIN_ALMCT,16)      ; 128
420          ; FDS129 = AND('YS-P-C100':DIN_ALMCT,16)      ; 129 JFB03
421 0216      FDS130 = AND('ZS-IDK-2':DIN_ALMCT,16)      ; 130
422 0217      FDS131 = AND('PSH-ILF-1':DIN_ALMCT,16)      ; 131
423 0218      FDS132 = AND('ZS-RC3-4':DIN_ALMCT,16)      ; 132
424 0219      FDS133 = AND('ZS-RC3-5':DIN_ALMCT,16)      ; 133
425 0220      FDS135 = AND('LI-C100-1':AI_ALMCT,16)      ; 135
426          ; FDS136 = AND('II-P-C100':AI_ALMCT,16)      ; 136      commented out, -mct5+
427 0221      FDS137 = AND('PDI-CA1-3':AI_ALMCT,16)      ; 137
428 0222      FDS138 = AND('LSL-SC-1':DIN_ALMCT,16)      ; 138
429 0223      FDS139 = AND('LSL-SC-2':DIN_ALMCT,16)      ; 139
430 0224      FDS140 = AND('LSL-SC-3':DIN_ALMCT,16)      ; 140
431 0225      FDS143 = AND('YS-CPE1-1':DIN_ALMCT,16)      ; 143
432 0226      FDS144 = AND('YS-CPE2-1':DIN_ALMCT,16)      ; 144
433 0227      FDS145 = AND('PI-IA-1':AI_ALMCT,16)      ; 145
434 0228      FDS146 = AND('PI-IA-1':AI_ALMCT,16)      ; 146
435          ; FDS147 = AND('PI-PA-1':AI_ALMCT,16)      ; 147
436 0229      FDS148 = AND('PI-PA-1':AI_ALMCT,16)      ; 148
437 0230      FDS149 = AND('TI-AMBS-1':AI_ALMCT,16)      ; 149
438 0231      FDS150 = AND('TI-AMBS-1':AI_ALMCT,16)      ; 150
439 0232      FDS151 = AND('TI-AMBS-1':AI_ALMCT,16)      ; 151
440 0233      FDS152 = AND('TI-AMBS-1':AI_ALMCT,16)      ; 152

```

```

441 0234      FDS153 = AND('TI1-AMBS-2':AI_ALMCT,16)      ; 153
442 0235      FDS154 = AND('TI1-AMBS-2':AI_ALMCT,16)      ; 154
443 0236      FDS155 = AND('TI1-AMBS-2':AI_ALMCT,16)      ; 155
444 0237      FDS156 = AND('TI1-AMBS-2':AI_ALMCT,16)      ; 156
445 0238      FDS157 = AND('YS-A-E101':DIN_ALMCT,16)      ; 157
446 0239      FDS158 = AND('YS-P-E101':DIN_ALMCT,16)      ; 158
447 0240      FDS159 = AND('YS-P-E102':DIN_ALMCT,16)      ; 159
448 0241      FDS160 = AND('YS-P-E103':DIN_ALMCT,16)      ; 160
449 0242      FDS161 = AND('YS-A-E104':DIN_ALMCT,16)      ; 161
450 0243      FDS162 = AND('YS-P-E104':DIN_ALMCT,16)      ; 162
451 0244      FDS163 = AND('TI1-E101-2':AI_ALMCT,16)      ; 163
452 0245      FDS164 = AND('WF1-E101':AI_ALMCT,16)      ; 164
453 0246      FDS165 = AND('WF1-E101':AI_ALMCT,16)      ; 165
454 0247      FDS166 = AND('WF1-E102':AI_ALMCT,16)      ; 166
455 0248      FDS167 = AND('WF1-E102':AI_ALMCT,16)      ; 167
456 0249      FDS168 = AND('TI1-E104-2':AI_ALMCT,16)      ; 168
457 0250      FDS169 = AND('WF1-E104':AI_ALMCT,16)      ; 169
458 0251      FDS170 = AND('WF1-E104':AI_ALMCT,16)      ; 170
459          ;
460 0252      FDS171 = AND('ZS-10SPS1':DIN_ALMCT,16)      ; 171
461 0253      FDS172 = AND('ZS-10SPS2':DIN_ALMCT,16)      ; 172
462 0254      FDS173 = AND('YS-P-42':DIN_ALMCT,16)      ; 173
463 0255      FDS174 = AND('YS-P-43':DIN_ALMCT,16)      ; 174
464 0256      FDS175 = AND('YS-P-44':DIN_ALMCT,16)      ; 175
465          ;
466          FDS176 = AND('WV-CAT-10':DEV_ALMCT,16)      ; 176
467          ;
468          ;
469          ;
470          ;
471          ;
472          ;
473          ;
474          ;
475          ;
476          ;
477          ;
478          ;
479          ;
480          ;
481          ;
482          ;
483          ;
484          ;
485          ;
486          ;
487          ;
488          ;
489          ;
490          ;
491          ;
492          ;
493          ;
494          ;
495          ;

```

JF802
JF802
*mc14-
*mc14-
JF801
JF801
JF801

; DETERMINE SIGNAL STATUS, PSTAT (FXn):

```

FX1 = 'YS-AM-101':DIN_VAL      ; 1
FX2 = 'YS-AM-102':DIN_VAL      ; 2
FX3 = 'YS-AM-103':DIN_VAL      ; 3
FX4 = 'YS-AM-104':DIN_VAL      ; 4
FX5 = 'YS-AM-105':DIN_VAL      ; 5
FX6 = 'YS-AM-106':DIN_VAL      ; 6
FX7 = 'YS-AM-107':DIN_VAL      ; 7
FX8 = 'YS-AM-108':DIN_VAL      ; 8
FX9 = 'YS-AP-101':DIN_VAL      ; 9
FX10 = 'YS-AP-102':DIN_VAL      ; 10
FX11 = 'YS-AP-103':DIN_VAL      ; 11
FX12 = 'YS-AP-104':DIN_VAL      ; 12
FX13 = 'YS-AP-105':DIN_VAL      ; 13
FX14 = 'YS-AP-106':DIN_VAL      ; 14
FX15 = 'YS-AP-107':DIN_VAL      ; 15
FX16 = 'YS-AP-108':DIN_VAL      ; 16
FX17 = 'YS-AP-109':DIN_VAL      ; 17
FX18 = 'YS-AP-110':DIN_VAL      ; 18
FX19 = 'PSHAP102M':DIN_VAL      ; 19
FX20 = 'PSHAP102M':DIN_VAL      ; 20
FX21 = 'YS-AP102R':DIN_VAL      ; 21
FX22 = 'PSHAP102M':DIN_VAL      ; 22
FX23 = 'PSHAP102M':DIN_VAL      ; 23
FX24 = 'PSHAP102M':DIN_VAL      ; 24
FX25 = 'YS-AP-21':DIN_VAL      ; 25
FX26 = 'PSHAP2711':DIN_VAL      ; 26
FX27 = 'PSH-APFP1':DIN_VAL      ; 27

```

MHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:04:09 08-Jan-96

Page 10

```

496 0277      FX28 = 'ZS-AP-FP1':DIN_VAL      ; 28
497 0278      FX29 = 'FSL-APK13':DIN_VAL      ; 29
498 0279      FX30 = 'FSL-APK12':DIN_VAL      ; 30
499 0280      FX31 = 'FSL-APK11':DIN_VAL      ; 31
500 0281      FX32 = 'FSL-APK23':DIN_VAL      ; 32
501 0282      FX33 = 'FSL-APK22':DIN_VAL      ; 33
502 0283      FX34 = 'FSL-APK21':DIN_VAL      ; 34
503 0284      FX35 = 'FSH-DR712':DIN_VAL      ; 35
504 0285      FX36 = 'FXS-DR712':DIN_VAL      ; 36 -mct3+
505 0286      FX37 = 'PSH-APFL1':DIN_VAL      ; 37
506 0287      FX38 = 'ZS-AP-DR1':DIN_VAL      ; 38
507 0288      FX39 = 'ZS-AP-FL1':DIN_VAL      ; 39
508 0289      FX40 = 'YS-AW-101':DIN_VAL      ; 40
509 0290      FX41 = 1                        ; 41 -mct5+
510 0291      FX42 = 'YS-AW-103':DIN_VAL      ; 42
511 0292      FX43 = 'YS-AW-104':DIN_VAL      ; 43
512 0293      FX44 = 'YS-AW-105':DIN_VAL      ; 44
513 0294      FX45 = 'YS-AW-106':DIN_VAL      ; 45
514 0295      FX46 = 'TXS-AW-2':DIN_VAL      ; 46
515      ;
516      ; deleted YS-AY-101, -102, and YS-AZ-101, -102: 47-50
517      ;
518 0296      FX51 = 'LSH-AX-1':DIN_VAL        ; 51
519 0297      FX52 = 'LSL-AX-1':DIN_VAL        ; 52
520      ; -mct2-
521      FX53 = AND('P-AN-101':DEV_STAT,16)    ; 53
522      FX54 = AND('P-AN-102':DEV_STAT,16)    ; 54
523      FX55 = AND('P-AN-103':DEV_STAT,16)    ; 55
524      FX56 = AND('P-AN-104':DEV_STAT,16)    ; 56
525      FX57 = AND('P-AN-105':DEV_STAT,16)    ; 57
526      FX58 = AND('P-AN-106':DEV_STAT,16)    ; 58
527      ; -mct2-
528      FX59 = AND('P-AN-107':DEV_STAT,16)    ; 59
529      FX60 = AND('P-AP-101':DEV_STAT,16)    ; 60
530      FX61 = AND('P-AP-102':DEV_STAT,16)    ; 61
531      FX62 = AND('P-AP-103':DEV_STAT,16)    ; 62
532      FX63 = AND('P-AP-104':DEV_STAT,16)    ; 63
533      FX64 = AND('P-AP-105':DEV_STAT,16)    ; 64
534      FX65 = AND('P-AP-106':DEV_STAT,16)    ; 65
535      FX66 = AND('P-AP-107':DEV_STAT,16)    ; 66
536      FX67 = AND('P-AP-108':DEV_STAT,16)    ; 67
537      FX68 = AND('P-AP-102M':DEV_STAT,16)   ; 68 jfb03
538      FX69 = AND('P-AP-102R':DEV_STAT,16)   ; 69 jfb03
539      ; -mct1-
540      FX70 = AND('P-AW-101':DEV_STAT,16)    ; 70
541      FX71 = AND('P-AW-102':DEV_STAT,16)    ; 71
542      FX72 = AND('P-AW-103':DEV_STAT,16)    ; 72
543      FX73 = AND('P-AW-104':DEV_STAT,16)    ; 73
544      FX74 = AND('P-AW-105':DEV_STAT,16)    ; 74
545      ; -mct1-
546      FX75 = AND('P-AW-106':DEV_STAT,16)    ; 75
547      FX76 = AND('P-AY-101':DEV_STAT,16)    ; 76
548      FX77 = AND('P-AY-102':DEV_STAT,16)    ; 77
549      FX78 = AND('P-AZ-101':DEV_STAT,16)    ; 78
550      FX79 = AND('P-AZ-102':DEV_STAT,16)    ; 79
551      FXB4 = 'PSH-FP1':DIN_VAL              ; 84 JFB05
552      ;
553      ;
554      IF FX90 = 0 THEN
555      IF 'T1-244-TK':AI_MEAS >= 200.0 THEN FX90 = 1      ; 90

```

```

551 0313      ELSE FX90 = 0
552 0315      ELSE IF 'TI-244-TK':AI_MEAS <= 200.0 - 'TI-244-TK':AI_DEAD THEN FX90 = 0
553      ;
554 0316      IF FX91 = 0 THEN
555 0318      IF 'FI-244EXH':AI_MEAS <= 100.0 THEN FX91 = 1          ; 91
556 0319      ELSE FX91 = 0
557 0321      ELSE IF 'FI-244EXH':AI_MEAS >= 100.0 + 'FI-244EXH':AI_DEAD THEN FX91 = 0
558      ;
559 0322      IF FX92 = 0 THEN
560 0324      IF 'FI-244EXH':AI_MEAS >= 200.0 THEN FX92 = 1          ; 92
561 0325      ELSE FX92 = 0
562 0327      ELSE IF 'FI-244EXH':AI_MEAS <= 200.0 - 'FI-244EXH':AI_DEAD THEN FX92 = 0
563      ;
564 0328      IF FX94 = 0 THEN
565 0330      IF 'II-AW-102':AI_MEAS <= 12.0 THEN FX94 = 1          ; 94
566 0331      ELSE FX94 = 0
567 0333      ELSE IF 'II-AW-102':AI_MEAS >= 12.0 + 'II-AW-102':AI_DEAD THEN FX94 = 0
568      ;
569      IF FX95 = 0 THEN
570      IF 'II-AW-102':AI_MEAS >= 25.0 THEN FX95 = 1          ; 95
571      ELSE FX95 = 0
572      ELSE IF 'II-AW-102':AI_MEAS <= 25.0 - 'II-AW-102':AI_DEAD THEN FX95 = 0
573      ;
574      IF FX96 = 0 THEN
575      IF 'DDI-CA1-1':AI_MEAS <= -0.05 THEN FX96 = 1          ; 96
576      ELSE FX96 = 0
577      ELSE IF 'DDI-CA1-1':AI_MEAS >= -0.05 + 'DDI-CA1-1':AI_DEAD THEN FX96 = 0
578      ;
579      IF FX97 = 0 THEN
580      IF 'DDI-CA1-1':AI_MEAS >= +0.05 THEN FX97 = 1          ; 97
581      ELSE FX97 = 0
582      ELSE IF 'DDI-CA1-1':AI_MEAS <= +0.05 - 'DDI-CA1-1':AI_DEAD THEN FX97 = 0
583      ;
584 0334      IF FX98 = 0 THEN
585 0336      IF 'PDI-FH1-1':AI_MEAS >= 15.0 THEN FX98 = 1          ; 98
586 0337      ELSE FX98 = 0
587 0339      ELSE IF 'PDI-FH1-1':AI_MEAS <= 15.0 - 'PDI-FH1-1':AI_DEAD THEN FX98 = 0
588      ;
589 0340      IF FX99 = 0 THEN
590 0342      IF 'PDI-FH2-1':AI_MEAS >= 15.0 THEN FX99 = 1          ; 99
591 0343      ELSE FX99 = 0
592 0345      ELSE IF 'PDI-FH2-1':AI_MEAS <= 15.0 - 'PDI-FH2-1':AI_DEAD THEN FX99 = 0
593      ;
594 0346      IF FX100 = 0 THEN
595 0348      IF 'PDI-FH3-1':AI_MEAS >= 5.0 THEN FX100 = 1          ; 100
596 0349      ELSE FX100 = 0
597 0351      ELSE IF 'PDI-FH3-1':AI_MEAS <= 5.0 - 'PDI-FH3-1':AI_DEAD THEN FX100 = 0
598      ;
599 0352      IF FX101 = 0 THEN
600 0354      IF 'SIC-PB2-1':AI_MEAS >= 95.0 THEN FX101 = 1          ; 101
601 0355      ELSE FX101 = 0
602 0357      ELSE IF 'SIC-PB2-1':AI_MEAS <= 95.0 - 'SIC-PB2-1':AI_DEAD THEN FX101 = 0
603      ;
604      FX102 = 'PSL-EA1-1':DIN_VAL          ; 102 JF804
605      ;

```

```

606 0358      IF FX103 = 0 THEN
607 0360      IF 'PI-EA1-13':AI_MEAS <= 75.0 THEN FX103 = 1          ; 103
608 0361      ELSE FX103 = 0
609 0363      ELSE IF 'PI-EA1-13':AI_MEAS >= 75.0 + 'PI-EA1-13':AI_DEAD THEN FX103 = 0
610
611 0364      ;
612 0366      IF FX104 = 0 THEN
613 0367      IF 'PI-EA1-13':AI_MEAS <= 80.0 THEN FX104 = 1          ; 104
614 0369      ELSE FX104 = 0
615      ELSE IF 'PI-EA1-13':AI_MEAS >= 80.0 + 'PI-EA1-13':AI_DEAD THEN FX104 = 0
616
617 0370      ;
618 0372      IF FX105 = 0 THEN
619 0373      IF 'FIC-EA1-1':AI_MEAS <= 3000.0 THEN FX105 = 1 ; 105
620 0375      ELSE FX105 = 0
621      ELSE IF 'FIC-EA1-1':AI_MEAS >= 3000.0 + 'FIC-EA1-14':AI_DEAD THEN FX105 = 0
622
623 0376      ;
624 0378      IF FX106 = 0 THEN
625 0379      IF 'PI-EA1-1':AI_MEAS <= 17.7 THEN FX106 = 1          ; 106
626 0381      ELSE FX106 = 0
627      ELSE IF 'PI-EA1-1':AI_MEAS >= 17.7 + 'PI-EA1-1':AI_DEAD THEN FX106 = 0
628
629 0382      ;
630 0384      IF FX107 = 0 THEN
631 0385      IF 'PI-EA1-1':AI_MEAS >= 29.7 THEN FX107 = 1          ; 107
632 0387      ELSE FX107 = 0
633      ELSE IF 'PI-EA1-1':AI_MEAS <= 29.7 - 'PI-EA1-1':AI_DEAD THEN FX107 = 0
634
635 0388      ;
636 0390      IF FX108 = 0 THEN
637 0391      IF 'TI-EA1-5':AI_MEAS >= 190.0 THEN FX108 = 1          ; 108
638 0393      ELSE FX108 = 0
639      ELSE IF 'TI-EA1-5':AI_MEAS <= 190.0 - 'TI-EA1-5':AI_DEAD THEN FX108 = 0
640
641 0394      ;
642 0396      IF FX109 = 0 THEN
643 0397      IF 'PI-EC1-5':AI_MEAS >= 100.0 THEN FX109 = 1          ; 109
644 0399      ELSE FX109 = 0
645      ELSE IF 'PI-EC1-5':AI_MEAS <= 100.0 - 'PI-EC1-5':AI_DEAD THEN FX109 = 0
646
647 0400      ;
648 0402      IF FX110 = 0 THEN
649 0403      IF 'FIC-EC3-1':AI_MEAS <= 50.0 THEN FX110 = 1          ; 110
650 0405      ELSE FX110 = 0
651      ELSE IF 'FIC-EC3-1':AI_MEAS >= 50.0 + 'FIC-EC3-1':AI_DEAD THEN FX110 = 0
652
653 0406      ;
654 0408      IF FX111 = 0 THEN
655 0409      IF 'TIC-EC1-2':AI_MEAS >= 110.0 THEN FX111 = 1          ; 111
656 0411      ELSE FX111 = 0
657      ELSE IF 'TIC-EC1-2':AI_MEAS <= 110.0 - 'TIC-EC1-2':AI_DEAD THEN FX111 = 0
658
659 0412      ;
660 0413      FX112 = 'LDS-BFP-1':DIN_VAL          ; 112
661 0414      FX113 = 'LSL-RWSMP':DIN_VAL          ; 113
662      FX114 = 'LDS-BFP-2':DIN_VAL          ; 114
663
664 0415      ;
665 0417      IF FX115 = 0 THEN
666 0418      IF 'PDI-RW-1':AI_MEAS >= 13.0 THEN FX115 = 1          ; 115
667 0420      ELSE FX115 = 0
668      ELSE IF 'PDI-RW-1':AI_MEAS <= 13.0 - 'PDI-RW-1':AI_DEAD THEN FX115 = 0
669
670 0421      ;
671      IF FX116 = 0 THEN

```



```

661 0423      IF 'PDI-RW-2':AI_MEAS >= 13.0 THEN FX116 = 1          ; 116
662 0424      ELSE FX116 = 0
663 0426      ELSE IF 'PDI-RW-2':AI_MEAS <= 13.0 - 'PDI-RW-2':AI_DEAD THEN FX116 = 0
664
665 0427      ; IF FX117 = 0 THEN
666 0429      IF 'PI-RW-2':AI_MEAS <= 60.0 THEN FX117 = 1          ; 117
667 0430      ELSE FX117 = 0
668 0432      ELSE IF 'PI-RW-2':AI_MEAS >= 60.0 + 'PI-RW-2':AI_DEAD THEN FX117 = 0
669
670      ; IF FX118 = 0 THEN
671      ; IF 'PI-RW-2':AI_MEAS >= 160.0 THEN FX118 = 1          ; 118
672      ; ELSE FX118 = 0
673      ; ELSE IF 'PI-RW-2':AI_MEAS <= 160.0 - 'PI-RW-2':AI_DEAD THEN FX118 = 0
674
675 0433      IF FX119 = 0 THEN
676 0435      IF 'FIC-EC1-1':AI_MEAS <= 1200.0 THEN FX119 = 1 ; 119
677 0436      ELSE FX119 = 0
678 0438      ELSE IF 'FIC-EC1-1':AI_MEAS >= 1200.0 + 'FIC-EC1-1':AI_DEAD THEN FX119 = 0
679
680      ; IF FX120 = 0 THEN
681      ; IF 'FIC-EC1-1':AI_MEAS >= 2800.0 THEN FX120 = 1 ; 120
682      ; ELSE FX120 = 0
683      ; ELSE IF 'FIC-EC1-1':AI_MEAS <= 2800.0 - 'FIC-EC1-1':AI_DEAD THEN FX120 = 0
684
685 0439      IF FX121 = 0 THEN
686 0441      IF 'II-EXC-1':AI_MEAS <= 1.0 THEN FX121 = 1          ; 121
687 0442      ELSE FX121 = 0
688 0444      ELSE IF 'II-EXC-1':AI_MEAS >= 1.0 + 'II-EXC-1':AI_DEAD THEN FX121 = 0
689
690 0445      IF FX126 = 0 THEN
691 0447      IF 'II-AS-1':AI_MEAS <= 2.0 THEN FX126 = 1          ; 126
692 0448      ELSE FX126 = 0
693 0450      ELSE IF 'II-AS-1':AI_MEAS >= 2.0 + 'II-AS-1':AI_DEAD THEN FX126 = 0
694
695 0451      ; FX128 = 'YS-A-C100':DIN_VAL          ; 128
696      ; FX129 = 'YS-P-C100':DIN_VAL          ; 129 JFB03
697 0452      FX130 = 'ZS-IDX-2':DIN_VAL          ; 130
698 0453      FX131 = 'PSH-ILF-1':DIN_VAL          ; 131
699 0454      FX132 = 'ZS-RC3-4':DIN_VAL          ; 132
700 0455      FX133 = 'ZS-RC3-5':DIN_VAL          ; 133
701
702 0456      IF FX135 = 0 THEN
703 0458      IF 'LI-C100-1':AI_MEAS <= 0.98 THEN FX135 = 1          ; 135
704 0459      ELSE FX135 = 0
705 0461      ELSE IF 'LI-C100-1':AI_MEAS >= 0.98 + 'LI-C100-1':AI_DEAD THEN FX135 = 0
706
707      ; IF FX136 = 0 THEN                                ;commented out, -mct5+
708      ; IF 'II-P-C100':AI_MEAS <= 5.0 THEN FX136 = 1          ; 136
709      ; ELSE FX136 = 0
710      ; ELSE IF 'II-P-C100':AI_MEAS >= 5.0 + 'II-P-C100':AI_DEAD THEN FX136 = 0
711
712 0462      IF FX137 = 0 THEN
713 0464      IF 'PDI-CA1-3':AI_MEAS >= 15.0 THEN FX137 = 1
714 0465      ELSE FX137 = 0
715 0467      ELSE IF 'PDI-CA1-3':AI_MEAS <= 15.0 - 'PDI-CA1-3':AI_DEAD THEN FX137 = 0

```

```

716      ;
717 0468      FX138 = 'LSL-SC-1':DIN_VAL      ; 138
718 0469      FX139 = 'LSL-SC-2':DIN_VAL      ; 139
719 0470      FX140 = 'LSL-SC-3':DIN_VAL      ; 140
720 0471      FX143 = 'YS-CPE1-1':DIN_VAL     ; 143
721 0472      FX144 = 'YS-CPE2-1':DIN_VAL     ; 144
722      ;
723 0473      IF FX145 = 0 THEN
724 0475      IF 'PI-IA-1':AI_MEAS <= 18.0 THEN FX145 = 1      ; 145
725 0476      ELSE FX145 = 0
726 0478      ELSE IF 'PI-IA-1':AI_MEAS >= 18.0 + 'PI-IA-1':AI_DEAD THEN FX145 = 0
727      ;
728 0479      IF FX146 = 0 THEN
729 0481      IF 'PI-IA-1':AI_MEAS >= 25.0 THEN FX146 = 1      ; 146
730 0482      ELSE FX146 = 0
731 0484      ELSE IF 'PI-IA-1':AI_MEAS <= 25.0 - 'PI-IA-1':AI_DEAD THEN FX146 = 0
732      ;
733      IF FX147 = 0 THEN
734      IF 'PI-PA-1':AI_MEAS <= 80.0 THEN FX147 = 1      ; 147
735      ELSE FX147 = 0
736      ELSE IF 'PI-PA-1':AI_MEAS >= 80.0 + 'PI-PA-1':AI_DEAD THEN FX147 = 0
737      ;
738 0485      IF FX148 = 0 THEN
739 0487      IF 'PI-PA-1':AI_MEAS >= 120.0 THEN FX148 = 1      ; 148
740 0488      ELSE FX148 = 0
741 0490      ELSE IF 'PI-PA-1':AI_MEAS <= 120.0 - 'PI-PA-1':AI_DEAD THEN FX148 = 0
742      ;
743 0491      IF FX149 = 0 THEN
744 0493      IF 'TI-AMBS-1':AI_MEAS <= 48.0 THEN FX149 = 1      ; 149
745 0494      ELSE FX149 = 0
746 0496      ELSE IF 'TI-AMBS-1':AI_MEAS >= 48.0 + 'TI-AMBS-1':AI_DEAD THEN FX149 = 0
747      ;
748 0497      IF FX150 = 0 THEN
749 0499      IF 'TI-AMBS-1':AI_MEAS <= 55.0 THEN FX150 = 1      ; 150
750 0500      ELSE FX150 = 0
751 0502      ELSE IF 'TI-AMBS-1':AI_MEAS >= 55.0 + 'TI-AMBS-1':AI_DEAD THEN FX150 = 0
752      ;
753 0503      IF FX151 = 0 THEN
754 0505      IF 'TI-AMBS-1':AI_MEAS >= 80.0 THEN FX151 = 1      ; 151
755 0506      ELSE FX151 = 0
756 0508      ELSE IF 'TI-AMBS-1':AI_MEAS <= 80.0 - 'TI-AMBS-1':AI_DEAD THEN FX151 = 0
757      ;
758 0509      IF FX152 = 0 THEN
759 0511      IF 'TI-AMBS-1':AI_MEAS >= 90.0 THEN FX152 = 1      ; 152
760 0512      ELSE FX152 = 0
761 0514      ELSE IF 'TI-AMBS-1':AI_MEAS <= 90.0 - 'TI-AMBS-1':AI_DEAD THEN FX152 = 0
762      ;
763 0515      IF FX153 = 0 THEN
764 0517      IF 'TI-AMBS-2':AI_MEAS <= 48.0 THEN FX153 = 1      ; 153
765 0518      ELSE FX153 = 0
766 0520      ELSE IF 'TI-AMBS-2':AI_MEAS >= 48.0 + 'TI-AMBS-2':AI_DEAD THEN FX153 = 0
767      ;
768 0521      IF FX154 = 0 THEN
769 0523      IF 'TI-AMBS-2':AI_MEAS <= 55.0 THEN FX154 = 1      ; 154
770 0524      ELSE FX154 = 0

```

```

771 0526      ELSE IF 'TI-AMBS-2':AI_MEAS >= 55.0 + 'TI-AMBS-2':AI_DEAD THEN FX154 = 0
772      ;
773 0527      IF FX155 = 0 THEN
774 0529      IF 'TI-AMBS-2':AI_MEAS >= 80.0 THEN FX155 = 1          ; 155
775 0530      ELSE FX155 = 0
776 0532      ELSE IF 'TI-AMBS-2':AI_MEAS <= 80.0 - 'TI-AMBS-2':AI_DEAD THEN FX155 = 0
777      ;
778 0533      IF FX156 = 0 THEN
779 0535      IF 'TI-AMBS-2':AI_MEAS >= 90.0 THEN FX156 = 1          ; 156
780 0536      ELSE FX156 = 0
781 0538      ELSE IF 'TI-AMBS-2':AI_MEAS <= 90.0 - 'TI-AMBS-2':AI_DEAD THEN FX156 = 0
782      ;
783 0539      FX157 = 'YS-A-E101':DIN_VAL          ; 157
784 0540      FX158 = 'YS-P-E101':DIN_VAL          ; 158
785 0541      FX159 = 'YS-A-E102':DIN_VAL          ; 159
786 0542      FX160 = 'YS-P-E102':DIN_VAL          ; 160
787 0543      FX161 = 'YS-A-E104':DIN_VAL          ; 161
788 0544      FX162 = 'YS-P-E104':DIN_VAL          ; 162
789      ;
790 0545      IF FX163 = 0 THEN
791 0547      IF 'TI-E101-2':AI_MEAS >= 150.0 THEN FX163 = 1          ; 163
792 0548      ELSE FX163 = 0
793 0550      ELSE IF 'TI-E101-2':AI_MEAS <= 150.0 - 'TI-E101-2':AI_DEAD THEN FX163 = 0
794      ;
795 0551      IF FX164 = 0 THEN
796 0553      IF 'WFI-E101':AI_MEAS <= 15.0 THEN FX164 = 1          ; 164
797 0554      ELSE FX164 = 0
798 0556      ELSE IF 'WFI-E101':AI_MEAS >= 15.0 + 'WFI-E101':AI_DEAD THEN FX164 = 0
799      ;
800 0557      IF FX165 = 0 THEN
801 0559      IF 'WFI-E101':AI_MEAS <= 25.0 THEN FX165 = 1          ; 165
802 0560      ELSE FX165 = 0
803 0562      ELSE IF 'WFI-E101':AI_MEAS >= 25.0 + 'WFI-E101':AI_DEAD THEN FX165 = 0
804      ;
805 0563      IF FX166 = 0 THEN
806 0565      IF 'WFI-E102':AI_MEAS <= 5.0 THEN FX166 = 1          ; 166
807 0566      ELSE FX166 = 0
808 0568      ELSE IF 'WFI-E102':AI_MEAS >= 5.0 + 'WFI-E102':AI_DEAD THEN FX166 = 0
809      ;
810 0569      IF FX167 = 0 THEN
811 0571      IF 'WFI-E102':AI_MEAS <= 12.0 THEN FX167 = 1          ; 167
812 0572      ELSE FX167 = 0
813 0574      ELSE IF 'WFI-E102':AI_MEAS >= 12.0 + 'WFI-E102':AI_DEAD THEN FX167 = 0
814      ;
815 0575      IF FX168 = 0 THEN
816 0577      IF 'TI-E104-2':AI_MEAS >= 170.0 THEN FX168 = 1          ; 168
817 0578      ELSE FX168 = 0
818 0580      ELSE IF 'TI-E104-2':AI_MEAS <= 170.0 - 'TI-E104-2':AI_DEAD THEN FX168 = 0
819      ;
820 0581      IF FX169 = 0 THEN
821 0583      IF 'WFI-E104':AI_MEAS <= 5 THEN FX169 = 1          ; 169
822 0584      ELSE FX169 = 0
823 0586      ELSE IF 'WFI-E104':AI_MEAS >= 5 + 'WFI-E104':AI_DEAD THEN FX169 = 0
824      ;
825 0587      IF FX170 = 0 THEN

```

```

826 0589 IF 'WFI-E104':AI_MEAS <= 12.0 THEN FX170 = 1 ; 170
827 0590 ELSE FX170 = 0
828 0592 ELSE IF 'WFI-E104':AI_MEAS >= 12.0 + 'WFI-E104':AI_DEAD THEN FX170 = 0
829 ;
830 ;
831 ;
832 0593 FX171 = 'ZS-105PS1':DIM_VAL ; 171 JFB02
833 0594 FX172 = 'ZS-105PS2':DIM_VAL ; 172 JFB02
834 0595 FX173 = 'YS-P-42':DIM_VAL ; 173 +mct4-
835 0596 FX174 = 'YS-P-43':DIM_VAL ; 174 +mct4-
836 0597 FX175 = 'YS-P-44':DIM_VAL ; 175 +mct4-
837 0598 FX176 = 'YS-P-C106':DIM_VAL ; 176 JFB01
838 0599 FX177 = AND('HV-CA1-10':DEV_STAT,64) ; 177 JFB01
839 ;
840 ;
841 0598 LOP: ASSIGN PSTAT = FX1
842 0599 ASSIGN PALARM = FAL1
843 0600 ASSIGN PREV = FRV1
844 0601 ASSIGN PTYPE = FDS1
845 0602 EPN3 = "LOP:"
846 0603 IN1 = TIMSTR(TIME3)
847 ;
848 0604 FOR ILOOP = 1 TO 177 ; +mct4-
849 0605 IF I(RTYPE)THEN ; IF AN ACTIVE ALARM (not inha)
850 BEGIN
851 IF(PALARM)THEN ; IF NOT PREVIOUSLY ALARMED
852 BEGIN
853 IF PREV = PSTAT THEN GOSUB ALM ; IF ALARM CONDITION MET
854 END
855 ELSE IF PREV=PSTAT THEN GOSUB CLR ; IF ALARM CLEARED
856 END
857 ELSE
858 IF(PALARM)THEN GOSUB CLR
859 STEP PSTAT BY 1
860 STEP PALARM BY 1
861 STEP PREV BY 1
862 STEP PTYPE BY 1
863 NEXT ILOOP
864 ;
865 0619 IF IN4 > 16 THEN FSPAR4 = 1 ; ""MORE" ON GRAPHIC 100
866 0620 ELSE FSPAR4 = 0
867 0622 IF IN4 > 32 THEN FSPAR5 = 1 ; ""MORE" ON GRAHIC 101
868 0623 ELSE FSPAR5 = 0
869 0624 WAIT 2
870 0625 GOTO AGAIN
871 ;
872 ;
873 ;
874 ;
875 ;
876 0626 CLR: EPN4 = "CLR" ; SET PALARM = 0, SET PLINE = 0
877 ; MOVE LOWER LINES DOWN
878 0627 PALARM = 0
879 0628 PRINT 0," --- CLEAR ALARM, ILOOP = %i",ILOOP
880 0629 IN1 = TIMSTR(TIME4)

```

```

881 0630      ASSIGN PLINE = INTO
882 0631      STEP PLINE BY ILOOP
883 0632      IF PLINE = 0 THEN
884          begin
885 0633          PRINT 0,"      ***  W A R N I N G  PLINE = 0, ILOOP = %i",ILOOP
886 0634          LAMP 0,60,1
887 0635          goto term
888          end
889 0636      IN4 = IN4 - 1
890 0637      IN2 = ILOOP
891 0638      FIRST = 46 - PLINE - 1
892 0639      STPPER = 1
893      ;
894 0640      ASSIGN NXTGRA = GRAFX0
895 0641      ASSIGN PGRAFX = GRAFX1
896      ;
897 0642      ASSIGN NXTLSG = LSG0
898 0643      ASSIGN PLRGSG = LSG1
899      ;
900 0644      ASSIGN NXTMSG = MSSG0
901 0645      ASSIGN PMESSG = MSSG1
902      ;
903 0646      STEP NXTGRA BY (PLINE - 1)
904 0647      STEP PGRAFX BY (PLINE - 1)
905 0648      STEP NXTMSG BY PLINE - 1
906 0649      STEP PMESSG BY PLINE - 1
907 0650      STEP NXTLSG BY PLINE - 1
908 0651      STEP PLRGSG BY PLINE - 1
909 0652      GOSUB SRT
910      ;
911 0653      LAST = PLINE
912 0654      ASSIGN PLINE = INT1
913      ;
914 0655      FOR KLOOP = 1 TO 177      ; +mct4-
915 0656      IF KLOOP = ILOOP THEN
916          BEGIN
917 0657          PLINE = 0
918          PRINT 0,"      ***  PLINE NOW = 0, KLOOP = %i",KLOOP
919          END
920 0658      ELSE IF PLINE > LAST THEN
921          BEGIN
922 0659          PLINE = PLINE - 1      ***  (PLINE - 1), KLOOP = %i, PLINE = %i",KLOOP,pline
923          PRINT 0,"
924 0661          if pline = 0 then goto term
925          END
926 0662      STEP PLINE BY 1
927 0663      NEXT KLOOP
928 0664      RETURN
929      ;
930      ;
931      ;
932      ;
933      ;
934 0665      SRT:      EPN3 = "SRT"
935 0666      IN1 = TIMSTR(TIME3)

```

```

936 0667      FOR BVAL = 0 TO FIRST
937 0668      NXTGRA = PGRAF4
938 0669      NXTMSG = PMESSG
939 0670      NXTLSG = PLRSGG
940 0671      STEP NXTGRA BY STPPER
941 0672      STEP PGRAF4 BY STPPER
942 0673      STEP NXTMSG BY STPPER
943 0674      STEP PMESSG BY STPPER
944 0675      STEP NXTLSG BY STPPER
945 0676      STEP PLRSGG BY STPPER
946 0677      NEXT BVAL
947 0678      RETURN
948
949      ;
950      ;
951      ;
952      ;
953 0679      ALM:      EPN4 = "ALM:"
954 0680      IN4 = IN4 + 1
955 0681      IN1 = TIMSTR(TIME4)
956 0682      IN2 = ILOOP
957 0683      PALARM = 1
958      ;
959      ;
960 0684      LAMP 0,60,3
961 0685      FIRST = 45
962 0686      STPPER = -1
963 0687      ASSIGN NXTGRA = GRAF46
964 0688      ASSIGN PGRAF4 = GRAF45
965 0689      ASSIGN NXTLSG = LSG46
966 0690      ASSIGN PLRSGG = LSG45
967 0691      ASSIGN NXTMSG = MSSG46
968 0692      ASSIGN PMESSG = MSSG45
969 0693      GOSUB SRT
970
971      ;
972      ;
973 0694      ON ILOOP GOTO S1,S2,S3,S4,S5,S6,S7,S8,S9,S10
974 0695      ON ILOOP - 10 GOTO S11,S12,S13,S14,S15,S16,S17,S18,S19,S20
975 0696      ON ILOOP - 20 GOTO S21,S22,S23,S24,S25,S26,S27,S28,S29,S30
976 0697      ON ILOOP - 30 GOTO S31,S32,S33,S34,S35,S36,S37,S38,S39,S40
977 0698      ON ILOOP - 40 GOTO S41,S42,S43,S44,S45,S46,S47,S48,S49,S50
978 0699      ON ILOOP - 50 GOTO S51,S52,S53,S54,S55,S56,S57,S58,S59,S60
979 0700      ON ILOOP - 60 GOTO S61,S62,S63,S64,S65,S66,S67,S68,S69,S70
980 0701      ON ILOOP - 70 GOTO S71,S72,S73,S74,S75,S76,S77,S78,S79,S80
981 0702      ON ILOOP - 80 GOTO S81,S82,S83,S84,S85,S86,S87,S88,S89,S90
982 0703      ON ILOOP - 90 GOTO S91,S92,S93,S94,S95,S96,S97,S98,S99,S100
983 0704      ON ILOOP - 100 GOTO S101,S102,S103,S104,S105,S106,S107,S108,S109,S110
984 0705      ON ILOOP - 110 GOTO S111,S112,S113,S114,S115,S116,S117,S118,S119,S120
985 0706      ON ILOOP - 120 GOTO S121,S122,S123,S124,S125,S126,S127,S128,S129,S130
986 0707      ON ILOOP - 130 GOTO S131,S132,S133,S134,S135,S136,S137,S138,S139,S140
987 0708      ON ILOOP - 140 GOTO S141,S142,S143,S144,S145,S146,S147,S148,S149,S150
988 0709      ON ILOOP - 150 GOTO S151,S152,S153,S154,S155,S156,S157,S158,S159,S160
989 0710      ON ILOOP - 160 GOTO S161,S162,S163,S164,S165,S166,S167,S168,S169,S170
990 0711      ON ILOOP - 170 GOTO S171,S172,S173,S174,S175,S176,S177
991 0712      GOTO AMSG

```

```

991      ;
992      0713 S1:  MSSGO = "AN-101 PUMP OFF"      "
993      0714      EPNO = "YS-AN-101"
994      0715      GRAFX0 = 270
995      0716      GOTO AMSG
996      0717 S2:  MSSGO = "AN-102 PUMP OFF"      "
997      0718      EPNO = "YS-AN-102"
998      0719      GRAFX0 = 270
999      0720      GOTO AMSG
1000     0721 S3:  MSSGO = "AN-103 PUMP OFF"      "
1001     0722      EPNO = "YS-AN-103"
1002     0723      GRAFX0 = 270
1003     0724      GOTO AMSG
1004     0725 S4:  MSSGO = "AN-104 PUMP OFF"      "
1005     0726      EPNO = "YS-AN-104"
1006     0727      GRAFX0 = 270
1007     0728      GOTO AMSG
1008     0729 S5:  MSSGO = "AN-105 PUMP OFF"      "
1009     0730      EPNO = "YS-AN-105"
1010     0731      GRAFX0 = 270
1011     0732      GOTO AMSG
1012     0733 S6:  MSSGO = "AN-106 PUMP OFF"      "
1013     0734      EPNO = "YS-AN-106"
1014     0735      GRAFX0 = 270
1015     0736      GOTO AMSG
1016     0737 S7:  MSSGO = "AN-107 PUMP OFF"      "
1017     0738      EPNO = "YS-AN-107"
1018     0739      GRAFX0 = 270
1019     0740      GOTO AMSG
1020     0741 SB:  MSSGO = "AN-FARM HEAT TRACE OVERTEMP"
1021     0742      EPNO = "TXS-AN-2"
1022     0743      GRAFX0 = 274
1023     0744      GOTO AMSG
1024     0745 S9:  MSSGO = "AP-101 PUMP MCC OFF"    "
1025     0746      EPNO = "YS-AP-101"
1026     0747      GRAFX0 = 290
1027     0748      GOTO AMSG
1028     0749 S10: MSSGO = "AP-102 PUMP MCC OFF"    "
1029     0750      EPNO = "YS-AP-102"
1030     0751      GRAFX0 = 290
1031     0752      GOTO AMSG
1032     0753 S11: MSSGO = "AP-103 PUMP MCC OFF"    "
1033     0754      EPNO = "YS-AP-103"
1034     0755      GRAFX0 = 290
1035     0756      GOTO AMSG
1036     0757 S12: MSSGO = "AP03D DRAIN PIT LEAK"    "
1037     0758      EPNO = "LDS-AP03D"
1038     0759      GRAFX0 = 290
1039     0760      GOTO AMSG
1040     0761 S13: MSSGO = "AP-104 PUMP MCC OFF"    "
1041     0762      EPNO = "YS-AP-104"
1042     0763      GRAFX0 = 290
1043     0764      GOTO AMSG
1044     0765 S14: MSSGO = "AP-105 PUMP MCC OFF"    "
1045     0766      EPNO = "YS-AP-105"

```

```

1046 0767      GRAFX0 = 290
1047 0768      GOTO AMSG
1048 0769 S15:   MSSGO = "AP-106 PUMP MCC OFF      "
1049 0770      EPNO = "YS-AP-106"
1050 0771      GRAFX0 = 290
1051 0772      GOTO AMSG
1052 0773 S16:   MSSGO = "AP-107 PUMP MCC OFF      "
1053 0774      EPNO = "YS-AP-107"
1054 0775      GRAFX0 = 290
1055 0776      GOTO AMSG
1056 0777 S17:   MSSGO = "AP-108 PUMP MCC OFF      "
1057 0778      EPNO = "YS-AP-108"
1058 0779      GRAFX0 = 290
1059 0780      GOTO AMSG
1060      ;
1061      ; MSSGO = "MIXER PUMP AP-102-3 MCC OFF" jfb03
1062      ; EPNO = "YS-AP102M"
1063      ; GRAFX0 = 291
1064      ; GOTO AMSG
1065 S19:   ; MSSGO = "MIXER PUMP AP-102-3 PRESS HI" jfb03
1066      ; EPNO = "PSHAP102M"
1067      ; GRAFX0 = 291
1068      ; GOTO AMSG
1069 S20:   ; MSSGO = "MIXER PUMP AP-102-3 PRESS LOW" jfb03
1070      ; EPNO = "PSLAP102M"
1071      ; GRAFX0 = 291
1072      ; GOTO AMSG
1073 S21:   ; MSSGO = "AP-102-3 ROTATOR MOTR MCC OFF " jfb03
1074      ; EPNO = "YS-AP102R"
1075      ; GRAFX0 = 291
1076      ; GOTO AMSG
1077 S22:   ; MSSGO = "MIXER PUMP AP-102-3 STRAIN HI " jfb03
1078      ; EPNO = "WSHAP102M"
1079      ; GRAFX0 = 291
1080      ; GOTO AMSG
1081 S23:   ; MSSGO = "MIXER PUMP AP-102-3 TEMP HI " jfb03
1082      ; EPNO = "TSHAP102M"
1083      ; GRAFX0 = 291
1084      ; GOTO AMSG
1085 S24:   ; MSSGO = "AP-102-3 VIBRATION HIGH      " jfb03
1086      ; EPNO = "VSHAP102M"
1087      ; GRAFX0 = 291
1088      ; GOTO AMSG
1089 0781 S25:   MSSGO = "AP-FARM HEAT TRACE OVER TEMP "
1090 0782      EPNO = "TXS-AP-2"
1091 0783      GRAFX0 = 295
1092 0784      GOTO AMSG
1093 0785 S26:   MSSGO = "271-AP BUILDING CAM FLOW LOW "
1094 0786      EPNO = "FSLAP2711"
1095 0787      GRAFX0 = 295
1096 0788      GOTO AMSG
1097 0789 S27:   MSSGO = "AP-FLSPIT FLUSH LINE PRESS HI "
1098 0790      EPNO = "PSH-APFP1"
1099 0791      GRAFX0 = 292
1100 0792      GOTO AMSG
1100 0793 S28:   MSSGO = "AP-FARM VALVE PIT V-FP-1 OPEN "

```



```

1101 0794      EPNO = "ZS-AP-FP1"
1102 0795      GRAFX0 = 292
1103 0796      GOTO AMSG
1104 0797      S29:  MSSGO = "AP-FARM K1 EXH MON SMPL FLO LO"
1105 0798      EPNO = "FSL-APK13"
1106 0799      GRAFX0 = 294
1107 0800      GOTO AMSG
1108 0801      S30:  MSSGO = "AP-FARM K1 EXH REC SMPL FLO LO"
1109 0802      EPNO = "FSL-APK12"
1110 0803      GRAFX0 = 294
1111 0804      GOTO AMSG
1112 0805      S31:  MSSGO = "AP-FARM K1 EXH STACK FLOW LOW "
1113 0806      EPNO = "FSL-APK11"
1114 0807      GRAFX0 = 294
1115 0808      GOTO AMSG
1116 0809      S32:  MSSGO = "AP-FARM K2 EXH MON SMPL FLO LO"
1117 0810      EPNO = "FSL-APK23"
1118 0811      GRAFX0 = 294
1119 0812      GOTO AMSG
1120 0813      S33:  MSSGO = "AP-FARM K2 EXH REC SMPL FLO LO"
1121 0814      EPNO = "FSL-APK22"
1122 0815      GRAFX0 = 294
1123 0816      GOTO AMSG
1124 0817      S34:  MSSGO = "AP-FARM K2 EXH STACK FLOW LOW "
1125 0818      EPNO = "FSL-APK21"
1126 0819      GRAFX0 = 294
1127 0820      GOTO AMSG
1128 0821      S35:  MSSGO = "AP-FARM VLVPIIT DR-712 FLOW HI "
1129 0822      EPNO = "FSH-DR712"
1130 0823      GRAFX0 = 293
1131 0824      GOTO AMSG
1132 0825      S36:  MSSGO = "AP-VLVPIT DR-712 FLO DET FAIL "
1133 0826      EPNO = "FXS-DR712"
1134 0827      GRAFX0 = 293
1135 0828      GOTO AMSG
1136 0829      S37:  MSSGO = "AP-VLVPIT FLUSH LINE PRES HIGH"
1137 0830      EPNO = "PSH-APFL1"
1138 0831      GRAFX0 = 293
1139 0832      GOTO AMSG
1140 0833      S38:  MSSGO = "AP-FARM VALVE PIT V-DR-1 OPEN "
1141 0834      EPNO = "ZS-AP-DR1"
1142 0835      GRAFX0 = 293
1143 0836      GOTO AMSG
1144 0837      S39:  MSSGO = "AP-FARM VALVE PIT V-FL-1 OPEN "
1145 0838      EPNO = "ZS-AP-FL1"
1146 0839      GRAFX0 = 293
1147 0840      GOTO AMSG
1148 0841      S40:  MSSGO = "AW-101 PUMP OFF          "
1149 0842      EPNO = "YS-AW-101"
1150 0843      GRAFX0 = 300
1151 0844      GOTO AMSG
1152      S41:  ;MSSGO = "AW-102 PUMP OFF          "      commented out, -mct5+
1153      ;      EPNO = "YS-AW102"
1154      ;      GRAFX0 = 301
1155      ;      GOTO AMSG

```

1156	0845	S42:	MSGO = "AN-103 PUMP OFF	"
1157	0846		EPNO = "YS-AN-103"	
1158	0847		GRAFXO = 300	
1159	0848		GOTO ANSG	"
1160	0849	S43:	MSGO = "AN-104 PUMP OFF	
1161	0850		EPNO = "YS-AN-104"	
1162	0851		GRAFXO = 300	
1163	0852		GOTO ANSG	"
1164	0853	S44:	MSGO = "AN-105 PUMP OFF	
1165	0854		EPNO = "YS-AN-105"	
1166	0855		GRAFXO = 300	
1167	0856		GOTO ANSG	"
1168	0857	S45:	MSGO = "AN-106 PUMP OFF	
1169	0858		EPNO = "YS-AN-106"	
1170	0859		GRAFXO = 300	
1171	0860		GOTO ANSG	"
1172	0861	S46:	MSGO = "AN-FARM HEAT TRACE OVERTEMP	
1173	0862		EPNO = "YS-AN-2"	
1174	0863		GRAFXO = 302	
1175	0864		GOTO ANSG	"
1176		S47:	MSGO = "AY-101 PUMP MCC OFF	
1177			EPNO = "YS-AY-101"	
1178			GRAFXO = 322	
1179			GOTO ANSG	"
1180		S48:	MSGO = "AY-102 PUMP MCC OFF	
1181			EPNO = "YS-AY-102"	
1182			GRAFXO = 322	
1183			GOTO ANSG	"
1184		S49:	MSGO = "AZ-101 PUMP MCC OFF	
1185			EPNO = "YS-AZ-101"	
1186			GRAFXO = 323	
1187			GOTO ANSG	"
1188		S50:	MSGO = "AZ-102 PUMP MCC OFF	
1189			EPNO = "YS-AZ-102"	
1190			GRAFXO = 323	
1191			GOTO ANSG	"
1192	0845	S51:	MSGO = "AX-A/B SEAL POT LEVEL HIGH	
1193	0846		EPNO = "LSH-AX-1"	
1194	0847		GRAFXO = 320	
1195	0848		GOTO ANSG	"
1196	0849	S52:	MSGO = "AX-A/B SEAL POT LEVEL LOW	
1197	0870		EPNO = "LSI-AX-1"	
1198	0871		GRAFXO = 320	
1199	0872		GOTO ANSG	"
1200		--nct2-		
1201		S53:	MSGO = "PUMP AN-101 SHUTDOWN"	"
1202			EPNO = "P-AN-101"	
1203			GRAFXO = 270	
1204			GOTO ANSG	"
1205		S54:	MSGO = "PUMP AN-102 SHUTDOWN	
1206			EPNO = "P-AN-102"	
1207			GRAFXO = 270	
1208			GOTO ANSG	"
1209		S55:	MSGO = "PUMP AN-103 SHUTDOWN	
1210			EPNO = "P-AN-103"	

```

1211      ;      GRAFX0 = 270
1212      ;      GOTO AMSG
1213      S56:      ;      MSSGO = "PUMP AN-104 SHUTDOWN"      "
1214      ;      EPNO = "P-AN-104"
1215      ;      GRAFX0 = 270
1216      ;      GOTO AMSG
1217      S57:      ;      MSSGO = "PUMP AN-105 SHUTDOWN"      "
1218      ;      EPNO = "P-AN-105"
1219      ;      GRAFX0 = 270
1220      ;      GOTO AMSG
1221      S58:      ;      MSSGO = "PUMP AN-106 SHUTDOWN"      "
1222      ;      EPNO = "P-AN-106"
1223      ;      GRAFX0 = 270
1224      ;      GOTO AMSG
1225      S59:      ;      MSSGO = "PUMP AN-107 SHUTDOWN"      "
1226      ;      EPNO = "P-AN-107"
1227      ;      GRAFX0 = 270
1228      ;      GOTO AMSG
1229      ; -mct2-
1230      0873      S60:      MSSGO = "PUMP AP-101 SHUTDOWN"      "
1231      0874      ;      EPNO = "P-AP-101"
1232      0875      ;      GRAFX0 = 290
1233      0876      ;      GOTO AMSG
1234      0877      S61:      MSSGO = "PUMP AP-102 SHUTDOWN"      "
1235      0878      ;      EPNO = "P-AP-102"
1236      0879      ;      GRAFX0 = 291
1237      0880      ;      GOTO AMSG
1238      0881      S62:      MSSGO = "PUMP AP-103 SHUTDOWN"      "
1239      0882      ;      EPNO = "P-AP-103"
1240      0883      ;      GRAFX0 = 290
1241      0884      ;      GOTO AMSG
1242      0885      S63:      MSSGO = "PUMP AP-104 SHUTDOWN"      "
1243      0886      ;      EPNO = "P-AP-104"
1244      0887      ;      GRAFX0 = 290
1245      0888      ;      GOTO AMSG
1246      0889      S64:      MSSGO = "PUMP AP-105 SHUTDOWN"      "
1247      0890      ;      EPNO = "P-AP-105"
1248      0891      ;      GRAFX0 = 290
1249      0892      ;      GOTO AMSG
1250      0893      S65:      MSSGO = "PUMP AP-106 SHUTDOWN"      "
1251      0894      ;      EPNO = "P-AP-106"
1252      0895      ;      GRAFX0 = 290
1253      0896      ;      GOTO AMSG
1254      0897      S66:      MSSGO = "PUMP AP-107 SHUTDOWN"      "
1255      0898      ;      EPNO = "P-AP-107"
1256      0899      ;      GRAFX0 = 290
1257      0900      ;      GOTO AMSG
1258      0901      S67:      MSSGO = "PUMP AP-108 SHUTDOWN"      "
1259      0902      ;      EPNO = "P-AP-108"
1260      0903      ;      GRAFX0 = 290
1261      0904      ;      GOTO AMSG
1262      ;      S68:      ;      MSSGO = "MIXER PUMP AP-102M SHUTDOWN"      " jfb03
1263      ;      ;      EPNO = "P-AP-102M"
1264      ;      ;      GRAFX0 = 291
1265      ;      ;      GOTO AMSG

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:04:09 08-Jan-96

Page 24

```

1266      S69:      ;      MSSGO = "ROTOR AP-102R SHUTDOWN      " jfb03
1267      ;      EPNO = "P-AP-102R"
1268      ;      GRAFXO = 291
1269      ;      GOTO AMSG
1270      ;-mct1-
1271      S70:      ;      MSSGO = "PUMP AW-101 SHUTDOWN      "
1272      ;      EPNO = "P-AW-101"
1273      ;      GRAFXO = 300
1274      ;      GOTO AMSG
1275      S71:      ;      MSSGO = "PUMP AW-102 SHUTDOWN      "
1276      ;      EPNO = "P-AW-102"
1277      ;      GRAFXO = 301
1278      ;      GOTO AMSG
1279      S72:      ;      MSSGO = "PUMP AW-103 SHUTDOWN      "
1280      ;      EPNO = "P-AW-103"
1281      ;      GRAFXO = 300
1282      ;      GOTO AMSG
1283      S73:      ;      MSSGO = "PUMP AW-104 SHUTDOWN      "
1284      ;      EPNO = "P-AW-104"
1285      ;      GRAFXO = 300
1286      ;      GOTO AMSG
1287      S74:      ;      MSSGO = "PUMP AW-105 SHUTDOWN      "
1288      ;      EPNO = "P-AW-105"
1289      ;      GRAFXO = 300
1290      ;      GOTO AMSG
1291      S75:      ;      MSSGO = "PUMP AW-106 SHUTDOWN      "
1292      ;      EPNO = "P-AW-106"
1293      ;      GRAFXO = 300
1294      ;      GOTO AMSG
1295      ;-mct1-
1296      0905      S76:      MSSGO = "PUMP AY-101 SHUTDOWN      "
1297      0906      ;      EPNO = "P-AY-101"
1298      0907      ;      GRAFXO = 322
1299      0908      ;      GOTO AMSG
1300      0909      S77:      MSSGO = "PUMP AY-102 SHUTDOWN      "
1301      0910      ;      EPNO = "P-AY-102"
1302      0911      ;      GRAFXO = 322
1303      0912      ;      GOTO AMSG
1304      0913      S78:      MSSGO = "PUMP AZ-101 SHUTDOWN      "
1305      0914      ;      EPNO = "P-AZ-101"
1306      0915      ;      GRAFXO = 323
1307      0916      ;      GOTO AMSG
1308      0917      S79:      MSSGO = "PUMP AZ-102 SHUTDOWN      "
1309      0918      ;      EPNO = "P-AZ-102"
1310      0919      ;      GRAFXO = 323
1311      0920      ;      GOTO AMSG
1312      S80:      ;      MSSGO = "AW K1 STACK MON SMPL PUMP OFF "
1313      ;      EPNO = "PSH-AWK11"
1314      ;      GRAFXO = 304
1315      ;      GOTO AMSG
1316      S81:      ;      MSSGO = "AW K1 RECORD SAMPLE PUMP OFF "
1317      ;      EPNO = "PSH-AWK12"
1318      ;      GRAFXO = 304
1319      ;      GOTO AMSG
1320      S82:      ;      MSSGO = "AW K2 STACK MON SMPL PUMP OFF "

```

```

1321      ;      EPNO = "PSH-AWK21"
1322      ;      GRAFXO = 304
1323      ;      GOTO MSG
1324      S83:    ;      MSSGO = "AW K2 RECORD SAMPLE PUMP OFF "
1325      ;      EPNO = "PSH-AWK22"
1326      ;      GRAFXO = 304
1327      ;      GOTO MSG
1328      S84:    ;      MSSGO = "244-A FLUSH PIT PRESS HIGH " JFB05
1329      ;      EPNO = "PSH-FP"
1330      ;      GRAFXO = 340
1331      ;      GOTO MSG
1332      S85:    ;      MSSGO = "TK-244-A PUMP 1 CURRENT HIGH "
1333      ;      EPNO = "I1-P244A1"
1334      ;      GRAFXO = 340
1335      ;      GOTO MSG
1336      S86:    ;      MSSGO = "TK-244-A PUMP 2 CURRENT HIGH "
1337      ;      EPNO = "I1-P244A2"
1338      ;      GRAFXO = 340
1339      ;      GOTO MSG
1340      S87:    ;      MSSGO = "244-A SUMP PUMP CURRENT HIGH "
1341      ;      EPNO = "I1-P244A3"
1342      ;      GRAFXO = 340
1343      ;      GOTO MSG
1344      S88:    ;      MSSGO = "244-A SUMP PUMP CURRENT HIHI "
1345      ;      EPNO = "I1-P244A3"
1346      ;      GRAFXO = 340
1347      ;      GOTO MSG
1348      S89:    ;      MSSGO = "TK-244-A AGITATOR CURRENT HI "
1349      ;      EPNO = "I1-A-244A"
1350      ;      GRAFXO = 340
1351      ;      GOTO MSG
1352      0921   S90:    MSSGO = "TK-244-A LIQUID TEMPATURE HI "
1353      0922      EPNO = "T1-244-TK"
1354      0923      GRAFXO = 340
1355      0924      GOTO MSG
1356      S91:    MSSGO = "244A-EXHAUST STACK FLOW LOW "
1357      0926      EPNO = "F1-244EXH"
1358      0927      GRAFXO = 341
1359      0928      GOTO MSG
1360      0929   S92:    MSSGO = "244A-EXHAUST STACK FLOW HIGH "
1361      0930      EPNO = "F1-244EXH"
1362      0931      GRAFXO = 341
1363      0932      GOTO MSG
1364      S93:    ;      MSSGO = "TK-A-350 PUMP CURRENT HIGH "
1365      ;      EPNO = "I1-350-1"
1366      ;      GRAFXO = 343
1367      ;      GOTO MSG
1368      0933   S94:    MSSGO = "AW-102 FEED PUMP CURRENT LOW "
1369      0934      EPNO = "I1-AW-102"
1370      0935      GRAFXO = 301
1371      0936      GOTO MSG
1372      S95:    ;      MSSGO = "AW-102 FEED PUMP CURRENT HIGH"
1373      ;      EPNO = "I1-AW-102"
1374      ;      GRAFXO = 301
1375      ;      GOTO MSG

```

```

1376      S96:      ;      MSSGO = "EVAP DENSITY DIFFERENCE LOLO "
1377      ;      EPNO = "DD1-CA1-1"
1378      ;      GRAFX0 = 10
1379      ;      GOTO AMSG
1380      S97:      ;      MSSGO = "EVAP DENSITY DIFFERENCE HIHI "
1381      ;      EPNO = "DD1-CA1-1"
1382      ;      GRAFX0 = 10
1383      ;      GOTO AMSG
1384      0937      S98:      MSSGO = "F-H-1 RAW WATER FILTER DP HIGH"
1385      0938      EPNO = "PD1-FH1-1"
1386      0939      GRAFX0 = 10
1387      0940      GOTO AMSG
1388      0941      S99:      MSSGO = "F-H-2 RAW WATER FILTER DP HIGH"
1389      0942      EPNO = "PD1-FH2-1"
1390      0943      GRAFX0 = 10
1391      0944      GOTO AMSG
1392      0945      S100:     MSSGO = "F-H-3 RAW WATER FILTER DP HIGH"
1393      0946      EPNO = "PD1-FH3-1"
1394      0947      GRAFX0 = 10
1395      0948      GOTO AMSG
1396      0949      S101:     MSSGO = "SLURRY PUMP SPEED CNTRLR HIGH "
1397      0950      EPNO = "S1C-PB2-1"
1398      0951      GRAFX0 = 15
1399      0952      GOTO AMSG
1400      S102:     ;      MSSGO = "REBOILER 10 LB STM PRESS LOW " JFB04
1401      ;      EPNO = "PSL-EA1-1"
1402      ;      GRAFX0 = 13
1403      ;      GOTO AMSG
1404      0953      S103:     MSSGO = "90 PSI STEAM HEADER PRESS LOLO"
1405      0954      EPNO = "PI-EA1-13"
1406      0955      GRAFX0 = 13
1407      0956      GOTO AMSG
1408      0957      S104:     MSSGO = "90 PSI STEAM HEADER PRESS LOW "
1409      0958      EPNO = "PI-EA1-13"
1410      0959      GRAFX0 = 13
1411      0960      GOTO AMSG
1412      0961      S105:     MSSGO = "REBOILER STEAM FLOW LOW "
1413      0962      EPNO = "PIC-EA1-1"
1414      0963      GRAFX0 = 13
1415      0964      GOTO AMSG
1416      0965      S106:     MSSGO = "REBOILER STEAM INLET PRESS LOW"
1417      0966      EPNO = "PI-EA1-1"
1418      0967      GRAFX0 = 13
1419      0968      GOTO AMSG
1420      0969      S107:     MSSGO = "REBOILER STEAM INLET PRESS HI "
1421      0970      EPNO = "PI-EA1-1"
1422      0971      GRAFX0 = 13
1423      0972      GOTO AMSG
1424      0973      S108:     MSSGO = "REBOILER STM CONDNSATE TEMP HI"
1425      0974      EPNO = "TI-EA1-5"
1426      0975      GRAFX0 = 13
1427      0976      GOTO AMSG
1428      0977      S109:     MSSGO = "E-C-1 RAW WATER INLET PRESS HI"
1429      0978      EPNO = "PI-EC1-5"
1430      0979      GRAFX0 = 16

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:04:09 08-Jan-96

Page 27

```

1431 0980      GOTO MSG
1432 0981 S110:  MSGO = "E-C-3 CONDENS URW FLOW LOW  "
1433 0982      EPNO = "FIC-EC3-1"
1434 0983      GRAFXO = 16
1435 0984      GOTO MSG
1436 0985 S111:  MSGO = "E-C-1 PC OUTLET TEMP HIGH  "
1437 0986      EPNO = "TIC-EC1-2"
1438 0987      GRAFXO = 16
1439 0988      GOTO MSG
1440 0989 S112:  MSGO = "WATER SERVICE BLD BFP-1 LEAK  "
1441 0990      EPNO = "LDS-BFP-1"
1442 0991      GRAFXO = 11
1443 0992      GOTO MSG
1444 0993 S113:  MSGO = "WATER SERVICE BLD SUMP LEVEL LO"
1445 0994      EPNO = "LSL-RWSMP"
1446 0995      GRAFXO = 11
1447 0996      GOTO MSG
1448 0997 S114:  MSGO = "WTR SRVICE BUILDING BFP-2 LEAK"
1449 0998      EPNO = "LDS-BFP-2"
1450 0999      GRAFXO = 11
1451 1000      GOTO MSG
1452 1001 S115:  MSGO = "F-RW-1 FILTER DELTA PRESS HI  "
1453 1002      EPNO = "PDI-RW-1"
1454 1003      GRAFXO = 11
1455 1004      GOTO MSG
1456 1005 S116:  MSGO = "F-RW-2 FILTER DELTA PRESS HI  "
1457 1006      EPNO = "PDI-RW-2"
1458 1007      GRAFXO = 11
1459 1008      GOTO MSG
1460 1009 S117:  MSGO = "F-RW-1/2 FLTR UPSTREAM PRES LO"
1461 1010      EPNO = "PI-RW-2"
1462 1011      GRAFXO = 11
1463      S118:  ;      MSGO = "F-RW-1/2 FLTR UPSTRM PRES HIHI"
1464      ;      EPNO = "PI-RW-2"
1465      ;      GRAFXO = 11
1466      ;      GOTO MSG
1467 1012 S119:  MSGO = "E-C-1 CONDENS URW FLOW LOW  "
1468 1013      EPNO = "FIC-EC1-1"
1469 1014      GRAFXO = 17
1470 1015      GOTO MSG
1471      S120:  ;      MSGO = "E-C-1 CONDENS URW FLOW HIGH  "
1472      ;      EPNO = "FIC-EC1-1"
1473      ;      GRAFXO = 17
1474      ;      GOTO MSG
1475 1016 S121:  MSGO = "VESSEL VENT EXHSTR CURRENT LO"
1476 1017      EPNO = "II-EXC-1"
1477 1018      GRAFXO = 20
1478 1019      GOTO MSG
1479      S122:  ;      MSGO = "VESSEL VENT EXHSTR CURRENT HI  "
1480      ;      EPNO = "II-EXC-1"
1481      ;      GRAFXO = 20
1482      ;      GOTO MSG
1483      S123:  ;      MSGO = "VESSEL VENT EXHAUSTER FLO LOLO"
1484      ;      EPNO = "FI-AS-5"
1485      ;      GRAFXO = 20

```

```

1486      ; GOTO AMSG
1487      ; S124:      MSSGO = "VESSEL VENT EXHAUSTER FLOW LOW"
1488      ; EPNO = "FI-AS-5"
1489      ; GRAFXO = 20
1490      ; GOTO AMSG
1491      ; S125:      MSSGO = "VESSEL VENT EXHAUSTER FLOW HI "
1492      ; EPNO = "FI-AS-5"
1493      ; GRAFXO = 20
1494      ; GOTO AMSG
1495      ; S126:      MSSGO = "AIR SAMPLE PUMP CURRENT LOW "
1496      ; EPNO = "FI-AS-1"
1497      ; GRAFXO = 20
1498      ; GOTO AMSG
1499      ; S127:      MSSGO = "AIR SAMPLE PUMP CURRENT HIGH "
1500      ; EPNO = "FI-AS-1"
1501      ; GRAFXO = 20
1502      ; GOTO AMSG
1503      ; S128:      MSSGO = "A-C100 AGITATOR SHUTDOWN "
1504      ; EPNO = "VYS-A-C100"
1505      ; GRAFXO = 18
1506      ; GOTO AMSG
1507      ; S129:      MSSGO = "P-C100 PUMP SHUTDOWN " ; JFB03
1508      ; EPNO = "VYS-P-C100"
1509      ; GRAFXO = 18
1510      ; GOTO AMSG
1511      ; S130:      MSSGO = "HANDVALVE 1-29 CLOSED "
1512      ; EPNO = "ZS-IDX-2"
1513      ; GRAFXO = 18
1514      ; GOTO AMSG
1515      ; S131:      MSSGO = "PC3 FILTER INLET PRESSURE HIGH"
1516      ; EPNO = "PSN-ILF-1"
1517      ; GRAFXO = 18
1518      ; GOTO AMSG
1519      ; S132:      MSSGO = "HNDVLV 1-16 PC TO TK-C100 OPEN"
1520      ; EPNO = "ZS-RC3-4"
1521      ; GRAFXO = 18
1522      ; GOTO AMSG
1523      ; S133:      MSSGO = "HNDVLV 1-17 PC TO TK-102AW OPN"
1524      ; EPNO = "ZS-RC3-5"
1525      ; GRAFXO = 18
1526      ; GOTO AMSG
1527      ; S134:      MSSGO = "PROCESS CONDENSES AGIT AMPS HI "
1528      ; EPNO = "FI-A-C100"
1529      ; GRAFXO = 18
1530      ; GOTO AMSG
1531      ; S135:      MSSGO = "TK-C-100 UPPER INTERFACE LOW "
1532      ; EPNO = "FI-C100-1"
1533      ; GRAFXO = 18
1534      ; GOTO AMSG
1535      ; S136:      MSSGO = "PROCESS CONDENSES PUMP AMPS LOW"
1536      ; EPNO = "FI-P-C100"
1537      ; GRAFXO = 18
1538      ; GOTO AMSG
1539      ; S137:      MSSGO = "NF-CA1-L/R DELTA PRESS HI "
1540      ; EPNO = "PD1-CA1-3"

```

;commented out, -mct5+


```

1541 1051      GRAFX0 = 12
1542 1052      GOTO AMSG
1543 1053 S138:  MSSGO = "STM CONDSATE BASIN-1 LEVEL LOW"
1544 1054      EPNO = "LSL-SC-1"
1545 1055      GRAFX0 = 22
1546 1056      GOTO AMSG
1547 1057 S139:  MSSGO = "STM CONDSATE BASIN-2 LEVEL LOW"
1548 1058      EPNO = "LSL-SC-2"
1549 1059      GRAFX0 = 22
1550 1060      GOTO AMSG
1551 1061 S140:  MSSGO = "STM CONDSATE BASIN-3 LEVEL LOW"
1552 1062      EPNO = "LSL-SC-3"
1553 1063      GRAFX0 = 22
1554 1064      GOTO AMSG
1555      ;      MSSGO = "207-A SUMP PUMP P1 CURRENT HI "
1556      ;      EPNO = "I1-SUMPP1"
1557      ;      GRAFX0 = 22
1558      ;      GOTO AMSG
1559      ;      MSSGO = "STEAM CONDENS8 PUMP P3 AMPS HI"
1560      ;      EPNO = "I1-SC-P3"
1561      ;      GRAFX0 = 999
1562      ;      GOTO AMSG
1563 1065 S143:  MSSGO = "PROCESS AIR CMPRESSR NO. 1 OFF"
1564 1066      EPNO = "YS-CPE1-1"
1565 1067      GRAFX0 = 21
1566 1068      GOTO AMSG
1567 1069 S144:  MSSGO = "PROCESS AIR CMPRESSR NO. 2 OFF"
1568 1070      EPNO = "YS-CPE2-1"
1569 1071      GRAFX0 = 21
1570 1072      GOTO AMSG
1571 1073 S145:  MSSGO = "INSTRUMENT AIR HEADER PRESS LO"
1572 1074      EPNO = "PI-IA-1"
1573 1075      GRAFX0 = 21
1574 1076      GOTO AMSG
1575 1077 S146:  MSSGO = "INSTRUMENT AIR HEADER PRESS HI"
1576 1078      EPNO = "PI-IA-1"
1577 1079      GRAFX0 = 21
1578 1080      GOTO AMSG
1579      ;      MSSGO = "PROCESS AIR HEADER PRESSURE LO"
1580      ;      EPNO = "PI-PA-1"
1581      ;      GRAFX0 = 21
1582      ;      GOTO AMSG
1583 1081 S148:  MSSGO = "PROCESS AIR HEADER PRESSURE HI"
1584 1082      EPNO = "PI-PA-1"
1585 1083      GRAFX0 = 21
1586 1084      GOTO AMSG
1587 1085 S149:  MSSGO = "COLD JNCTN AMBIENT SENSOR LOLO"
1588 1086      EPNO = "TI-AMBS-1"
1589 1087      GRAFX0 = 21
1590 1088      GOTO AMSG
1591 1089 S150:  MSSGO = "COLD JNCTN AMBIENT SENSOR LOW "
1592 1090      EPNO = "TI-AMBS-1"
1593 1091      GRAFX0 = 21
1594 1092      GOTO AMSG
1595 1093 S151:  MSSGO = "COLD JNCTN AMBIENT SENSOR HIGH"

```

1596	1094		EPNO = "TI-AMBS-1"
1597	1095		GRAFX0 = 21
1598	1096		GOTO AMSG
1599	1097	S152:	MSSGO = "COLD JNCTN AMBIENT SENSOR HIHI"
1600	1098		EPNO = "TI-AMBS-1"
1601	1099		GRAFX0 = 21
1602	1100		GOTO AMSG
1603	1101	S153:	MSSGO = "COLD JNCTN AMBIENT SENSOR LOLO"
1604	1102		EPNO = "TI-AMBS-2"
1605	1103		GRAFX0 = 21
1606	1104		GOTO AMSG
1607	1105	S154:	MSSGO = "COLD JNCTN AMBIENT SENSOR LOW "
1608	1106		EPNO = "TI-AMBS-2"
1609	1107		GRAFX0 = 21
1610	1108		GOTO AMSG
1611	1109	S155:	MSSGO = "COLD JNCTN AMBIENT SENSOR HIGH"
1612	1110		EPNO = "TI-AMBS-2"
1613	1111		GRAFX0 = 21
1614	1112		GOTO AMSG
1615	1113	S156:	MSSGO = "COLD JNCTN AMBIENT SENSOR HIHI"
1616	1114		EPNO = "TI-AMBS-2"
1617	1115		GRAFX0 = 21
1618	1116		GOTO AMSG
1619	1117	S157:	MSSGO = "ELUANT AGITATOR MOTOR CF-ON "
1620	1118		EPNO = "YS-A-E101"
1621	1119		GRAFX0 = 19
1622	1120		GOTO AMSG
1623	1121	S158:	MSSGO = "ELUANT PUMP MOTOR CONFIRM ON "
1624	1122		EPNO = "YS-P-E101"
1625	1123		GRAFX0 = 19
1626	1124		GOTO AMSG
1627	1125	S159:	MSSGO = "ANTIFOAM AGITATOR MOTOR CF-ON "
1628	1126		EPNO = "YS-A-E102"
1629	1127		GRAFX0 = 19
1630	1128		GOTO AMSG
1631	1129	S160:	MSSGO = "ANTIFOAM PUMP MOTOR CONFIRM ON"
1632	1130		EPNO = "YS-P-E102"
1633	1131		GRAFX0 = 19
1634	1132		GOTO AMSG
1635	1133	S161:	MSSGO = "DECON AGITATOR MOTOR CF-ON "
1636	1134		EPNO = "YS-A-104"
1637	1135		GRAFX0 = 19
1638	1136		GOTO AMSG
1639	1137	S162:	MSSGO = "DECON PUMP MOTOR CONFIRM ON "
1640	1138		EPNO = "YS-P-E104"
1641	1139		GRAFX0 = 19
1642	1140		GOTO AMSG
1643	1141	S163:	MSSGO = "TK-E-101 LIQUID TEMPERATURE HI"
1644	1142		EPNO = "TI-E101-2"
1645	1143		GRAFX0 = 19
1646	1144		GOTO AMSG
1647	1145	S164:	MSSGO = "TK-E-101 WEIGHT FACTOR LOLO "
1648	1146		EPNO = "WF1-E101"
1649	1147		GRAFX0 = 19
1650	1148		GOTO AMSG

```

1651 1149 S165:  MSSGO = "TK-E-101 WEIGHT FACTOR LO      "
1652 1150          EPNO = "WFI-E101"
1653 1151          GRAFXO = 19
1654 1152          GOTO AMSG
1655 1153 S166:  MSSGO = "TK-E-102 WEIGHT FACTOR LOLO    "
1656 1154          EPNO = "WFI-E102"
1657 1155          GRAFXO = 19
1658 1156          GOTO AMSG
1659 1157 S167:  MSSGO = "TK-E-102 WEIGHT FACTOR LO      "
1660 1158          EPNO = "WFI-E102"
1661 1159          GRAFXO = 19
1662 1160          GOTO AMSG
1663 1161 S168:  MSSGO = "TK-E-104 LIQUID TEMP HIGH     "
1664 1162          EPNO = "TI-E104-2"
1665 1163          GRAFXO = 19
1666 1164          GOTO AMSG
1667 1165 S169:  MSSGO = "TK-E-104 WEIGHT FACTOR LOLO    "
1668 1166          EPNO = "WFI-E104"
1669 1167          GRAFXO = 19
1670 1168          GOTO AMSG
1671 1169 S170:  MSSGO = "TK-E-104 WEIGHT FACTOR LO      "
1672 1170          EPNO = "WFI-E104"
1673 1171          GRAFXO = 19
1674 1172          GOTO AMSG
1675          S171:  ;      MSSGO = "LERF NORTH GATE OPEN      " ; JFB02
1676          ;      EPNO = "ZS-105PS1"
1677          ;      GRAFXO = 24
1678          ;      GOTO AMSG
1679          S172:  ;      MSSGO = "LERF SOUTH GATE OPEN     " ; JFB02
1680          ;      EPNO = "ZS-105PS2"
1681          ;      GRAFXO = 24
1682          ;      GOTO AMSG
1683 1173 S173:  MSSGO = "LERF LEACHATE PUMP 42 ON        " ; +mct4-
1684 1174          EPNO = "YS-P-42"
1685 1175          GRAFXO = 24
1686 1176          GOTO AMSG
1687 1177 S174:  MSSGO = "LERF LEACHATE PUMP 43 ON        " ; +mct4-
1688 1178          EPNO = "YS-P-43"
1689 1179          GRAFXO = 24
1690 1180          GOTO AMSG
1691 1181 S175:  MSSGO = "LERF LEACHATE PUMP 44 ON        " ; +mct4-
1692 1182          EPNO = "YS-P-44"
1693 1183          GRAFXO = 24
1694 1184          GOTO AMSG
1695 1185 S176:  MSSGO = "P-C106 PUMP SHUTDOWN            " ; JFB01
1696 1186          EPNO = "YS-P-C106"
1697 1187          GRAFXO = 18
1698 1188          GOTO AMSG
1699 1189 S177:  MSSGO = "HV-CA1-10 TRAVEL TO RAW WATER  " ; JFB01
1700 1190          EPNO = "HV-CA1-10"
1701 1191          GRAFXO = 12
1702 1192          GOTO AMSG
1703          ;
1704          ;      END STRINGS COLLECTION FOR ALARM SUBROUTINE
1705          ;

```

```

1706 :
1707 :
1708 :
1709 :
1710 :
1711 :
1712 :
1713 :
1714 :
1715 :
1716 :
1717 :
1718 :
1719 :
1720 :
1721 :
1722 :
1723 :
1724 :
1725 :
1726 :
1727 :
1728 :
1729 :
1730 :
1731 :
1732 :
1733 :
1734 :
1735 :
1736 :
1737 :
1738 :
1739 :
1740 :
1741 :
1742 :
1743 :
1744 :
1745 :
1746 :
1747 :
1748 :
1749 :
1750 :
1751 :
1752 :
1753 :
1754 :
1755 :
1756 :
1757 :
1758 :
1759 :
1760 :
1761 :
1762 :
1763 :
1764 :
1765 :
1766 :
1767 :
1768 :
1769 :
1770 :
1771 :
1772 :
1773 :
1774 :
1775 :
1776 :
1777 :
1778 :
1779 :
1780 :
1781 :
1782 :
1783 :
1784 :
1785 :
1786 :
1787 :
1788 :
1789 :
1790 :
1791 :
1792 :
1793 :
1794 :
1795 :
1796 :
1797 :
1798 :
1799 :
1800 :
1801 :
1802 :
1803 :
1804 :
1805 :
1806 :
1807 :
1808 :
1809 :
1810 :
1811 :
1812 :
1813 :
1814 :
1815 :
1816 :
1817 :
1818 :
1819 :
1820 :
1821 :
1822 :
1823 :
1824 :
1825 :
1826 :
1827 :
1828 :
1829 :
1830 :
1831 :
1832 :
1833 :
1834 :
1835 :
1836 :
1837 :
1838 :
1839 :
1840 :
1841 :
1842 :
1843 :
1844 :
1845 :
1846 :
1847 :
1848 :
1849 :
1850 :
1851 :
1852 :
1853 :
1854 :
1855 :
1856 :
1857 :
1858 :
1859 :
1860 :
1861 :
1862 :
1863 :
1864 :
1865 :
1866 :
1867 :
1868 :
1869 :
1870 :
1871 :
1872 :
1873 :
1874 :
1875 :
1876 :
1877 :
1878 :
1879 :
1880 :
1881 :
1882 :
1883 :
1884 :
1885 :
1886 :
1887 :
1888 :
1889 :
1890 :
1891 :
1892 :
1893 :
1894 :
1895 :
1896 :
1897 :
1898 :
1899 :
1900 :
1901 :
1902 :
1903 :
1904 :
1905 :
1906 :
1907 :
1908 :
1909 :
1910 :
1911 :
1912 :
1913 :
1914 :
1915 :
1916 :
1917 :
1918 :
1919 :
1920 :
1921 :
1922 :
1923 :
1924 :
1925 :
1926 :
1927 :
1928 :
1929 :
1930 :
1931 :
1932 :
1933 :
1934 :
1935 :
1936 :
1937 :
1938 :
1939 :
1940 :
1941 :
1942 :
1943 :
1944 :
1945 :
1946 :
1947 :
1948 :
1949 :
1950 :
1951 :
1952 :
1953 :
1954 :
1955 :
1956 :
1957 :
1958 :
1959 :
1960 :
1961 :
1962 :
1963 :
1964 :
1965 :
1966 :
1967 :
1968 :
1969 :
1970 :
1971 :
1972 :
1973 :
1974 :
1975 :
1976 :
1977 :
1978 :
1979 :
1980 :
1981 :
1982 :
1983 :
1984 :
1985 :
1986 :
1987 :
1988 :
1989 :
1990 :
1991 :
1992 :
1993 :
1994 :
1995 :
1996 :
1997 :
1998 :
1999 :
2000 :

```

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:04:09 08-Jan-96

Page 33

```
1761      ;  
1762 1224 TERM: FABORT = 0  
1763 1225 PRINT 0," --- WHITE1 TERMINATED"  
1764      END
```

```
Total Errors Detected      0      Total Warnings      0  
Sequence Program Size = 26128 Bytes, 1633 Pages  
305 Forward Reference(s)
```

ANBATCH

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:32:44 24-Feb-95

Page 1

```

1      ; ANBATC H . S E Q                                VERSION 2
2      ;
3      ; Function: Sequence program for Lamp function,
4      ;           message display on graphics, and
5      ;           printer messages.
6      ;
7      ; Written By : R. N. Kersey
8      ; Date       : 05-31-92
9      ; Facility   : 242-A Evaporator Bldg.
10     ; Machine    : VAX 4104-A
11     ; Op. System : VAX/VMS 5.5-2AH
12     ; D/3 Version : 6.3-2 BASELINE
13     ; Compiler   : SIC
14     ; Build command : @[320,20]ANBATC H
15     ;
16     ; UNIT = 1, SKDKEY = 0, 1, 2, 3
17     ;
18     ; Assoc. UPF: [300,1]_TNKFRM.____ TNKFRM.LST
19     ; Assoc. RPF: [300,1]_FARM.____ FARM.LST
20     ; SDB:        [300,1]PCMTASDB.____ PCMTASDB.CMD
21     ;
22     ; POINTER SDB VAR. FUNCTION
23     ; -----
24     ; PSTAT INTn Current EPN alarm state is an integer tied to
25     ;           graphic flashing string of value:
26     ;           (n=0-100) 0 = previously not alarmed
27     ;           .         1 = previously alarmed, not ack
28     ;           .         2 = previously alarmed, is ack
29     ;           .         3 = alarm is inhibited
30     ;           .         4 = input is off scan
31     ;           .         5 = input is bad
32     ; PDIN 'EPN' ---
33     ; PTYPE IALMn alarm type 0 = none, 1 = white, 2 = yellow,
34     ;           3 = yellow and yellow, 4 = red,
35     ;           6 = yellow and red
36     ; PREV FXn Reverse acting (not fail safe) flag (1=set)
37     ; PSKID SKIDn Tracks which graphic has an alarm
38     ; PPASS FPASn If set (1), turn lamp off when alarm is ACKnowledged
39     ; PSN FSMn Flag for the graphic to ACKnowledge
40     ; --- FHOLD Flag to update the lamps
41     ; --- FWING Flag for DOGIA.C micro select
42     ; --- ACK Flag for C program to ACK system wide
43     ;
44     ; MODIFICATION LOG:
45     ;
46     ; 05-APR-93 Modified MCC OFF white alarm on all 7 M. Teats
47     ; YS- signals to pump OFF white alarm because
48     ; the YS DINs do not indicate MCC status - they
49     ; indicate the pump status.
50     ;
51     ; -mct1+ 13-MAY-93 Remove pump shutdown DEV white alarms, which M. Teats
52     ; are based on the analog signals, because they
53     ; are not needed due to the 4/5/93 mods. See
54     ; CCN# 93MAY13-01.
55     ;

```

```

56      ;      15-JUN-93      Added comments and baselined for      M. Teats
57      ;      D/3 version 3.3.  CCW# 93JUN01-01.
58      ;
59      ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
60      ;      system. Rebaselined document. Removed
61      ;      obsolete Select() statements.
62      ;
63      ;      END MOD LOG
64      ;
65      UNIT      ANFARM
66      ;
67      PUMP == FLSPIT
68      AUTO == 'FQI-AN-1':CB_FLG:1      ; For FQI-RESET auto mode check
69      CALOUT == 'FQI-AN-1':CB_OTVL:1      ; For FQI-RESET calc block outvalue
70      ;
71      0000      GETUNIT TRIP      ; Which sequence unit is program running in
72      0001      IF TRIP != 1 THEN      ; Abort if running in wrong seq. unit
73      ;
74      0002      BEGIN
75      ;
76      0003      FSTART = 1
77      0004      ILOOP = 54
78      0005      GOSUB CYCL
79      0006      FSTART = 1
80      0007      PRINT ADI," >>>>> ATTEMPT TO RUN ANBATCH IN WRONG UNIT"
81      ;
82      ;      END
83      ;      IF ISELECT() THEN GOTO AGAIN
84      ;
85      ;      BEGIN INITIALIZING:
86      0008      NOP
87      0009      INIT:  LET FABORT = 0      ; If FABORT = 1 then abort program
88      0010      LET FSTART = 0      ; Program start flag
89      0011      LET FLDALM = 0
90      0012      LET FSL7 = 0      ; For FQI-RESET manual mode flag
91      0013      LAMP 0,3,0      ; Turn off the lamps
92      0014      LAMP 0,2,0
93      0015      LAMP 0,1,0
94      0016      LAMP 0,0,0
95      0017      ASSIGN PDEV = ANFARM      ; First of 5 alarm ack devices
96      0018      FOR ILOOP = 1 TO 5
97      0019      PDEV:=DEV_TMPO = 1      ; Set all 5 devices to ack state
98      0020      IF AND(PDEV:=DEV_ALMCT,16) THEN      ; If alarm device is inha, remove the inha
99      0021      PDEV:=DEV_ALMCT=XOR(PDEV:=DEV_ALMCT,16)
100     0022      IF AND(PDEV:=DEV_ALMCT,32) THEN      ; If alarm device is ack, clear it
101     0023      PDEV:=DEV_ALMCT = XOR(PDEV:=DEV_ALMCT,32)
102     0024      STEP PDEV BY 1
103     0025      NEXT ILOOP
104     ;
105     ;      First, set all flags and integers:
106     0026      ASSIGN PDIN = 'YS-AN-101'      ; Assign pointer to first DIN
107     0027      ASSIGN PSTAT = INTO
108     0028      ASSIGN PTYPE = IALMO
109     0029      ASSIGN PREV = FXO
110     0030      ASSIGN PSKID = SKIDO

```



```

111 0031      ASSIGN PPASS = FPASO
112      ;
113 0032      LET FARM = 0                      ; Set up values for different graphics
114 0033      SERPIT = 1                        ; SKID of each EPN indicates graphic
115 0034      VLVPIT = 2
116 0035      HVAC = 3
117 0036      BLD271 = 4
118      ;
119 0037      LEAK0 = 3                        ; FARM
120 0038      LEAK1 = 64                       ; SERVICE PIT ( NOT USED )
121 0039      LEAK2 = 1                       ; VALVE PIT
122 0040      LEAK3 = 2                       ; HVAC
123 0041      LEAK4 = 0                       ; 271
124      ;
125 0042      FOR ILOOP = 0 TO 44              ; For DINs
126 0043      LET PPASS = 0
127 0044      LET PREV = 0
128 0045      LET PTYPE = 2
129      ;
130 0046      BLET PSTAT = 0
131      IF AND(PDIN:DIN_DFLG,3) THEN          ; Check for alarms (must be red or yellow)
132      BEGIN
133      LET PTYPE = 2                          ; Default to yellow alarm
134      IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1 ; Reverse acting
135      IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4 ; Set to red critical alarm
136      END
137 0053      IF ILOOP < 21 THEN LET PSKID = FARM ; These are set by IPN order
138 0055      IF (ILOOP > 20 & ILOOP < 29) THEN LET PSKID = VLVPIT
139 0057      IF (ILOOP > 28 & ILOOP < 37) THEN LET PSKID = HVAC
140 0059      IF ILOOP > 36 THEN LET PSKID = BLD271
141 0060      IF AND(PDIN:DIN_ALMCT,32) THEN
142 0061      PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
143      SKP: STEP PSTAT BY 1
144      STEP PDIN BY 1
145      STEP PTYPE BY 1
146      STEP PREV BY 1
147      STEP PSKID BY 1
148      STEP PPASS BY 1
149      NEXT ILOOP
150      ;
151      ;-mct1- Abandoned code for white alarms tied to pump devices:
152      ;
153      ; FOR ILOOP = 45 TO 51
154      ; BLET PSTAT = 0
155      ; BLET PTYPE = 1
156      ; BLET PREV = 0
157      ; BLET PSKID = 0
158      ; BLET PPASS = 1
159      ; STEP PSTAT BY 1
160      ; STEP PTYPE BY 1
161      ; STEP PREV BY 1
162      ; STEP PSKID BY 1
163      ; STEP PPASS BY 1
164      ; NEXT ILOOP
165      ;-mct1-
166      ;

```

```

166      ; Set the white alarms:
167      ;
168      0069      LET IALM0 = 1
169      0070      LET IALM3 = 1
170      0071      LET IALM6 = 1
171      0072      LET IALM9 = 1
172      0073      LET IALM12 = 1
173      0074      LET IALM15 = 1
174      0075      LET IALM18 = 1
175      0076      LET IALM42 = 1
176      ;
177      ; Pumps alarm on 0, No lamp when Ack:
178      ;
179      0077      FPAS0 = 1
180      0078      FPAS3 = 1
181      0079      FPAS6 = 1
182      0080      FPAS9 = 1
183      0081      FPAS12 = 1
184      0082      FPAS15 = 1
185      0083      FPAS18 = 1
186      ;
187      ; END INITIALIZING
188      ;
189      ; SEND PROGRAM START MESSAGE
190      ;
191      0084      ILOOP = 53
192      0085      FSTART = 1
193      0086      TIMERCLR STMR2
194      0087      TIMERON STMR2
195      0088      GOSUB CYCL
196      0089      ON ERROR GOSUB ERRSUB
197      ;
198      ; BEGIN PROGRAM LOOP:
199      ;
200      0090      AGAIN: MARK STEP "AGAIN",0,0
201      ; IF ISELECT() THEN FNING = 0
202      ; ELSE FNING = 1
203      ; WAITUNTIL SELECT() ! FABORT
204      ; IF (FABORT) THEN GOTO TERM
205      ; IF FNING = 0 THEN FHOLD = 1
206      ; IF STMR2 > 30 THEN FHOLD = 1
207      0093      WAIT 1
208      ;
209      0094      DEV0 = 0
210      0095      DEV1 = 0
211      0096      DEV2 = 0
212      0097      DEV3 = 0
213      0098      DEV4 = 0
214      ;
215      ; LOOP CHECKS IF ANY OF 5 ALARM DEVICES HAS BEEN ACKNOWLEDGED.
216      ; IF ALARM DEVICE ACKNOWLEDGED AND DEV_TMO = 2 THEN ACKNOWLEDGE
217      ; EVERY SIGNAL ON THAT GRAPHIC
218      ;
219      ;
220      ;

```

DEVICE	FLAG	OFFSET
----	-----	-----

```

221      ;          ANFARM          FLEAK0          447
222      ;          AN-SERPIT        FLEAK1          448      (NOT USED)
223      ;          AN-VLVPIT        FLEAK2          449
224      ;          AN-HVAC          FLEAK3          450
225      ;          AN-271           FLEAK4          451
226      ;
227 0099      ASSIGN PDEV = ANFARM
228 0100      ASSIGN PSN = FSN0
229 0101      ASSIGN PACK = FLEAK0
230 0102      FOR ILOOP = 0 TO 4
231 0103      LET PSN = 0
232 0104      IF (PDEV:DEV_TMP0 = 2) & AND(PDEV:DEV_ALMCT,32) THEN
233          BEGIN
234 0105          PDEV:DEV_TMP0 = 1          ; Put device in ack state
235 0106          LET PSN = 1              ; Set flag for the graphic to ack
236 0107          LET FLGACK = 1          ; WATCH.C spawns ACKFLG.C, which resets flag
237 0108          LET ACK = 1              ; Set flag for C program to ACK system wide
238 0109          LET PACK = 1
239 0110          ASSIGN PMSG0 = LEAK0
240 0111          STEP PMSG0 BY ILOOP
241 0113          IF ILOOP != 1 THEN LAMP 0,PMSG0,1      ; Flash Lamp
242          END
243 0114          STEP PSN BY 1
244 0115          STEP PACK BY 1
245 0117          IF ILOOP < 6 THEN STEP PDEV BY 1 ; Only 5 devices
246 0118          WAIT 2
247 0119          NEXT ILOOP
248      ;
249      ;          FIELD ALARM SILENCE FEATURE REMOVED FROM ANFARM
250      ;          UNTIL 271-AN PANEL BOARD WIRING RESOLVED (1/24/92 RNK):
251      ;
252      ;          IF (ACK) THEN
253      ;              BEGIN
254      ;                  BTIMERCLR STMR1
255      ;                  BTIMERON STMR1
256      ;                  BLET FLDALM = 1
257      ;                  END
258      ;          IF (FLDALM) THEN IF STMR1 >= 30 THEN      ; If field alarm then ack in 30 sec
259      ;              BEGIN
260      ;                  BTIMEROFF STMR1
261      ;                  BTIMERCLR STMR1
262      ;                  BLET FLDALM = 0
263      ;                  OPEN 'HS-AN-SIL'                  ; Silence alarm from PCM
264      ;                  END
265      ;
266      ;          ; If field alarm ack then set flag and start timer:
267      ;
268 0121      IF (RESET) THEN IF LMPVAR < 2 THEN
269          BEGIN
270 0122          LET FLDALM = 1
271 0123          TIMERON STMR1
272 0124          LET RESET = 0
273          END
274 0126          IF LMPVAR = 2 THEN LET RESET = 1
275      ;

```

```

276 ; BEGIN DINs:
277 ;
278 0127 ASSIGN PDIN = 'YS-AN-101' ; First DIN in ANFARM
279 0128 ASSIGN PTYPE = IALMO
280 0129 ASSIGN PREV = FXO
281 0130 ASSIGN PSTAT = INTO
282 0131 ASSIGN PSKID = SKID0
283 0132 ASSIGN PPASS = FPASO
284 0133 CURENT = 0 ; Used to set lamp status
285 0134 NXT = 0 ; Temporary INT
286 0135 BITVAL = 0
287 ;
288 0136 LOP: FOR ILOOP = 1 TO 45
289 0137 IF PTYPE THEN ; If some alarm
290 BEGIN
291 0139 IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3 ;be1 ; If EPN is inha
292 0141 IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5 ; If EPN is ibad
293 0143 IF AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4 ; If EPN is dis
294 0146 IF ! PSTAT THEN IF PREV = PDIN:DIN_VAL THEN GOSUB ALM ; If not previously alarmed
295 0147 IF PSTAT THEN
296 BEGIN ;be2
297 0148 IF PSTAT < 3 THEN ; If not inha
298 BEGIN ;be3
299 0149 IF PSTAT = 1 THEN ;be4
300 BEGIN ;be5
301 0150 IF AND(PDIN:DIN_ALMCT,32) THEN ; If alarm is ack
302 BEGIN ;be6
303 0151 LET PSTAT = 2 ; If a graphic has been ack
304 0152 FHOLD = 1 ; go find which graphic
305 END ;en5
306 0153 IF ACK THEN ; If a graphic has been ack
307 BEGIN ;be6 - go find which graphic
308 0154 CRIT
309 0155 ASSIGN PSN = FSNO ; Routine to find graphic
310 0156 STEP PSN BY PSKID
311 0157 IF PSN THEN ; If this is the graphic
312 BEGIN ;be7 - then ack it
313 0158 LET PSTAT = 2
314 0160 IF PTYPE = 1 THEN PDIN:DIN_ALMCT = OR(PDIN:DIN_ALMCT,32)
315 0161 FPASS = 1
316 0162 GOSUB ACK ; Print signal was ack
317 END ;en7
318 END ;en6
319 END ;en4
320 0164 IF PREV != PDIN:DIN_VAL THEN GOSUB CLR ; If alarm has cleared
321 0165 ASSIGN PMSGO = DEVO ; Find active alarms & graphic:
322 0166 STEP PMSGO BY PSKID
323 0167 IF PPASS THEN
324 BEGIN
325 0169 IF PSTAT = 1 THEN PMSGO = OR(PMSGO,PSTAT)
326 END
327 0170 ELSE PMSGO = OR(PMSGO,PSTAT)
328 END ;en3
329 0171 ELSE IF AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8)THEN
330 BEGIN

```

PAGE 562

```

386      ;          BEGIN
387      ;          BLET PSTAT = 1
388      ;          IF AND(PEPN:DEV_ALMCT,32)THEN PEPN:DEV_ALMCT = XOR(PEPN:DEV_ALMCT,32)
389      ;          ASSIGN PDEV = ANFARM
390      ;          STEP PDEV BY PSKID
391      ;          PDEV:DEV_TMO = 2
392      ;          GOSUB CYCL
393      ;          END
394      ;          END
395      ;          ELSE IF !AND(PEPN:DEV_STAT,16)THEN
396      ;          BEGIN
397      ;          BLET PSTAT = 0
398      ;          IF AND(PEPN:DEV_ALMCT,32)THEN PEPN:DEV_ALMCT = XOR(PEPN:DEV_ALMCT,32)
399      ;          Fhold = 1
400      ;          END
401      ;          IF PSTAT THEN
402      ;          BEGIN
403      ;          ASSIGN PMSG0 = DEV6
404      ;          STEP PMSG0 BY PSKID
405      ;          IF PPASS THEN
406      ;          BEGIN
407      ;          IF PSTAT = 1 THEN PMSG0 = OR(PMSG0,PSTAT)
408      ;          END
409      ;          ELSE PMSG0 = OR(PMSG0,PSTAT)
410      ;          END
411      ;          END
412      ;          ELSE IF !AND(PEPN:DEV_ALMCT,16)&!AND(PEPN:DEV_ALMCT,4)&AND(PEPN:DEV_FLG,8)THEN PSTAT = 0
413      ;          END
414      ;          STEP PEPN BY 1
415      ;          STEP PTYPE BY 1
416      ;          STEP PREV BY 1
417      ;          STEP PSTAT BY 1
418      ;          STEP PSKID BY 1
419      ;          STEP PPASS BY 1
420      ;          NEXT ILOOP
421      ;          ;mct1-
422      ;          ;
423      0189  S2ND:  ACK = 0
424      0190      SORT = 0
425      ;
426      0191      DEV12 = OR(DEV0,DEV6)
427      0192      DEV13 = OR(DEV1,DEV7)
428      0193      DEV14 = OR(DEV2,DEV8)
429      0194      DEV15 = OR(DEV3,DEV9)
430      0195      DEV16 = OR(DEV4,DEV10)
431      0196      DEV17 = OR(DEV5,DEV11)
432      ;
433      0197      ASSIGN PMSG0 = DEV12
434      0198      ASSIGN PMSG1 = FARM
435      0199      FOR LLOOP = 1 TO 5
436      0200      IF PMSG0 THEN
437      BEGIN
438      IF AND(PMSG0,1) THEN PMSG1 = 2
439      ELSE PMSG1 = 1
440      END

```

```

441 0204      ELSE PMSG1 = 0
442 0205      STEP PMSG0 BY 1
443 0206      STEP PMSG1 BY 1
444 0207      NEXT LLOOP
445
446 0208      ;
447      IF (FHOLD = 1) THEN
448      BEGIN
449      LAMP 0,3,FARM
450 0210      LAMP 0,2,HVAC
451 0211      LAMP 0,1,VLVPIT
452 0212      LAMP 0,0,BLD271
453 0213      FHOLD = 0
454 0214      TIMERCLR STM2
455 0215      TIMERON STM2
456      END
457 0216      MARK STEP "BOTTOM",0,0
458 0217      NOCRIT
459
460      ;
461      ;
462      ;
463      ;
464      ;
465 0219      IF !AND(AUTO,64) THEN FSL7 = 1 ; If manual, set manual flag
466 0220      IF AND(AUTO,64) & AND(FSL7,1) THEN ; For manual-auto transition
467      BEGIN
468 0221      FSL7 = 0
469 0222      CALOUT = 0.0
470      END
471      ;
472      ;
473      ;
474 0223      GOTO AGAIN
475      ;
476      ;
477      ;
478      ;
479      ;
480 0224      CLR: LET PSTAT = 0
481 0226      IF AND(PDIN:DIN_ALMCT,32)THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
482 0227      FHOLD = 1
483 0228      RETURN
484      ;
485 0229      ALM: LET PSTAT = 1 ; Print message, flash lamp, horn on, scroll message at bottom of graphic
486 0230      CRIT ; Critical
487 0232      IF AND(PTYPE,3)THEN LET FALMON = 1 ; Audible alarm on
488 0233      IF AND(PDIN:DIN_ALMCT,32)THEN ; Clear the scan block
489 0234      PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
490 0235      ASSIGN PMSG0 = LEAK0
491 0236      STEP PMSG0 BY PSKID
492 0237      LAMP 0,PMSG0,2 ; Lamp flash
493 0238      ASSIGN PDEV = ANFARM ; Find DEV and turn it on
494 0239      STEP PDEV BY PSKID
495 0240      PDEV:DEV_TMPO = 2

```

```

496      ;
497      ;       Scrolling alarm summary region in graphic (message, time, date, alarm type):
498      ;
499 0241  CYCL:  LET LSG4 = LSG3                      ; LSG = message
500 0242      LET LSG3 = LSG2
501 0243      LET LSG2 = LSG1
502 0244      LET LSG1 = LSG0
503 0245      LET COLOR4 = COLOR3                    ; COLOR = type of alarm
504 0246      LET COLOR3 = COLOR2
505 0247      LET COLOR2 = COLOR1
506 0248      LET COLOR1 = COLOR0
507 0249      LET EPNS4 = EPNS3                      ; EPN
508 0250      LET EPNS3 = EPNS2
509 0251      LET EPNS2 = EPNS1
510 0252      LET EPNS1 = STRNG1
511 0253      LET GRAFX4 = GRAFX3                    ; GRAFX = graphic number
512 0254      LET GRAFX3 = GRAFX2
513 0255      LET GRAFX2 = GRAFX1
514 0256      LET GRAFX1 = GRAFX0
515 0257      LET SDATE4 = SDATE3                    ; SDATE = time and date
516 0258      LET SDATE3 = SDATE2
517 0259      LET SDATE2 = SDATE1
518 0260      LET SDATE1 = SDATE0
519      ;
520      ;       ALARM TYPER STRINGS:
521      ;
522 0261  ACKK:  FHOLD = 1
523 0262      ON ILOOP GOTO S1,S2,S3,S4,S5,S6,S7,S8,S9,S10
524 0263      ON ILOOP - 10 GOTO S11,S12,S13,S14,S15,S16,S17,S18,S19,S20
525 0264      ON ILOOP - 20 GOTO S21,S22,S23,S24,S25,S26,S27,S28,S29,S30
526 0265      ON ILOOP - 30 GOTO S31,S32,S33,S34,S35,S36,S37,S38,S39,S40
527 0266      ON ILOOP - 40 GOTO S41,S42,S43,S44,S45,S46,S47,S48,S49,S50
528 0267      ON ILOOP - 50 GOTO S51,S52,S53,S54,S55
529 0268      GOTO AMSG
530 0269  S1:   LET STRNG0 = "PUMP AN-101 OFF      "
531 0270      LET STRNG1 = "YS-AN-101"
532 0271      GOTO AMSG
533 0272  S2:   LET STRNG0 = "AN-101 ANN RD HI/FAIL "
534 0273      LET STRNG1 = "RSH-AN101"
535 0274      GOTO AMSG
536 0275  S3:   LET STRNG0 = "AN-01C RAD HI/FAIL    "
537 0276      LET STRNG1 = "RSH-AN01C"
538 0277      GOTO AMSG
539 0278  S4:   LET STRNG0 = "PUMP AN-102 OFF      "
540 0279      LET STRNG1 = "YS-AN-102"
541 0280      GOTO AMSG
542 0281  S5:   LET STRNG0 = "AN-102 ANN RD HI/FAIL "
543 0282      LET STRNG1 = "RSH-AN102"
544 0283      GOTO AMSG
545 0284  S6:   LET STRNG0 = "AN-02C RADN HI/FAIL   "
546 0285      LET STRNG1 = "RSH-AN02C"
547 0286      GOTO AMSG
548 0287  S7:   LET STRNG0 = "PUMP AN-103 OFF      "
549 0288      LET STRNG1 = "YS-AN-103"
550 0289      GOTO AMSG

```



```

551 0290 S8: LET STRNG0 = "AN-103 ANN RD HI/FAIL "
552 0291 LET STRNG1 = "RSH-AN103"
553 0292 GOTO AMSG
554 0293 S9: LET STRNG0 = "AN-03C RADN HI/FAIL "
555 0294 LET STRNG1 = "RSH-AN03C"
556 0295 GOTO AMSG
557 0296 S10: LET STRNG0 = "PUMP AN-104 OFF      "
558 0297 LET STRNG1 = "YS-AN-104"
559 0298 GOTO AMSG
560 0299 S11: LET STRNG0 = "AN-104 ANN RD HI/FAIL "
561 0300 LET STRNG1 = "RSH-AN104"
562 0301 GOTO AMSG
563 0302 S12: LET STRNG0 = "AN-04C RADN HI/FAIL "
564 0303 LET STRNG1 = "RSH-AN04C"
565 0304 GOTO AMSG
566 0305 S13: LET STRNG0 = "PUMP AN-105 OFF      "
567 0306 LET STRNG1 = "YS-AN105"
568 0307 GOTO AMSG
569 0308 S14: LET STRNG0 = "AN-105 ANN RD HI/FAIL "
570 0309 LET STRNG1 = "RSH-AN105"
571 0310 GOTO AMSG
572 0311 S15: LET STRNG0 = "AN-05C RADN HI/FAIL "
573 0312 LET STRNG1 = "RSH-AN05C"
574 0313 GOTO AMSG
575 0314 S16: LET STRNG0 = "PUMP AN-106 OFF      "
576 0315 LET STRNG1 = "YS-AN-106"
577 0316 GOTO AMSG
578 0317 S17: LET STRNG0 = "AN-106 ANN RD HI/FAIL "
579 0318 LET STRNG1 = "RSH-AN106"
580 0319 GOTO AMSG
581 0320 S18: LET STRNG0 = "AN-06C RADN HI/FAIL "
582 0321 LET STRNG1 = "RSH-AN06C"
583 0322 GOTO AMSG
584 0323 S19: LET STRNG0 = "PUMP AN-107 OFF      "
585 0324 LET STRNG1 = "YS-AN-107"
586 0325 GOTO AMSG
587 0326 S20: LET STRNG0 = "AN-107 ANN RD HI/FAIL "
588 0327 LET STRNG1 = "RSH-AN107"
589 0328 GOTO AMSG
590 0329 S21: LET STRNG0 = "AN-07C RADN HI/FAIL "
591 0330 LET STRNG1 = "RSH-AN07C"
592 0331 GOTO AMSG
593 0332 S22: LET STRNG0 = "AN SERVICE PIT LEAK  "
594 0333 LET STRNG1 = "LDS-SP-AN"
595 0334 GOTO AMSG
596 0335 S23: LET STRNG0 = "AN SERPIT RAD HI/FAIL "
597 0336 LET STRNG1 = "RSH-SP-AN"
598 0337 GOTO AMSG
599 0338 S24: LET STRNG0 = "AN PROCESS PITS LEAK  "
600 0339 LET STRNG1 = "LDS-AN-PP"
601 0340 GOTO AMSG
602 0341 S25: LET STRNG0 = "AN SL-160/SL-168 LEAK "
603 0342 LET STRNG1 = "LDS-ANSL1"
604 0343 GOTO AMSG
605 0344 S26: LET STRNG0 = "AN SN-260/SN-268 LEAK "

```

606	0345	LET STRNG1 = "LDS-ANSN2"
607	0346	GOTO AMSG
608	0347	S27: LET STRNG0 = "AN CLEANOUT BOX LEAK "
609	0348	LET STRNG1 = "LDS-COBAN"
610	0349	GOTO AMSG
611	0350	S28: LET STRNG0 = "AN FLUSHLINE PRESS HI "
612	0351	LET STRNG1 = "PSH-ANSN1"
613	0352	GOTO AMSG
614	0353	S29: LET STRNG0 = "AN DR-368 ENCASE LEAK "
615	0354	LET STRNG1 = "LDS-DR368"
616	0355	GOTO AMSG
617	0356	S30: LET STRNG0 = "AN TANK EXH RADN HI "
618	0357	LET STRNG1 = "RSH-AN-K1"
619	0358	GOTO AMSG
620	0359	S31: LET STRNG0 = "AN K1 STKMOM SMPL PMP "
621	0360	LET STRNG1 = "PSH-ANK11"
622	0361	GOTO AMSG
623	0362	S32: LET STRNG0 = "AN K1 REC SMPL PUMP "
624	0363	LET STRNG1 = "PSH-ANK12"
625	0364	GOTO AMSG
626	0365	S33: LET STRNG0 = "AN TNK EX RD MON FAIL "
627	0366	LET STRNG1 = "RXS-AN-K1"
628	0367	GOTO AMSG
629	0368	S34: LET STRNG0 = "AN ANN EXH RADN HIGH "
630	0369	LET STRNG1 = "RSH-AN-K2"
631	0370	GOTO AMSG
632	0371	S35: LET STRNG0 = "AN K2 STK MN SMPL PMP "
633	0372	LET STRNG1 = "PSH-ANK21"
634	0373	GOTO AMSG
635	0374	S36: LET STRNG0 = "AN K2 REC SMPL PUMP "
636	0375	LET STRNG1 = "PSH-ANK22"
637	0376	GOTO AMSG
638	0377	S37: LET STRNG0 = "AN ANN EX RD MON FAIL "
639	0378	LET STRNG1 = "RXS-AN-K2"
640	0379	GOTO AMSG
641	0380	S38: LET STRNG0 = "AN-271 BLD POWER LOSS "
642	0381	LET STRNG1 = "JXS-AN271"
643	0382	GOTO AMSG
644	0383	S39: LET STRNG0 = "AN-271 RADN HI/FAIL "
645	0384	LET STRNG1 = "RSHAN2711"
646	0385	GOTO AMSG
647	0386	S40: LET STRNG0 = "AN-271 RADN HI/FAIL "
648	0387	LET STRNG1 = "RSHAN2712"
649	0388	GOTO AMSG
650	0389	S41: LET STRNG0 = "AN-241 RADN HI/FAIL "
651	0390	LET STRNG1 = "RSH-AN-12"
652	0391	GOTO AMSG
653	0392	S42: LET STRNG0 = "AN FARM GENERAL ALARM "
654	0393	LET STRNG1 = "QS-AN-271"
655	0394	GOTO AMSG
656	0395	S43: LET STRNG0 = "AN HEAT TRACE OVERTMP "
657	0396	LET STRNG1 = "TXS-AN-2"
658	0397	GOTO AMSG
659	0398	S44: LET STRNG0 = "AN AREA RADN BYPASS "
660	0399	LET STRNG1 = "YS-AN-ARM"

```

661 0400      GOTO AMSG
662 0401      S45:  LET STRNG0 = "AN MASTER SHUTDOWN  "
663 0402      LET STRNG1 = "YS-AN-MSD"
664 0403      GOTO AMSG
665
666      ;
667      ;--mct1- Abandoned code for white alarms tied to pump devices:
668
669      S46:    ;      BLET STRNG0 = "PUMP AN-101 SHUTDOWN  "
670      ;      BLET STRNG1 = "P-AN-101"
671      ;      GOTO AMSG
672      S47:    ;      BLET STRNG0 = "PUMP AN-102 SHUTDOWN  "
673      ;      BLET STRNG1 = "P-AN-102"
674      ;      GOTO AMSG
675      S48:    ;      BLET STRNG0 = "PUMP AN-103 SHUTDOWN  "
676      ;      BLET STRNG1 = "P-AN-103"
677      ;      GOTO AMSG
678      S49:    ;      BLET STRNG0 = "PUMP AN-104 SHUTDOWN  "
679      ;      BLET STRNG1 = "P-AN-104"
680      ;      GOTO AMSG
681      S50:    ;      BLET STRNG0 = "PUMP AN-105 SHUTDOWN  "
682      ;      BLET STRNG1 = "P-AN-105"
683      ;      GOTO AMSG
684      S51:    ;      BLET STRNG0 = "PUMP AN-106 SHUTDOWN  "
685      ;      BLET STRNG1 = "P-AN-106"
686      ;      GOTO AMSG
687      S52:    ;      BLET STRNG0 = "PUMP AN-107 SHUTDOWN  "
688      ;      BLET STRNG1 = "P-AN-107"
689      ;      GOTO AMSG
690      ;
691      ;--mct1-
692
693      S53:    LET STRNG0 = "ANBATCH PROGRAM START  "
694      GOTO AMSG
695      S54:    LET STRNG0 = "ANBATCH IN WRONG UNIT  "
696      GOTO AMSG
697      S55:    LET STRNG0 = "ANBATCH TERMINATING  "
698      GOTO AMSG
699
700      ;
701      MSG:    IF (FPASS)THEN
702      BEGIN
703      PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNG1
704      LET FPASS = 0
705      GOTO NXT
706      END
707
708      IF PTYPE = 4 THEN LET COLORO = 3
709      ELSE LET COLORO = PTYPE
710      LET EPNSO = STRNG1
711      INO = TIMSTR(SSNGO)
712      IF INO != 8 THEN LET SSNGO = "TIMEFAIL"
713      INO = APPEND(LSGO,STRNGO,SSNGO)
714      IF INO < 0 THEN LSGO = "APPEND FAILED!!"
715      INO = DATSTR(SDATEO)
716      IF INO != 9 THEN SDATEO = "DATE FAIL"
717      IF PSKID = 1 THEN LET GRAFXO = 272
718      IF PSKID = 2 THEN LET GRAFXO = 273
719      IF PSKID = 3 THEN LET GRAFXO = 273

```

```

716 0434      IF PSKID = 4 THEN LET GRAFXO = 274
717 0436      IF PSKID = 5 THEN LET GRAFXO = 270
718 0438      IF PSKID = 0 THEN LET GRAFXO = 270
719 0440      IF PTYPE = 1 THEN PRINT ADI,"WHITE ALARM %p %p GRAPHICS %i",EPNSO,STRNGO,GRAFXO
720 0441  NXT:  IF FSTART = 1 THEN
721              BEGIN
722                  LET COLORO = 0
723                  LET GRAFXO = 0
724                  LET STRNG1 = " "
725                  LET FSTART = 0
726                  PRINT ADI,"%p",STRNGO
727              END
728 0447      RETURN
729
730 0448  ERRSUB: IF ERRN != ERRNUM() THEN
731              BEGIN
732                  LET ERRN = ERRNUM()
733                  LET ERRL = ERRLIN()
734                  PRINT ADI,"ERROR # %i AT LINE %i, ILOOP = %i",errn,errl,ILOOP
735              END
736 0452      ERSCLR
737 0453      GOTO AGAIN
738
739      ;
740 0454  TERM:  LET FABORT = 0
741 0455  TERM:  STRNGO = "ANBATCH TERMINATED "
742 0456      LET ILOOP = 55
743 0457      LET FSTART = 1
744      GOSUB CYCL
745      END

```

Total Errors Detected 0 Total Warnings 0
 Sequence Program Size = 8864 Bytes, 554 Pages
 112 Forward Reference(s)

APBATCH

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

06:44:22 16-May-95

Page 1

```

1      ;      A P B A T C H . S E Q                      VERSION 3
2      ;
3      ;      Function:      Sequence program for Lamp function,
4      ;                      message display on graphics, and
5      ;                      printer messages.
6      ;
7      ;      Written By      : R. M. Kersey
8      ;      Date           : 05-31-92
9      ;      Facility       : 242-A Evaporator Bldg.
10     ;      Machine        : VAX 4104-A
11     ;      Op. System     : VAX/VMS 5.5-2AH
12     ;      D/3 Version    : 6.3-2 BASILINE
13     ;      Compiler       : SIC
14     ;      Build command  : @320,20)APBATCH
15     ;
16     ;      UNIT = 2, SKDKEY = 4, 5, 6, 7
17     ;
18     ;      Assoc. UPF:      [300,1]_TNKFRM.____ TNKFRM.LST
19     ;      Assoc. RPF:      [300,1]_FARM.____ FARM.LST
20     ;      SDB:            [300,1]PCM1ASDB.____ PCM1ASDB.CMD
21     ;
22     ;      POINTER      SDB VAR.      FUNCTION
23     ;      -----
24     ;      PSTAT      INTn      Current EPN alarm state is an integer tied to
25     ;                      graphic flashing string of value:
26     ;      .      (n=0-100)      0 = previously not alarmed
27     ;      .                      1 = previously alarmed, not ack
28     ;      .                      2 = previously alarmed, is ack
29     ;      .                      3 = alarm is inhibited
30     ;      .                      4 = input is off scan
31     ;      .                      5 = input is bad
32     ;
33     ;      PDIN      'EPN'      ---
34     ;      PTYPE      IALMn      alarm type 0 = none, 1 = white, 2 = yellow,
35     ;                      3 = yellow and yellow, 4 = red,
36     ;                      6 = yellow and red
37     ;      PREV      FXn      Reverse acting (not fail safe) flag (1=set)
38     ;      PSKID      SKIDn      Tracks which graphic has an alarm
39     ;      PPASS      FPASn      If set (1), turn lamp off when alarm is ACKnowledged
40     ;      PSN      FSNn      Flag for the graphic to ACKnowledge
41     ;      ---      FHOLD      Flag to update the lamps
42     ;      ---      FNING      Flag for DOG1A.C micro select
43     ;      ---      ACK      Flag for C program to ACK system wide
44     ;
45     ;      MODIFICATION LOG:
46     ;
47     ;      12-DEC-91      30 Second SKID update added.      R. Kersey
48     ;
49     ;      -mct1+ 14-MAY-93      Added FALMON for DEV pump shutdown.      M. Teats
50     ;
51     ;      11-JUN-93      Added comments and baselined for      M. Teats
52     ;                      D/3 version 3.3. CCN# 93JUN01-01.
53     ;
54     ;      15-DEC-94      Revised header to indicate the new VMS
55     ;                      system. Rebaselined document. Removed
56     ;                      obsolete Select() statements.

```

```

56 ;
57 ; 15-MAY-95 Revised to delete the grout signals and J. Berger
58 ; skid key 22. See EE-95-00126 and ECN
59 ; 620366. (JFB01)
60 ;
61 ; END MOD LOG
62 ;
63 UNIT APFARM
64 ;
65 GROUT == FLSPIT
66 AUTO == 'FQI-APSP1':CB_FLG:1 ; For FQI-RESET auto mode check
67 CALOUT == 'FQI-APSP1':CB_OTVL:1 ; For FQI-RESET calc block outvalue
68 ;
69 0000 ON HOLD GOSUB HLD
70 ;
71 0001 GETUNIT TRIP ; Which sequence unit is program running in
72 0002 IF TRIP != 2 THEN ; Abort program if running in wrong unit
73 BEGIN
74 FSTART = 1
75 ILOOP = 95
76 GOSUB CYCL
77 FSTART = 1
78 0007 PRINT ADL," >>>> ATTEMPT TO RUN APBATCH IN WRONG UNIT"
79 0008 GOTO TERM
80 END
81 ; IF ISELECT() THEN GOTO AGAIN
82 ;
83 ; BEGIN INITIALIZING:
84 ;
85 0009 NOP
86 0010 ERRN = 0
87 0011 ERRL = 0
88 ;
89 0012 INIT: LET FABORT = 0 ; If FABORT = 1 then abort program
90 0013 LET FSTART = 0 ; Program start flag
91 0014 LET FLDALM = 0
92 0015 LET FSL7 = 0 ; For FQI-RESET manual mode flag
93 0016 LAMP 0,7,0 ; Turn off the lamps
94 0017 LAMP 0,6,0
95 0018 LAMP 0,5,0
96 0019 LAMP 0,4,0
97 0020 LAMP 0,63,0
98 ; LAMP 0,22,9 ; JFB01
99 0021 ASSIGN PDEV = APFARM ; JFB01 ; First of 6 alarm ack devices
100 0022 FOR ILOOP = 1 TO 5 ; JFB01
101 0023 PDEV:DEV_TMPO = 1 ; Set all devices to ack state
102 0024 IF AND(PDEV:DEV_ALMCT,16) THEN ; If alarm device is inha, remove the inha
103 0025 PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
104 0026 IF AND(PDEV:DEV_ALMCT,32) THEN ; If alarm device is ack, clear it
105 0027 PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
106 0028 STEP PDEV BY 1
107 0029 NEXT ILOOP
108 ;
109 0030 FARM = 0 ; LAMP 7 LEAKO
110 0031 SERPIT = 1 ; not used

```

```

111 0032      VLVPI1 = 2      ; LAMP 5      LEAK2
112 0033      HVAC = 3      ; LAMP 6      LEAK3
113 0034      BLD271 = 4      ; LAMP 4      LEAK4
114      ;      GROUT = 5      ; LAMP22      LEAK5      JFB01
115      ;
116 0035      LEAK0 = 7      ; FARM
117 0036      LEAK1 = 63      ; not used
118 0037      LEAK2 = 5      ; VALVE PIT
119 0038      LEAK3 = 6      ; HVAC
120 0039      LEAK4 = 4      ; AW-271
121      ;      LEAK5 = 22      ; AP-102 GROUT      JFB01
122      ;
123      ;      First, set all flags and integers:
124      ;
125 0040      ASSIGN PDIN = 'YS-AP-101'      ; Assign pointer to first DIN
126 0041      ASSIGN PSTAT = INTO
127 0042      ASSIGN PTYPE = IALMO
128 0043      ASSIGN PREV = FXO
129 0044      ASSIGN PSKID = SKID0
130 0045      ASSIGN PPASS = FPASO
131      ;
132      ;      Note that iloop 0 is string 1 (S1:), and so on, in Alarm Typer String section
133      ;
134 0046      FOR ILOOP = 0 TO 82      ; For DINs
135 0047      LET PPASS = 0
136 0048      LET PREV = 0
137 0049      LET PTYPE = 2
138 0050      LET PSTAT = 0
139 0052      IF ILOOP = 30 THEN GOTO LOP2      ; JFB01
140 0054      IF ILOOP = 31 THEN GOTO LOP2      ; JFB01
141 0056      IF ILOOP = 32 THEN GOTO LOP2      ; JFB01
142 0058      IF ILOOP = 33 THEN GOTO LOP2      ; JFB01
143 0060      IF ILOOP = 34 THEN GOTO LOP2      ; JFB01
144 0062      IF ILOOP = 35 THEN GOTO LOP2      ; JFB01
145 0064      IF ILOOP = 36 THEN GOTO LOP2      ; JFB01
146 0066      IF ILOOP = 37 THEN GOTO LOP2      ; JFB01
147 0068      IF ILOOP = 82 THEN GOTO LOP2      ; JFB01
148 0069      IF AND(PDIN:DIN_DFLG,3) THEN      ; Check for alarms (must be red or yellow)
149      BEGIN
150      LET PTYPE = 2      ; Default to yellow alarm
151 0072      IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1      ; Reverse acting
152 0074      IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4      ; Set to red critical alarm
153      END
154 0076      IF ILOOP < 29 THEN LET PSKID = FARM
155      IF ILOOP = 3 THEN LET PSKID = GROUT      ; JFB01
156 0078      IF (ILOOP > 28 & ILOOP < 38) THEN LET PSKID = FARM      ; JFB01
157 0080      IF (ILOOP > 37 & ILOOP < 51) THEN LET PSKID = BLD271
158 0082      IF (ILOOP > 50 & ILOOP < 54) THEN LET PSKID = VLVPI1
159 0084      IF (ILOOP > 53 & ILOOP < 67) THEN LET PSKID = HVAC
160 0086      IF (ILOOP > 66 & ILOOP < 70) THEN LET PSKID = VLVPI1
161 0088      IF (ILOOP = 70 & ILOOP = 71) THEN LET PSKID = VLVPI1
162 0090      IF (ILOOP = 72 & ILOOP = 73) THEN LET PSKID = FARM      ; JFB01
163 0092      IF (ILOOP = 74 & ILOOP = 75) THEN LET PSKID = BLD271
164 0094      IF (ILOOP > 75 ) THEN LET PSKID = VLVPI1
165 0096      IF (ILOOP = 82 ) THEN LET PSKID = FARM      ; JFB01

```


Sequence Instruction Compiler Listing

06:44:22 16-May-95

Page 4

PAGE 574

```

221 0137      IALM77 = 1
222 0138      IALM78 = 1
223 0139      IALM79 = 1
224 0140      IALM80 = 1
225 0141      IALM81 = 1
226          ;
227 0142      FX38 = 1
228 0143      FX39 = 1
229          ;
230          ;
231          ;
232          ;
233 0144      FPAS1 = 0
234 0145      FPAS0 = 1
235 0146      FPAS3 = 1
236 0147      FPAS6 = 1
237 0148      FPAS12 = 1
238 0149      FPAS15 = 1
239 0150      FPAS20 = 1
240 0151      FPAS23 = 1
241 0152      FPAS26 = 1
242          ;
243          ;
244          ;
245 0153      FX0 = 1
246 0154      FX3 = 1
247 0155      FX6 = 1
248 0156      FX12 = 1
249 0157      FX15 = 1
250 0158      FX20 = 1
251 0159      FX23 = 1
252 0160      FX26 = 1
253          ;
254          ;
255          ;
256          ;
257          ;
258          ;
259          ;
260 0161      ILOOP = 94
261 0162      FSTART = 1
262 0163      TIMERCLR STM2
263 0164      TIMERON STM2
264 0165      GOSUB CYCL
265 0166      ON ERROR GOSUB ERRSUB
266          ;
267          ;
268          ;
269 0167  AGAIN:  MARK STEP "AGAIN",0,0
270          ;
271          ;
272          ;
273          ;
274          ;
275 0169      IF !SELECT() THEN FNING = 0
                ELSE FNING = 1
                WAITUNTIL SELECT() ! FABORT
                IF (FABORT) THEN GOTO TERM
                IF FNING = 0 THEN FHOLD = 1
                IF STM2 > 30 THEN FHOLD = 1

```

JFB01
JFB01

JFB01
JFB01

; Flag 480 determines if selected micro for DOG1A.C
; Wait until micro selected or program aborted
; Terminate program if abort flag set

```

276 0170      WAIT 1
277          ;
278 0171      DEV0 = 0
279 0172      DEV1 = 0
280 0173      DEV2 = 0
281 0174      DEV3 = 0
282 0175      DEV4 = 0
283          ;
284          ;
285          ;
286          ;
287          ;
288          ;
289          ;
290          ;
291          ;
292          ;
293          ;
294          ;
295          ;
296 0176      ASSIGN PDEV = APFARM
297 0177      ASSIGN PSN = FSND
298 0178      ASSIGN PACK = FLEAK0
299 0179      FOR ILOOP = 0 TO 4
300 0180          LET PSN = 0
301 0181      IF (PDEV:DEV_TMPO = 2) & AND(PDEV:DEV_ALMCT,32) THEN
302          BEGIN
303 0182          PDEV:DEV_TMPO = 1
304 0183          LET PSN = 1
305 0184          LET FLGACK = 1
306 0185          LET ACK = 1
307 0186          LET PACK = 1
308 0187          PHOLD = 1
309 0188          ASSIGN PMSG0 = LEAK0
310 0189          STEP PMSG0 BY ILOOP
311 0191          IF ILOOP != 1 THEN LAMP 0,PMSG0,1
312          END
313 0192      STEP PDEV BY 1
314 0193      STEP PSN BY 1
315 0194      STEP PACK BY 1
316 0195      NEXT ILOOP
317          ;
318 0196      LEAK0 = 7
319 0197      LEAK1 = 63
320 0198      LEAK2 = 5
321 0199      LEAK3 = 6
322 0200      LEAK4 = 4
323          ;
324          ;
325          ;
326          ;
327 0201      IF ACK THEN
328          BEGIN
329 0202          TIMERCLR STMR1
330 0203          TIMERON STMR1

```

JFB01

If alarm device is alarmed and its ALMCT indicates alarms are ack'd:

DEVICE	FLAG	OFFSET
APFARM	FLEAK0	447
AP-SERPIT	FLEAK1	448
AP-VLVPIT	FLEAK2	449
AP-IVAC	FLEAK3	450
AP-271	FLEAK4	451
GROUT	FLEAK5	452

NO LONGER USED JFB01

; JFB01

```

; Put device in ack state
; Set flag for the graphic to ack
; WATCH.C spawns ACKFLG.C which resets flag
; Set flag for C program to ACK system wide

```

; Flash lamp

JFB01

Field Alarm Silence feature:

```

331 0204          FLDALM = 1
332              END
333 0206          IF FLDALM THEN IF STMR1 >= 30 THEN      ; If field alarm then ack in 30 sec
334              BEGIN
335 0207              TIMEROFF STMR1
336 0208              TIMERCLR STMR1
337 0209              FLDALM = 0
338 0210              OPEN 'HS-AP-SIL'                      ; Silence alarm from PCM
339              END
340              ;
341              ; If field alarm ack then set flag and start timer:
342              ;
343 0212          IF RESET THEN IF LMPVAR < 2 THEN
344              BEGIN
345 0213              FLDALM = 1
346 0214              TIMERON STMR1
347 0215              RESET = 0
348              END
349 0217          IF LMPVAR = 2 THEN RESET = 1
350              ;
351              ;
352              ;
353 0218          ASSIGN PDIN = 'YS-AP-101'                  ; First DIN in APFARM
354 0219          ASSIGN PTYPE = IALMO
355 0220          ASSIGN PREV = FXO
356 0221          ASSIGN PSTAT = INTO
357 0222          ASSIGN PSKID = SKID0
358 0223          ASSIGN PPASS = FPAS0
359 0224          CURENT = 0                                  ; Used to set lamp status
360 0225          NXT = 0                                     ; Temporary INT
361 0226          BITVAL = 0
362              ;
363 0227          LOP:  FOR ILOOP = 1 TO 83
364 0229              IF ILOOP = 31 THEN GOTO LOP1          ; JF801
365 0231              IF ILOOP = 32 THEN GOTO LOP1          ; JF801
366 0233              IF ILOOP = 33 THEN GOTO LOP1          ; JF801
367 0235              IF ILOOP = 34 THEN GOTO LOP1          ; JF801
368 0237              IF ILOOP = 35 THEN GOTO LOP1          ; JF801
369 0239              IF ILOOP = 36 THEN GOTO LOP1          ; JF801
370 0241              IF ILOOP = 37 THEN GOTO LOP1          ; JF801
371 0243              IF ILOOP = 38 THEN GOTO LOP1          ; JF801
372 0245              IF ILOOP = 83 THEN GOTO LOP1          ; JF801
373 0246              IF PTYPE THEN
374                  BEGIN
375 0248                  IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3 ;be1      ; If some alarm
376 0250                  IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5 ; If EPN is inha
377 0252                  IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4 ; If EPN is ibad
378 0255                  IF ! PSTAT THEN IF PREV = PDIN:DIN_VAL THEN GOSUB ALM ; If EPN is dis
379 0256                  IF PSTAT THEN
380                      BEGIN
381 0257                      IF PSTAT < 3 THEN
382                          BEGIN
383 0258                      IF PSTAT = 1 THEN
384                          BEGIN
385 0259                      IF AND(PDIN:DIN_ALMCT,32) THEN

```

```

386          BEGIN                                ;be5
387 0260      LET PSTAT = 2
388 0261      F HOLD = 1
389          END                                ;en5
390 0262      IF ACK THEN
391          BEGIN                                ;be6 - go find which graphic
392          CRIT
393 0264      ASSIGN PSN = FSN0                    ; Routine to find graphic
394 0265      STEP PSN BY PSKID
395 0266      IF PSN THEN                          ; If this is the graphic
396          BEGIN                                ;be7
397 0267      LET PSTAT = 2
398 0269      IF PTYP = 1 THEN PDIN:DIN_ALMCT = OR(PDIN:DIN_ALMCT,32)
399 0270      FPASS = 1
400 0271      GOSUB ACKK                          ; Print signal was ack
401          END                                ;en7
402          END                                ;en6
403          END                                ;en4
404 0273      IF PREV != PDIN:DIN_VAL THEN GOSUB CLR ; If alarm has cleared
405 0274      ASSIGN PMSG0 = DEVO                  ; Find active alarms & graphic:
406 0275      STEP PMSG0 BY PSKID
407 0276      IF PPASS THEN
408          BEGIN
409 0278      IF PSTAT = 1 THEN PMSG0 = OR(PMSG0,PSTAT)
410          END
411 0279      ELSE PMSG0 = OR(PMSG0,PSTAT)
412          END                                ;en3
413 0280      ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8)THEN
414          BEGIN
415 0281      PSTAT = 0
416 0282      F HOLD = 1
417          END
418          END                                ;en2
419          END                                ;en1
420 0284      ELSE IF PSTAT THEN PSTAT = 0
421 0285      STEP PDIN BY 1                      ; JF801
422 0286      STEP PTYP BY 1
423 0287      STEP PREV BY 1
424 0288      STEP PSTAT BY 1
425 0289      STEP PSKID BY 1
426 0290      STEP PPASS BY 1
427 0291      NEXT ILOOP
428      ;
429      ;
430      ;
431 0292      DEV6 = 0
432 0293      DEV7 = 0
433 0294      DEV8 = 0
434 0295      DEV9 = 0
435 0296      DEV10 = 0
436 0297      DEV11 = 0
437      ;
438      ;
439      ;
440 0298  AUX:  ASSIGN PEPN = 'P-AP-101'

```

```

441 0299      FOR ILOOP = 84 TO 91                                ; JFB01
442 0300      IF PTYPE THEN
443          BEGIN
444 0302          IF AND(PEPN:DEV_ALMCT,16) THEN LET PSTAT = 3      ; If inha
445 0304          IF AND(PEPN:DEV_ALMCT,4) THEN LET PSTAT = 5      ; If ibad
446 0306          IF !AND(PEPN:DEV_FLG,8) THEN LET PSTAT = 4        ; If dis
447 0307          IF PSTAT < 3 THEN                                  ; If not inha
448              BEGIN
449 0308              IF (PSTAT = 1) & AND(PEPN:DEV_ALMCT,32) THEN    ; If prev unack alarm, and is now ack
450                  BEGIN
451 0309                  LET PSTAT = 2
452 0310                  FHOLD = 1
453                  END
454 0311              IF ACK THEN
455                  BEGIN
456 0312                  ASSIGN PSN = FSN0
457 0313                  STEP PSN BY PSKID
458 0314                  IF PSN & PSTAT = 1 THEN
459                      BEGIN
460 0315                      LET PSTAT = 2
461 0317                      IF PTYPE = 1 THEN PEPN:DEV_ALMCT = OR(PEPN:DEV_ALMCT,32)
462 0318                      PPASS = 1
463 0319                      GOSUB ACKK
464                      END
465                  END
466 0320                  IF !PSTAT THEN                                ; If prev not alarmed
467                      BEGIN
468 0321                      IF AND(PEPN:DEV_STAT,16) THEN            ; If Shutdown
469                          BEGIN
470 0322                          LET PSTAT = 1
471 0323                          FALMON = 1                                ; -mct1+
472 0325                          IF AND(PEPN:DEV_ALMCT,32) THEN PEPN:DEV_ALMCT = XOR(PEPN:DEV_ALMCT,32)
473 0326                          ASSIGN PDEV = APFARM
474 0327                          STEP PDEV BY PSKID
475 0328                          PDEV:DEV_TMPO = 2
476 0329                          GOSUB CYCL
477                          END
478                      ELSE IF !AND(PEPN:DEV_STAT,16) THEN
479                          BEGIN
480 0330                          LET PSTAT = 0
481 0331                          IF AND(PEPN:DEV_ALMCT,32) THEN PEPN:DEV_ALMCT = XOR(PEPN:DEV_ALMCT,32)
482 0333                          FHOLD = 1
483 0334                          END
484                      IF PSTAT THEN
485 0335                          BEGIN
486 0336                          ASSIGN PMSG0 = DEV6
487 0337                          STEP PMSG0 BY PSKID
488 0338                          IF PPASS THEN
489                              BEGIN
490 0340                              IF PSTAT = 1 THEN PMSG0 = OR(PMSG0,PSTAT)
491                              END
492                          ELSE PMSG0 = OR(PMSG0,PSTAT)
493 0341                          END
494                      END
495                  END

```

```

496 0342      ELSE IF !AND(PEPN:DEV_ALMCT,16)&!AND(PEPN:DEV_ALMCT,4)&AND(PEPN:DEV_FLG,8)THEN
497          BEGIN
498 0343      PSTAT = 0
499 0344      FHOLD = 1
500          END
501      END
502 0345      STEP PEPN BY 1
503 0346      STEP PTYPE BY 1
504 0347      STEP PREV BY 1
505 0348      STEP PSTAT BY 1
506 0349      STEP PSKID BY 1
507 0350      STEP PPASS BY 1
508 0351      NEXT ILOOP
509      ;
510      ;      END WHITE ALARMS FOR PUMP DEVICES
511      ;
512 0352  S2ND:  ACK = 0
513 0353      SORT = 0
514      ;
515 0354      DEV12 = OR(DEV0,DEV6)
516 0355      DEV13 = OR(DEV1,DEV7)
517 0356      DEV14 = OR(DEV2,DEV8)
518 0357      DEV15 = OR(DEV3,DEV9)
519 0358      DEV16 = OR(DEV4,DEV10)
520 0359      DEV17 = OR(DEV5,DEV11)
521      ;
522 0360      ASSIGN PMSG0 = DEV12
523 0361      ASSIGN PMSG1 = FARM
524 0362      FOR LLOOP = 1 TO 6
525 0363      IF PMSG0 THEN
526          BEGIN
527 0365      IF AND(PMSG0,1) THEN PMSG1 = 2
528 0366      ELSE PMSG1 = 1
529          END
530 0367      ELSE PMSG1 = 0
531 0368      STEP PMSG0 BY 1
532 0369      STEP PMSG1 BY 1
533 0370      NEXT LLOOP
534      ;
535 0371      MARK STEP "BOTTOM",0,0
536 0372      NOCRIT
537      ;
538      ;      BEGIN FQI - RESET :
539      ;
540      ;      Function:      For totalizers that have a CALC block which performs the
541      ;                      integration, set the output of the CALC block to zero
542      ;                      after the FQI loop has been placed in AUTO from MANUAL.
543      ;
544 0374      IF !AND(AUTO,64) THEN FSL7 = 1 ; If manual, set manual flag
545 0375      IF AND(AUTO,64) & AND(FSL7,1) THEN ; For manual-auto transition
546          BEGIN
547 0376      FSL7 = 0
548 0377      CALOUT = 0.0
549          END
550      ;

```

```

551      ;      END FQI - RESET
552      ;
553      0378      IF FHOLD THEN
554          BEGIN
555              0379          LAMP 0,7,FARM
556              0380          LAMP 0,6,HVAC
557              0381          LAMP 0,5,VLVPIT
558              0382          LAMP 0,4,BLD271
559              ;          LAMP 0,22,GROUT          JFB01
560              0383          FHOLD = 0
561              0384          TIMERCLR STM2
562              0385          TIMERON STM2
563              END
564      ;
565      0386      GOTO AGAIN
566      ;
567      ;      END PROGRAM LOOP
568      ;
569      ;      BEGIN CLEAR, ALARM, CYCLE, ACKNOWLEDGE, MESSAGE, ERROR, AND TERMINATE SUBROUTINES:
570      ;
571      0387      CLR:      LET PSTAT = 0
572      0389          IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
573      0390          FHOLD = 1
574      0391          RETURN
575      ;
576      0392      ALM:      LET PSTAT = 1          ; Print message, flash lamp, horn on, scroll message at bottom of graphic
577      0393          CRIT          ; Critical
578      0395          IF AND(PTYPE,3) THEN FALMON = 1          ; Audible alarm on
579      0396          IF AND(PDIN:DIN_ALMCT,32) THEN          ; Clear the scan block
580      0397              PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
581      0398          ASSIGN PMSG0 = LEAKO
582      0399          STEP PMSG0 BY PSKID
583      0400          LAMP 0,PMSG0,2          ; Lamp flash
584      0401          ASSIGN PDEV = APFARM          ; Find dev and turn it on
585      0402          STEP PDEV BY PSKID
586      0403          PDEV:DEV_TMP0 = 2
587      ;
588      ;      Scrolling alarm summary region in graphic (Message, time, date, alarm type):
589      ;
590      0404      CYCL:      LET LSG4 = LSG3          ; LSG = message
591      0405          LET LSG3 = LSG2
592      0406          LET LSG2 = LSG1
593      0407          LET LSG1 = LSG0
594      0408          LET COLOR4 = COLOR3          ; COLOR = type of alarm
595      0409          LET COLOR3 = COLOR2
596      0410          LET COLOR2 = COLOR1
597      0411          LET COLOR1 = COLOR0
598      0412          LET EPMS4 = EPMS3          ; EPN
599      0413          LET EPMS3 = EPMS2
600      0414          LET EPMS2 = EPMS1
601      0415          LET EPMS1 = STRNG1
602      0416          LET GRAFX4 = GRAFX3          ; GRAFX = graphic number
603      0417          LET GRAFX3 = GRAFX2
604      0418          LET GRAFX2 = GRAFX1
605      0419          LET GRAFX1 = GRAFX0

```



```

606 0420      LET SDATE4 = SDATE3
607 0421      LET SDATE3 = SDATE2      ; SDATE = time and date
608 0422      LET SDATE2 = SDATE1
609 0423      LET SDATE1 = SDATE0
610
611      ;
612      ;
613      ;
614      ACKK:  FOLD = 1
615      ON ILOOP GOTO S1,S2,S3,S4,S5,S6,S7,S8,S9,S10
616      ON ILOOP - 10 GOTO S11,S12,S13,S14,S15,S16,S17,S18,S19,S20
617      ON ILOOP - 20 GOTO S21,S22,S23,S24,S25,S26,S27,S28,S29,S30
618      ON ILOOP - 30 GOTO S31,S32,S33,S34,S35,S36,S37,S38,S39,S40
619      ON ILOOP - 40 GOTO S41,S42,S43,S44,S45,S46,S47,S48,S49,S50
620      ON ILOOP - 50 GOTO S51,S52,S53,S54,S55,S56,S57,S58,S59,S60
621      ON ILOOP - 60 GOTO S61,S62,S63,S64,S65,S66,S67,S68,S69,S70
622      ON ILOOP - 70 GOTO S71,S72,S73,S74,S75,S76,S77,S78,S79,S80
623      ON ILOOP - 80 GOTO S81,S82,S83,S84,S85,S86,S87,S88,S89,S90
624      ON ILOOP - 90 GOTO S91,S92,S93,S94,S95,S96
625      GOTO AMSG
626      S1:    STRNG0 = "AP-101 PUMP MCC OFF "
627      STRNG1 = "YS-AP-101"
628      GOTO AMSG
629      S2:    STRNG0 = "AP-101 ANN EX RAD HI "
630      STRNG1 = "RSH-101AP"
631      GOTO AMSG
632      S3:    STRNG0 = "AP-101 ANN MON FAIL "
633      STRNG1 = "RXS-101AP"
634      GOTO AMSG
635      S4:    STRNG0 = "AP-102 PUMP MCC OFF "
636      STRNG1 = "YS-AP-102"
637      GOTO AMSG
638      S5:    STRNG0 = "AP-102 ANN EX RAD HI "
639      STRNG1 = "RSH-102AP"
640      GOTO AMSG
641      S6:    STRNG0 = "AP-102 RAD MON FAIL "
642      STRNG1 = "RXS-102AP"
643      GOTO AMSG
644      S7:    STRNG0 = "AP-103 PUMP MCC OFF "
645      STRNG1 = "YS-AP103"
646      GOTO AMSG
647      S8:    STRNG0 = "AP-103 ANN EX RAD HI "
648      STRNG1 = "RSH-103AP"
649      GOTO AMSG
650      S9:    STRNG0 = "AP-103 RAD MON FAIL "
651      STRNG1 = "RXS-103AP"
652      GOTO AMSG
653      S10:   STRNG0 = "AP-03C RAD MON FAIL "
654      STRNG1 = "RSH-AP03C"
655      GOTO AMSG
656      S11:   STRNG0 = "AP-03C LK PIT RAD HI "
657      STRNG1 = "RXS-AP03C"
658      GOTO AMSG
659      S12:   STRNG0 = "AP-03D DRAN PIT LEAK "
660      STRNG1 = "LDS-AP03D"
661      GOTO AMSG

```

```

661 0472 S13:  STRNG0 = "AP-104 PUMP MCC OFF  "
662 0473      STRNG1 = "YS-AP-104"
663 0474      GOTO AMSG
664 0475 S14:  STRNG0 = "AP-104 ANN EX RAD HI  "
665 0476      STRNG1 = "RSH-104AP"
666 0477      GOTO AMSG
667 0478 S15:  STRNG0 = "AP-104 RAD MON FAIL  "
668 0479      STRNG1 = "RXS-104AP"
669 0480      GOTO AMSG
670 0481 S16:  STRNG0 = "PUMP AP-105 MCC OFF  "
671 0482      STRNG1 = "YS-AP-105"
672 0483      GOTO AMSG
673 0484 S17:  STRNG0 = "AP-105 ANN EX RAD HI  "
674 0485      STRNG1 = "RSH-105AP"
675 0486      GOTO AMSG
676 0487 S18:  STRNG0 = "AP-105 RAD MON FAIL  "
677 0488      STRNG1 = "RXS-105AP"
678 0489      GOTO AMSG
679 0490 S19:  STRNG0 = "AP-05C LK-PIT RAD HI  "
680 0491      STRNG1 = "RSH-AP05C"
681 0492      GOTO AMSG
682 0493 S20:  STRNG0 = "AP-05C RAD MON FAIL  "
683 0494      STRNG1 = "RXS-AP05C"
684 0495      GOTO AMSG
685 0496 S21:  STRNG0 = "PUMP AP-106 MCC OFF  "
686 0497      STRNG1 = "YS-AP-106"
687 0498      GOTO AMSG
688 0499 S22:  STRNG0 = "AP-106 ANN EX RAD HI  "
689 0500      STRNG1 = "RSH-106AP"
690 0501      GOTO AMSG
691 0502 S23:  STRNG0 = "106 EXH RAD MON FAIL  "
692 0503      STRNG1 = "RXS-106AP"
693 0504      GOTO AMSG
694 0505 S24:  STRNG0 = "PUMP AP-107 MCC OFF  "
695 0506      STRNG1 = "YS-AP-107"
696 0507      GOTO AMSG
697 0508 S25:  STRNG0 = "AP-107 ANN EX RAD HI  "
698 0509      STRNG1 = "RSH-107AP"
699 0510      GOTO AMSG
700 0511 S26:  STRNG0 = "107 EXH RAD MON FAIL  "
701 0512      STRNG1 = "RXS-107AP"
702 0513      GOTO AMSG
703 0514 S27:  STRNG0 = "PUMP AP-108 MCC OFF  "
704 0515      STRNG1 = "YS-AP-108"
705 0516      GOTO AMSG
706 0517 S28:  STRNG0 = "AP-108 ANN EX RAD HI  "
707 0518      STRNG1 = "RSH-108AP"
708 0519      GOTO AMSG
709 0520 S29:  STRNG0 = "108 EXH RAD MON FAIL  "
710 0521      STRNG1 = "RXS-108AP"
711 0522      GOTO AMSG
712 0523 S30:  STRNG0 = "AP-02D PUMP PIT LEAK  "
713 0524      STRNG1 = "LDSAP02DC"
714 0525      GOTO AMSG
715      S31:  ; STRNG0 = "AP-102-3 MCC OFF  "

```

JFB01

```

716      ;      STRNG1 ="YS-AP102M"      JFB01
717      ;      GOTO AMSG      JFB01
718      S32:    ;      STRNG0 = "AP-102-3 DIS PRES HI  "      JFB01
719      ;      STRNG1 ="PSHAP102M"      JFB01
720      ;      GOTO AMSG      JFB01
721      S33:    ;      STRNG0 = "AP-102-3 DIS PRES LO  "      JFB01
722      ;      STRNG1 ="PSLAP102M"      JFB01
723      ;      GOTO AMSG      JFB01
724      S34:    ;      STRNG0 = "AP-102-3 LEAK      "      JFB01
725      ;      STRNG1 ="LDSAP102M"      JFB01
726      ;      GOTO AMSG      JFB01
727      S35:    ;      STRNG0 = "AP-102-3 ROTATOR OFF  "      JFB01
728      ;      STRNG1 ="YS-AP102R"      JFB01
729      ;      GOTO AMSG      JFB01
730      S36:    ;      STRNG0 = "AP-102-3 STRAIN HIGH  "      JFB01
731      ;      STRNG1 ="MSHAP102M"      JFB01
732      ;      GOTO AMSG      JFB01
733      S37:    ;      STRNG0 = "AP-102-3 TEMP HIGH  "      JFB01
734      ;      STRNG1 ="TSHAP102M"      JFB01
735      ;      GOTO AMSG      JFB01
736      S38:    ;      STRNG0 = "AP-102-3 VIBRATIN HI  "      JFB01
737      ;      STRNG1 ="VSHAP102M"      JFB01
738      ;      GOTO AMSG      JFB01
739      0526    S39:    STRNG0 = "AP-MASTER SHUTDOWN  "
740      0527      STRNG1 ="YS-AP-MSD"
741      0528      GOTO AMSG
742      0529    S40:    STRNG0 = "AP-AREA RAD BYPASS  "
743      0530      STRNG1 ="YS-AP-ARM"
744      0531      GOTO AMSG
745      0532    S41:    STRNG0 = "AP-FARM RADIATION HI  "
746      0533      STRNG1 ="RSH-AP-2"
747      0534      GOTO AMSG
748      0535    S42:    STRNG0 = "AP-FARM RAD MON FAIL  "
749      0536      STRNG1 ="RXS-AP-2"
750      0537      GOTO AMSG
751      0538    S43:    STRNG0 = "AP GEMERAL ALARM  "
752      0539      STRNG1 ="QS-AP-1"
753      0540      GOTO AMSG
754      0541    S44:    STRNG0 = "AP HEAT TRACE OVRTMP  "
755      0542      STRNG1 ="TXS-AP-2"
756      0543      GOTO AMSG
757      0544    S45:    STRNG0 = "CENTRAL PMP PIT LEAK  "
758      0545      STRNG1 ="LDS-AP-A"
759      0546      GOTO AMSG
760      0547    S46:    STRNG0 = "ANNULUS PMP PIT LEAK  "
761      0548      STRNG1 ="LDS-AP-B"
762      0549      GOTO AMSG
763      0550    S47:    STRNG0 = "LEAK-DET PMPIT LEAK  "
764      0551      STRNG1 ="LDS-AP-C"
765      0552      GOTO AMSG
766      0553    S48:    STRNG0 = "271 ANNUNC POWER OFF  "
767      0554      STRNG1 ="JXS-AP-1"
768      0555      GOTO AMSG
769      0556    S49:    STRNG0 = "271 BLDG CAM FLOW LO  "
770      0557      STRNG1 ="FSLAP2711"

```

771	0558	GOTO AMSG
772	0559	STRINGO = "271 BLDG RADN HIGH "
773	0560	STRING1 = "RSH-AP271"
774	0561	GOTO AMSG
775	0562	STRINGO = "271 BLD RAD MON FAIL "
776	0563	STRING1 = "RXS-AP271"
777	0564	GOTO AMSG
778	0565	STRINGO = "FLSH PIT FL LINE PRS "
779	0566	STRING1 = "PSH-APFP1"
780	0567	GOTO AMSG
781	0568	STRINGO = "FLSH PIT V-FP-1 OPEN "
782	0569	STRING1 = "ZS-AP-FP1"
783	0570	GOTO AMSG
784	0571	STRINGO = "AP-FLUSH PIT LEAK "
785	0572	STRING1 = "LDS-AP-FP"
786	0573	GOTO AMSG
787	0574	STRINGO = "K1 EXHAUST FAN OFF "
788	0575	STRING1 = "YS-AP-K1"
789	0576	GOTO AMSG
790	0577	STRINGO = "K1 EXH SMPL FLOW LO "
791	0578	STRING1 = "FSL-APK13"
792	0579	GOTO AMSG
793	0580	STRINGO = "K1 EXHAUST RAD HIGH "
794	0581	STRING1 = "RSH-APK11"
795	0582	GOTO AMSG
796	0583	STRINGO = "K1 EXH RAD MON FAIL "
797	0584	STRING1 = "RXS-APK11"
798	0585	GOTO AMSG
799	0586	STRINGO = "K1 EX REC SMPL LOFLO "
800	0587	STRING1 = "FSL-APK12"
801	0588	GOTO AMSG
802	0589	STRINGO = "K1 EXH STACK FLOW LO "
803	0590	STRING1 = "FSL-APK11"
804	0591	GOTO AMSG
805	0592	STRINGO = "AP SEAL-POT PIT LEAK "
806	0593	STRING1 = "LDS-AP-K1"
807	0594	GOTO AMSG
808	0595	STRINGO = "K2 EXH FAN SHUTDOWN "
809	0596	STRING1 = "YS-AP-K2"
810	0597	GOTO AMSG
811	0598	STRINGO = "K2 EXH MON SMPL FLOLO "
812	0599	STRING1 = "FSL-APK23"
813	0600	GOTO AMSG
814	0601	STRINGO = "K2 EXH RADIATION HI "
815	0602	STRING1 = "RSH-APK21"
816	0603	GOTO AMSG
817	0604	STRINGO = "K2 EXH RADN MON FAIL "
818	0605	STRING1 = "RXS-APK21"
819	0606	GOTO AMSG
820	0607	STRINGO = "K2 EXH REC SMPL FLOLO "
821	0608	STRING1 = "FSL-APK22"
822	0609	GOTO AMSG
823	0610	STRINGO = "K2 EXH STACK FLOW LO "
824	0611	STRING1 = "FSL-APK21"
825	0612	GOTO AMSG

```

826 0613 S68:  STRNG0 = "AP SERVICE PIT LEAK  "
827 0614      STRNG1 ="LDS-AP-SP"
828 0615      GOTO AMSG
829 0616 S69:  STRNG0 = "AP SER PIT RAD HI  "
830 0617      STRNG1 ="RSH-AP-SP"
831 0618      GOTO AMSG
832 0619 S70:  STRNG0 = "AP SER PIT RAD FAIL  "
833 0620      STRNG1 ="RXS-AP-SP"
834 0621      GOTO AMSG
835 0622 S71:  STRNG0 = "AP SL-509/510 LEAK  "
836 0623      STRNG1 ="LDS-AP-SL"
837 0624      GOTO AMSG
838 0625 S72:  STRNG0 = "AP SN-609/610 LEAK  "
839 0626      STRNG1 ="LDS-AP-SN"
840 0627      GOTO AMSG
841 0628 S73:  STRNG0 = "AP SN-621 LINE LEAK  "
842 0629      STRNG1 ="LDSAP621A"
843 0630      GOTO AMSG
844 0631 S74:  STRNG0 = "AP SN-650 LINE LEAK  "
845 0632      STRNG1 ="LDSAP6502"
846 0633      GOTO AMSG
847 0634 S75:  STRNG0 = "AP TANKS EXCESS VACU  "
848 0635      STRNG1 ="PSH-AP-2"
849 0636      GOTO AMSG
850 0637 S76:  STRNG0 = "AP TANKS VACUUM LOSS  "
851 0638      STRNG1 ="PSL-AP-2"
852 0639      GOTO AMSG
853 0640 S77:  STRNG0 = "AP VALVE PIT LEAK  "
854 0641      STRNG1 ="LDS-AP-VP"
855 0642      GOTO AMSG
856 0643 S78:  STRNG0 = "DR-712 FLOW DETECTED  "
857 0644      STRNG1 ="FSH-DR712"
858 0645      GOTO AMSG
859 0646 S79:  STRNG0 = "DR-712 FLOW DET FAIL  "
860 0647      STRNG1 ="FXS-DR712"
861 0648      GOTO AMSG
862 0649 S80:  STRNG0 = "FLUSH LINE PRESS HIGH  "
863 0650      STRNG1 ="PSH-APFL1"
864 0651      GOTO AMSG
865 0652 S81:  STRNG0 = "AP VPIT V-DR-1 OPEN  "
866 0653      STRNG1 ="ZS-AP-DR1"
867 0654      GOTO AMSG
868 0655 S82:  STRNG0 = "AP VPIT V-FL-1 OPEN  "
869 0656      STRNG1 ="ZS-AP-FL1"
870 0657      GOTO AMSG
871      S83:  ; STRNG0 = "ROTOR 102-3 BYPASS ON  " ; JFB01
872      ; STRNG1 ="YS-1023BY"
873      ; GOTO AMSG
874 0658 S84:  STRNG0 = "PUMP AP-101 SHUTDOWN  "
875 0659      STRNG1 = "P-AP-101"
876 0660      GOTO AMSG
877 0661 S85:  STRNG0 = "PUMP AP-102 SHUTDOWN  "
878 0662      STRNG1 = "P-AP-102"
879 0663      GOTO AMSG
880 0664 S86:  STRNG0 = "PUMP AP-103 SHUTDOWN  "

```

```

881 0665      STRNG1 = "P-AP-103"
882 0666      GOTO AMSG
883 0667      S87:  STRNG0 = "PUMP AP-104 SHUTDOWN  "
884 0668          STRNG1 = "P-AP-104"
885 0669          GOTO AMSG
886 0670      S88:  STRNG0 = "PUMP AP-105 SHUTDOWN  "
887 0671          STRNG1 = "P-AP-105"
888 0672          GOTO AMSG
889 0673      S89:  STRNG0 = "PUMP AP-106 SHUTDOWN  "
890 0674          STRNG1 = "P-AP-106"
891 0675          GOTO AMSG
892 0676      S90:  STRNG0 = "PUMP AP-107 SHUTDOWN  "
893 0677          STRNG1 = "P-AP-107"
894 0678          GOTO AMSG
895 0679      S91:  STRNG0 = "PUMP AP-108 SHUTDOWN  "
896 0680          STRNG1 = "P-AP-108"
897 0681          GOTO AMSG
898          S92:  ;      STRNG0 = "PUMP AP-102M SHUTDOWN  "      JFB01
899          ;      STRNG1 = "P-AP-102M"                        JFB01
900          ;      GOTO AMSG                                    JFB01
901          S93:  ;      STRNG0 = "ROTR AP-102R SHUTDOWN  "      JFB01
902          ;      STRNG1 = "P-AP-102R"                        JFB01
903          ;      GOTO AMSG                                    JFB01
904 0682      S94:  STRNG0 = "APBATCH PROGRAM START  "
905 0683          GOTO AMSG
906 0684      S95:  STRNG0 = "APBATCH IN WRONG UNIT  "
907 0685          GOTO AMSG
908 0686      S96:  STRNG0 = "APBATCH TERMINATING  "
909 0687          GOTO AMSG
910          ;
911 0688      AMSG:  IF (FPASS)THEN
912                BEGIN
913 0689                PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNG1
914 0690                FPASS = 0
915 0691                GOTO NXT
916                END
917 0693                IF PTYPE = 4 THEN COLOR0 = 3
918 0694                ELSE COLOR0 = PTYPE
919 0695                EPNS0 = STRNG1
920 0696                INO = TIMSTR(SSNGO)
921 0698                IF INO != 8 THEN SSNGO = "TIMEFAIL"
922 0699                INO = APPEND(LSG0,STRNG0,SSNGO)
923 0701                IF INO < 0 THEN LSG0 = "APPEND FAILED!!!"
924 0702                INO = DATSTR(SDATE0)
925 0704                IF INO != 9 THEN SDATE0 = "DATE FAIL"
926 0705                GRAFX0 = 291 + PSKID
927 0707                IF PSKID = 5 THEN GRAFX0 = 291
928 0709                IF PSKID = 0 THEN GRAFX0 = 290
929 0711                IF PTYPE = 1 THEN PRINT ADI,"WHITE ALARM %p %p GRAPHICS %i",EPNS0,STRNG0,GRAFX0
930 0712      NXT:  IF FSTART = 1 THEN
931                BEGIN
932 0713                COLOR0 = 0
933 0714                GRAFX0 = 0
934 0715                STRNG1 = ""
935 0716                FSTART = 0

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

06:44:22 16-May-95

Page 18

```

936 0717      PRINT ADI,"%p",STRNGO
937          END
938 0718      RETURN
939      ;
940 0720  ERRSUB: IF(ERRNUM() = -187)THEN ERSRET
941          ELSE
942              BEGIN
943 0721          ERRN = ERRNUM()
944 0722          ERRL = ERRLIN()
945 0723          PRINT ADI,"ERROR # %i AT LINE %i, ILOOP = %i",errn,erri,ILOOP
946 0724          ERSCLR
947          END
948 0725          GOTO AGAIN
949      ;
950 0726  HLD:   HRSCLR
951 0727          WAITUNTIL HLDSTP != 0
952 0728          ON HLDSTP GOTO INIT,AGAIN,TERM
953      ;
954      ; TERM: FABORT = 0
955 0729  TERM:  STRNGO = 'APBATCH TERMINATED  '
956 0730          ILOOP = 96
957 0731          FSTART = 1
958 0732          TIMEROFF STMR1
959 0733          TIMERCLR STMR1
960 0734          FLDALM = 0
961 0735          OPEN 'HS-AP-SIL'
962 0736          GOSUB CYCL
963          END

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 14224 Bytes, 889 Pages
212 Forward Reference(s)

AWBATCH

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:33:38 24-Feb-95

Page 1

```

1      ;      A M B A T C H . S E Q      VERSION 2
2      ;
3      ;      Function:      Sequence program for Lamp function,
4      ;      message display on graphics, and
5      ;      printer messages.
6      ;
7      ;      Written By      : R. M. Kersey
8      ;      Date           : 05-31-92
9      ;      Facility       : 242-A Evaporator Bldg.
10     ;      Machine        : VAX 4104-A
11     ;      Op. System     : VAX/VMS 5.5-2AH
12     ;      D/3 Version    : 6.3-2 BASELINE
13     ;      Compiler       : SIC
14     ;      Build command  : @[320,20]AMBATCH
15     ;
16     ;      UNIT = 3, SKDKEY = 8, 9, 10, 11
17     ;
18     ;      Assoc. UPF:      [300,1]_TNKFRM.____   TNKFRM.LST
19     ;      Assoc. RPF:      [300,1]_FARM.____     FARM.LST
20     ;      SDB:            [300,1]PCMIASDB.____   PCMIASDB.CMD
21     ;
22     ;      POINTER      SDB VAR.      FUNCTION
23     ;      -----
24     ;      PSTAT      INTn      Current EPW alarm state is an integer tied to
25     ;      (n=0-100)      graphic flashing string of value:
26     ;      .              0 = previously not alarmed
27     ;      .              1 = previously alarmed, not ack
28     ;      .              2 = previously alarmed, is ack
29     ;      .              3 = alarm is inhibited
30     ;      .              4 = input is off scan
31     ;      .              5 = input is bad
32     ;      PDIN      'EPW'      ---
33     ;      PTYPE      IALMn      alarm type 0 = none, 1 = white, 2 = yellow,
34     ;      .              3 = yellow and yellow, 4 = red,
35     ;      .              6 = yellow and red
36     ;      PREV      FXn      Reverse acting (not fail safe) flag (1=set)
37     ;      PSKID      SKIDn      Tracks which graphic has an alarm
38     ;      PPASS      FPASn      If set (1), turn lamp off when alarm is ACKnowledged
39     ;      PSN      FSNn      Flag for the graphic to ACKnowledge
40     ;      ---      FHOLD      Flag to update the lamps
41     ;      ---      FMIING      Flag for DOG1A.C micro select
42     ;      ---      ACK      Flag for C program to ACK system wide
43     ;
44     ;      MODIFICATION LOG:
45     ;
46     ;      12-DEC-91      Include MCC shutdown, pump shutdown, and R. Kersey
47     ;      30 second SKID update.
48     ;
49     ;      05-APR-93      Modified MCC OFF white alarm on all 5      M. Teats
50     ;      YS- signals to pump OFF white alarm because
51     ;      the YS DINs do not indicate MCC status - they
52     ;      indicate the pump status.
53     ;
54     ;      -mct1+ 13-MAY-93      Remove pump shutdown DEV white alarms, which      M. Teats
55     ;      are based on the analog signals, because they

```

```

56      ;      are not needed due to the 4/5/93 mods. See
57      ;      CCN# 93MAY13-01.
58      ;
59      ;      10-JUN-93      Added comments and baselined for      M. Teats
60      ;      D/3 version 3.3. CCN# 93JUN01-01
61      ;
62      ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
63      ;      system. Rebaselined document. Removed
64      ;      obsolete Select() statement.
65      ;
66      ;      END MOD LOG
67      ;
68      UNIT      AWFARM
69      ;
70      PUMP == FLSPIT
71      AUTO == 'F01-AW-1':CB_FLG:1      ; For F01-RESET auto mode check
72      CALOUT == 'F01-AW-1':CB_OTVL:1      ; For F01-RESET calc block outvalue
73      ;
74      0000      ;      GETUNIT TRIP      ; Which sequence unit is program running
75      0001      IF TRIP != 3 THEN      ; Abort if running in wrong seq. unit
76      ;      BEGIN
77      0002      FSTART = 1
78      0003      ILOOP = 54
79      0004      GOSUB CYCL
80      0005      FSTART = 1
81      0006      PRINT ADI," >>>>> ATTEMPT TO RUN AWPATCH IN WRONG UNIT"
82      0007      GOTO TERM
83      ;      END
84      ;      IF !SELECT() THEN GOTO AGAIN
85      ;
86      ;      B E G I N   I N I T I A L I Z I N G :
87      ;
88      0008      NOP
89      0009      INIT:      LET FABORT = 0      ; If FABORT = 1 then abort program
90      0010      LET FSTART = 0      ; Program start flag = 0
91      0011      LET ERN = 0      ; Set error number to zero
92      0012      LET ERRL = 0      ; Set error line to zero
93      0013      LET FLDALM = 0
94      0014      LET FSL7 = 0      ; For F01-RESET manual mode flag
95      0015      LAMP 0,11,0      ; Turn off the lamps
96      0016      LAMP 0,10,0
97      0017      LAMP 0,9,0
98      0018      LAMP 0,8,0
99      0019      ASSIGN PDEV = AWFARM      ; First of 6 alarm ack devices
100     0020      FOR ILOOP = 1 TO 6
101     0021      PDEV:DEV_TMPO = 1      ; Set all devices to ack state
102     0022      IF AND(PDEV:DEV_ALMCT,16) THEN      ; If alarm device is inha, remove the inha
103     0023      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
104     0024      IF AND(PDEV:DEV_ALMCT,32) THEN      ; If alarm device is ack, clear it
105     0025      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
106     0026      STEP PDEV BY 1
107     0027      NEXT ILOOP
108     ;
109     ;      First, set all flags and integers:
110     ;

```

```

111 0028      ASSIGN PDIN = 'YS-AW-101'      ; Assign pointer to first DIN
112 0029      ASSIGN PSTAT = INTO
113 0030      ASSIGN PTYPE = IALMO
114 0031      ASSIGN PREV = FX0
115 0032      ASSIGN PSKID = SKID0
116 0033      ASSIGN PPASS = FPAS0
117
118 0034      ;      LET FARM = 0      ; Set up values for different graphics
119 0035      SERPIT = 1      ; SKID of each EPN indicates graphic
120 0036      VLVPIIT = 2
121 0037      HVAC = 3
122 0038      BLD271 = 4
123 0039      PUMP = 5
124
125 0040      ;
126 0041      LEAK0 = 11      ; FARM
127 0042      LEAK1 = 63      ; not used
128 0043      LEAK2 = 9      ; VALVE PIT
129 0044      LEAK3 = 10      ; HVAC
130      LEAK4 = 8      ; AW-271
131
132 0045      ;      FOR ILOOP = 0 TO 46      ; For DINs
133 0046      LET PPASS = 0
134 0047      LET PREV = 0
135 0048      IF ILOOP > 31 & ILOOP < 36 THEN GOTO SKP
136 0049      IF ILOOP = 44 THEN GOTO SKP
137 0050      LET PTYPE = 2
138 0051      BLET PSTAT = 0
139 0052      ;      IF AND(PDIN:DIN_DFLG,3) THEN      ; Check for alarms (must be red or yellow)
140      BEGIN
141      LET PTYPE = 2
142      IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1      ; Reverse acting
143      IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4      ; Set to red critical alarm
144      END
145 0054      IF ILOOP < 17 THEN LET PSKID = FARM      ; These are set by IPN order
146 0055      IF (ILOOP > 16 & ILOOP < 31) THEN LET PSKID = VLVPIIT
147 0056      IF ILOOP = 28 THEN LET PSKID = BLD271
148 0057      IF (ILOOP > 30 & ILOOP < 39) THEN LET PSKID = HVAC
149 0058      IF ILOOP > 38 THEN LET PSKID = BLD271
150 0059      IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
151 0071      SKP: STEP PSTAT BY 1
152 0072      STEP PDIN BY 1
153 0073      STEP PTYPE BY 1
154 0074      STEP PREV BY 1
155 0075      STEP PSKID BY 1
156 0076      STEP PPASS BY 1
157 0077      NEXT ILOOP
158
159      ;      ;-mct1- abandoned code for white alarms tied to pump devices:
160      ;
161      ;      FOR ILOOP = 47 TO 51
162      ;      PSTAT = 0
163      ;      PTYPE = 1
164      ;      PREV = 0
165      ;      PSKID = 0
166      ;      PPASS = 1

```

```

166      ; STEP PSTAT BY 1
167      ; STEP PIYPE BY 1
168      ; STEP PREV BY 1
169      ; STEP PSKID BY 1
170      ; STEP PASS BY 1
171      ; NEXT ILOOP
172      ;
173      ;
174      ; Set the white alarms:
175      ;
176      LET IALMO = 1
177      LET IALMS = 1
178      LET IALMB = 1
179      LET IALM1 = 1
180      LET IALM14 = 1
181      LET IALM32 = 0
182      LET IALM33 = 0
183      LET IALM34 = 0
184      LET IALM35 = 0
185      LET IALM44 = 0
186      LET IALM46 = 1
187      ;
188      FXO = 0
189      FX5 = 0
190      FX8 = 0
191      FX11 = 0
192      FX14 = 0
193      FX28 = 1
194      ;
195      FXASO = 1
196      FXAS8 = 1
197      FXAS11 = 1
198      FXAS14 = 1
199      ;
200      ;
201      ;
202      ;
203      ;
204      ;
205      ILOOP = 53
206      FSTART = 1
207      TIMERCLR STM2
208      TIMERON STM2
209      GOSUB CYCL
210      ON ERROR GOSUB ERRSUB
211      ;
212      ;
213      ;
214      ;
215      ;
216      ;
217      ;
218      ;
219      ;
220      ;
221      ;
222      ;
223      ;
224      ;
225      ;
226      ;
227      ;
228      ;
229      ;
230      ;
231      ;
232      ;
233      ;
234      ;
235      ;
236      ;
237      ;
238      ;
239      ;
240      ;
241      ;
242      ;
243      ;
244      ;
245      ;
246      ;
247      ;
248      ;
249      ;
250      ;
251      ;
252      ;
253      ;
254      ;
255      ;
256      ;
257      ;
258      ;
259      ;
260      ;
261      ;
262      ;
263      ;
264      ;
265      ;
266      ;
267      ;
268      ;
269      ;
270      ;
271      ;
272      ;
273      ;
274      ;
275      ;
276      ;
277      ;
278      ;
279      ;
280      ;
281      ;
282      ;
283      ;
284      ;
285      ;
286      ;
287      ;
288      ;
289      ;
290      ;
291      ;
292      ;
293      ;
294      ;
295      ;
296      ;
297      ;
298      ;
299      ;
300      ;
301      ;
302      ;
303      ;
304      ;
305      ;
306      ;
307      ;
308      ;
309      ;
310      ;
311      ;
312      ;
313      ;
314      ;
315      ;
316      ;
317      ;
318      ;
319      ;
320      ;
321      ;
322      ;
323      ;
324      ;
325      ;
326      ;
327      ;
328      ;
329      ;
330      ;
331      ;
332      ;
333      ;
334      ;
335      ;
336      ;
337      ;
338      ;
339      ;
340      ;
341      ;
342      ;
343      ;
344      ;
345      ;
346      ;
347      ;
348      ;
349      ;
350      ;
351      ;
352      ;
353      ;
354      ;
355      ;
356      ;
357      ;
358      ;
359      ;
360      ;
361      ;
362      ;
363      ;
364      ;
365      ;
366      ;
367      ;
368      ;
369      ;
370      ;
371      ;
372      ;
373      ;
374      ;
375      ;
376      ;
377      ;
378      ;
379      ;
380      ;
381      ;
382      ;
383      ;
384      ;
385      ;
386      ;
387      ;
388      ;
389      ;
390      ;
391      ;
392      ;
393      ;
394      ;
395      ;
396      ;
397      ;
398      ;
399      ;
400      ;
401      ;
402      ;
403      ;
404      ;
405      ;
406      ;
407      ;
408      ;
409      ;
410      ;
411      ;
412      ;
413      ;
414      ;
415      ;
416      ;
417      ;
418      ;
419      ;
420      ;
421      ;
422      ;
423      ;
424      ;
425      ;
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432      ;
433      ;
434      ;
435      ;
436      ;
437      ;
438      ;
439      ;
440      ;
441      ;
442      ;
443      ;
444      ;
445      ;
446      ;
447      ;
448      ;
449      ;
450      ;
451      ;
452      ;
453      ;
454      ;
455      ;
456      ;
457      ;
458      ;
459      ;
460      ;
461      ;
462      ;
463      ;
464      ;
465      ;
466      ;
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473      ;
474      ;
475      ;
476      ;
477      ;
478      ;
479      ;
480      ;
481      ;
482      ;
483      ;
484      ;
485      ;
486      ;
487      ;
488      ;
489      ;
490      ;
491      ;
492      ;
493      ;
494      ;
495      ;
496      ;
497      ;
498      ;
499      ;
500      ;
501      ;
502      ;
503      ;
504      ;
505      ;
506      ;
507      ;
508      ;
509      ;
510      ;
511      ;
512      ;
513      ;
514      ;
515      ;
516      ;
517      ;
518      ;
519      ;
520      ;
521      ;
522      ;
523      ;
524      ;
525      ;
526      ;
527      ;
528      ;
529      ;
530      ;
531      ;
532      ;
533      ;
534      ;
535      ;
536      ;
537      ;
538      ;
539      ;
540      ;
541      ;
542      ;
543      ;
544      ;
545      ;
546      ;
547      ;
548      ;
549      ;
550      ;
551      ;
552      ;
553      ;
554      ;
555      ;
556      ;
557      ;
558      ;
559      ;
560      ;
561      ;
562      ;
563      ;
564      ;
565      ;
566      ;
567      ;
568      ;
569      ;
570      ;
571      ;
572      ;
573      ;
574      ;
575      ;
576      ;
577      ;
578      ;
579      ;
580      ;
581      ;
582      ;
583      ;
584      ;
585      ;
586      ;
587      ;
588      ;
589      ;
590      ;
591      ;
592      ;
593      ;
594      ;
595      ;
596      ;
597      ;
598      ;
599      ;
600      ;
601      ;
602      ;
603      ;
604      ;
605      ;
606      ;
607      ;
608      ;
609      ;
610      ;
611      ;
612      ;
613      ;
614      ;
615      ;
616      ;
617      ;
618      ;
619      ;
620      ;
621      ;
622      ;
623      ;
624      ;
625      ;
626      ;
627      ;
628      ;
629      ;
630      ;
631      ;
632      ;
633      ;
634      ;
635      ;
636      ;
637      ;
638      ;
639      ;
640      ;
641      ;
642      ;
643      ;
644      ;
645      ;
646      ;
647      ;
648      ;
649      ;
650      ;
651      ;
652      ;
653      ;
654      ;
655      ;
656      ;
657      ;
658      ;
659      ;
660      ;
661      ;
662      ;
663      ;
664      ;
665      ;
666      ;
667      ;
668      ;
669      ;
670      ;
671      ;
672      ;
673      ;
674      ;
675      ;
676      ;
677      ;
678      ;
679      ;
680      ;
681      ;
682      ;
683      ;
684      ;
685      ;
686      ;
687      ;
688      ;
689      ;
690      ;
691      ;
692      ;
693      ;
694      ;
695      ;
696      ;
697      ;
698      ;
699      ;
700      ;
701      ;
702      ;
703      ;
704      ;
705      ;
706      ;
707      ;
708      ;
709      ;
710      ;
711      ;
712      ;
713      ;
714      ;
715      ;
716      ;
717      ;
718      ;
719      ;
720      ;
721      ;
722      ;
723      ;
724      ;
725      ;
726      ;
727      ;
728      ;
729      ;
730      ;
731      ;
732      ;
733      ;
734      ;
735      ;
736      ;
737      ;
738      ;
739      ;
740      ;
741      ;
742      ;
743      ;
744      ;
745      ;
746      ;
747      ;
748      ;
749      ;
750      ;
751      ;
752      ;
753      ;
754      ;
755      ;
756      ;
757      ;
758      ;
759      ;
760      ;
761      ;
762      ;
763      ;
764      ;
765      ;
766      ;
767      ;
768      ;
769      ;
770      ;
771      ;
772      ;
773      ;
774      ;
775      ;
776      ;
777      ;
778      ;
779      ;
780      ;
781      ;
782      ;
783      ;
784      ;
785      ;
786      ;
787      ;
788      ;
789      ;
790      ;
791      ;
792      ;
793      ;
794      ;
795      ;
796      ;
797      ;
798      ;
799      ;
800      ;
801      ;
802      ;
803      ;
804      ;
805      ;
806      ;
807      ;
808      ;
809      ;
810      ;
811      ;
812      ;
813      ;
814      ;
815      ;
816      ;
817      ;
818      ;
819      ;
820      ;
821      ;
822      ;
823      ;
824      ;
825      ;
826      ;
827      ;
828      ;
829      ;
830      ;
831      ;
832      ;
833      ;
834      ;
835      ;
836      ;
837      ;
838      ;
839      ;
840      ;
841      ;
842      ;
843      ;
844      ;
845      ;
846      ;
847      ;
848      ;
849      ;
850      ;
851      ;
852      ;
853      ;
854      ;
855      ;
856      ;
857      ;
858      ;
859      ;
860      ;
861      ;
862      ;
863      ;
864      ;
865      ;
866      ;
867      ;
868      ;
869      ;
870      ;
871      ;
872      ;
873      ;
874      ;
875      ;
876      ;
877      ;
878      ;
879      ;
880      ;
881      ;
882      ;
883      ;
884      ;
885      ;
886      ;
887      ;
888      ;
889      ;
890      ;
891      ;
892      ;
893      ;
894      ;
895      ;
896      ;
897      ;
898      ;
899      ;
900      ;
901      ;
902      ;
903      ;
904      ;
905      ;
906      ;
907      ;
908      ;
909      ;
910      ;
911      ;
912      ;
913      ;
914      ;
915      ;
916      ;
917      ;
918      ;
919      ;
920      ;
921      ;
922      ;
923      ;
924      ;
925      ;
926      ;
927      ;
928      ;
929      ;
930      ;
931      ;
932      ;
933      ;
934      ;
935      ;
936      ;
937      ;
938      ;
939      ;
940      ;
941      ;
942      ;
943      ;
944      ;
945      ;
946      ;
947      ;
948      ;
949      ;
950      ;
951      ;
952      ;
953      ;
954      ;
955      ;
956      ;
957      ;
958      ;
959      ;
960      ;
961      ;
962      ;
963      ;
964      ;
965      ;
966      ;
967      ;
968      ;
969      ;
970      ;
971      ;
972      ;
973      ;
974      ;
975      ;
976      ;
977      ;
978      ;
979      ;
980      ;
981      ;
982      ;
983      ;
984      ;
985      ;
986      ;
987      ;
988      ;
989      ;
990      ;
991      ;
992      ;
993      ;
994      ;
995      ;
996      ;
997      ;
998      ;
999      ;
1000      ;

```

```

221 0109      WAIT 1
222      ;
223 0110      DEVO = 0                      ; Lamp 11
224 0111      DEV1 = 0                      ; not used
225 0112      DEV2 = 0                      ; Lamp 9
226 0113      DEV3 = 0                      ; Lamp 10
227 0114      DEV4 = 0                      ; Lamp 8
228 0115      DEV5 = 0                      ; not used
229      ;
230 0116      ASSIGN PDEV = AMFARM
231 0117      ASSIGN PSN = FSN0
232 0118      ASSIGN PACK = FLEAKO
233 0119      FOR ILOOP = 0 TO 5              ; For the 6 alarm devices (2 unused)
234 0120      PSN = 0
235      ;
236      ; If alarm device is alarmed and its ALMCT indicates alarms are ack'd:
237      ;
238 0121      IF (PDEV:DEV_TMPO = 2) & AND(PDEV:DEV_ALMCT,32) THEN
239      BEGIN
240 0122          PDEV:DEV_TMPO = 1              ; Put device in ack state
241 0123          LET PSN = 1                    ; Set flag for graphic to ack
242 0124          LET FLGACK = 1                ; WATCH.C spawns ACKFLG.C, which resets flag
243 0125          LET ACK = 1                    ; Set flag for C program to ACK system wide
244 0126          LET PACK = 1
245 0127          ASSIGN PMSG0 = LEAKO
246 0128          STEP PMSG0 BY ILOOP
247 0130          IF ILOOP !=1 THEN LAMP 0,PMSG0,1 ; Flash lamp
248          END
249 0131          STEP PDEV BY 1
250 0132          STEP PSN BY 1
251 0133          STEP PACK BY 1
252 0134          NEXT ILOOP
253      ;
254      ;
255      ;
256 0135      IF(ACK)THEN
257      BEGIN
258 0136          TIMERCLR STMR1
259 0137          TIMERON STMR1
260 0138          FLDALM = 1
261          END
262 0140      IF(FLDALM)THEN IF STMR1 >= 30 THEN ; If field alarm then ack in 30 sec
263      BEGIN
264 0141          TIMEROFF STMR1
265 0142          TIMERCLR STMR1
266 0143          FLDALM = 0
267 0144          OPEN 'HS-AW-SIL'              ; Silence alarm from PCM
268          END
269      ;
270      ; If field alarm ack then set flag and start timer:
271      ;
272 0146      IF(RESET)THEN IF LMPVAR < 2 THEN
273      BEGIN
274 0147          FLDALM = 1
275 0148          TIMERON STMR1

```

```

276 0149      RESET = 0
277          END
278 0151      IF LMPVAR = 2 THEN RESET = 1
279          ;
280          ;
281          ;
282 0152      ASSIGN PDIN = 'YS-AW-101'          ; First DIN in AMFARM
283 0153      ASSIGN PTYPE = IALMO
284 0154      ASSIGN PREV = FXO
285 0155      ASSIGN PSTAT = INTO
286 0156      ASSIGN PSKID = SKIDO
287 0157      ASSIGN PPASS = FPASO
288 0158      CURRENT = 0          ; Used to set lamp status
289 0159      NXT = 0          ; Temporary INT
290 0160      BITVAL = 0
291          ;
292 0161      LOP:  FOR ILOOP = 1 TO 47
293 0162          IF PTYPE THEN          ; If some alarm
294              BEGIN
295 0164          IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3      ;be1          ; If EPW is inha
296 0166          IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5      ; If EPW is ibad
297 0168          IF IAND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4      ; If EPW is dis
298 0171          IF ! PSTAT THEN IF PREV = PDIN:DIN_VAL THEN GOSUB ALM ; If not prev alarmed
299 0172          IF PSTAT THEN
300              BEGIN
301 0173          IF PSTAT < 3 THEN          ;be2
302              BEGIN          ;be3          ; If not inha
303 0174          IF PSTAT = 1 THEN
304              BEGIN          ;be4
305 0175          IF AND(PDIN:DIN_ALMCT,32) THEN          ;be5          ; If alarm is ack
306              BEGIN
307 0176          LET PSTAT = 2
308 0177          FHOOLD = 1
309              END          ;en5
310 0178          IF ACK THEN          ; If a graphic has been ack
311              BEGIN          ;be6 -          go find which graphic
312              CRIT
313 0179          ASSIGN PSN = FSN0
314 0180          STEP PSN BY PSKID          ; Routine to find graphic
315 0181          IF PSN THEN          ; If this is the graphic
316              BEGIN          ;be7 -          then ack it
317 0183          LET PSTAT = 2
318 0185          IF PTYPE = 1 THEN PDIN:DIN_ALMCT = OR(PDIN:DIN_ALMCT,32)
319 0186          FPASS = 1
320 0187          GOSUB ACKK          ; Print signal was ack
321              END          ;en7
322              END          ;en6
323              END          ;en4
324 0189          IF PREV != PDIN:DIN_VAL THEN GOSUB CLR          ; If alarm has cleared
325 0190          ASSIGN PMSG0 = DEVO          ; Find active alarms & graphic
326 0191          STEP PMSG0 BY PSKID
327 0192          IF PPASS THEN
328              BEGIN
329 0194          IF PSTAT = 1 THEN PMSG0 = OR(PMSG0,PSTAT)
330              END

```

```

331 0195          ELSE PMSGO = OR(PMSGO,PSTAT)
332          END                                         ;en3
333 0196          ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8)THEN
334          BEGIN
335 0197          LET PSTAT = 0
336 0198          FHOLD = 1
337          END
338          END                                         ;en2
339          END                                         ;en1
340 0200          ELSE IF PSTAT THEN PSTAT = 0
341 0201          STEP PDIN BY 1
342 0202          STEP PTYPE BY 1
343 0203          STEP PREV BY 1
344 0204          STEP PSTAT BY 1
345 0205          STEP PSKID BY 1
346 0206          STEP PPASS BY 1
347 0207          NEXT ILOOP
348          ;
349          ;
350          ;
351 0208          DEV6 = 0
352 0209          DEV7 = 0
353 0210          DEV8 = 0
354 0211          DEV9 = 0
355 0212          DEV10 = 0
356 0213          DEV11 = 0
357          ;
358          ; -mct1- abandoned code for white alarms tied to pump devices:
359          ;
360          ; AUX: ASSIGN PEPN = 'P-AW-101'
361          ;
362          ;
363          ;
364          ;
365          ;
366          ;
367          ;
368          ;
369          ;
370          ;
371          ;
372          ;
373          ;
374          ;
375          ;
376          ;
377          ;
378          ;
379          ;
380          ;
381          ;
382          ;
383          ;
384          ;
385          ;

```

```

386      ;      END
387      ;      IF IPSTAT THEN
388      ;      BEGIN
389      ;      IF AND(PEPN:DEV_STAT,16)THEN
390      ;      BEGIN
391      ;      BLET PSTAT = 1
392      ;      IF AND(PEPN:DEV_ALMCT,32)THEN PEPN:DEV_ALMCT = XOR(PEPN:DEV_ALMCT,32)
393      ;      ASSIGN PDEV = AUFARM
394      ;      STEP PDEV BY PSKID
395      ;      PDEV:DEV_TMPO = 2
396      ;      GOSUB CYCL
397      ;      END
398      ;      END
399      ;      ELSE IF !AND(PEPN:DEV_STAT,16)THEN
400      ;      BEGIN
401      ;      BLET PSTAT = 0
402      ;      FHOLD = 1
403      ;      IF AND(PEPN:DEV_ALMCT,32)THEN PEPN:DEV_ALMCT = XOR(PEPN:DEV_ALMCT,32)
404      ;      END
405      ;      IF PSTAT THEN
406      ;      BEGIN
407      ;      ASSIGN PMSG0 = DEV6
408      ;      STEP PMSG0 BY PSKID
409      ;      IF PPASS THEN
410      ;      BEGIN
411      ;      IF PSTAT = 1 THEN PMSG0 = OR(PMSG0,PSTAT)
412      ;      END
413      ;      ELSE PMSG0 = OR(PMSG0,PSTAT)
414      ;      END
415      ;      END
416      ;      ELSE IF !AND(PEPN:DEV_ALMCT,16)&!AND(PEPN:DEV_ALMCT,4)&AND(PEPN:DEV_FLG,8)THEN
417      ;      BEGIN
418      ;      BLET PSTAT = 0
419      ;      FHOLD = 1
420      ;      END
421      ;      END
422      ;      STEP PEPN BY 1
423      ;      STEP PTYPE BY 1
424      ;      STEP PREV BY 1
425      ;      STEP PSTAT BY 1
426      ;      STEP PSKID BY 1
427      ;      STEP PPASS BY 1
428      ;      NEXT ILOOP
429      ;      ;mct1-
430      ;
431      0214 S2ND:  ACK = 0
432      0215      SORT = 0
433      ;
434      0216      DEV12 = OR(DEV0,DEV6)
435      0217      DEV13 = OR(DEV1,DEV7)
436      0218      DEV14 = OR(DEV2,DEV8)
437      0219      DEV15 = OR(DEV3,DEV9)
438      0220      DEV16 = OR(DEV4,DEV10)
439      0221      DEV17 = OR(DEV5,DEV11)
440      ;

```



```

441 0222      ASSIGN PMSG0 = DEV12
442 0223      ASSIGN PMSG1 = FARM
443 0224      FOR LLOOP = 1 TO 5
444 0225          IF PMSG0 THEN
445              BEGIN
446 0227          IF AND(PMSG0,1) THEN PMSG1 = 2
447 0228          ELSE PMSG1 = 1
448              END
449 0229          ELSE PMSG1 = 0
450 0230          STEP PMSG0 BY 1
451 0231          STEP PMSG1 BY 1
452 0232          NEXT LLOOP
453      ;
454 0233      IF FHOLD THEN
455          BEGIN
456 0234          LAMP 0,11,FARM
457 0235          LAMP 0,10,HVAC
458 0236          LAMP 0,9,VLVPIT
459 0237          LAMP 0,8,BLD271
460 0238          FHOLD = 0
461 0239          TIMERCLR STM2
462 0240          TIMERON STM2
463          END
464 0241      MARK STEP "BOTTOM",0,0
465 0242      NOCRIT
466      ;
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473 0244      IF !AND(AUTO,64) THEN FSL7 = 1 ; If manual, set manual flag
474 0245      IF AND(AUTO,64) & AND(FSL7,1) THEN ; For manual-auto transition
475          BEGIN
476 0246          FSL7 = 0
477 0247          CALOUT = 0.0
478          END
479      ;
480      ;
481      ;
482 0248      GOTO AGAIN
483      ;
484      ;
485      ;
486      ;
487      ;
488 0249      CLR: LET PSTAT = 0
489 0251      IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
490 0252      FHOLD = 1
491 0253      RETURN
492      ;
493 0254      ALM: LET PSTAT = 1 ; Print message, flash lamp, horn on, scroll message at bottom of graphic
494 0255      CRIT ; Critical
495 0257      IF AND(PTYPE,3) THEN FALMON = 1 ; Audible alarm on

```

```

496 0258      IF AND(PDIN:DIN_ALMCT,32)THEN      ; Clear the scan block
497 0259      PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
498 0260      ASSIGN PMSG0 = LEAKO
499 0261      STEP PMSG0 BY PSKID
500 0262      LAMP 0,PMSG0,2      ; Lamp flash
501 0263      ASSIGN PDEV = AMFARM      ; Find DEV and turn it on
502 0264      STEP PDEV BY PSKID
503 0265      PDEV:DEV_TMPO = 2
504      ;
505      ;      Scrolling alarm summary region in graphic (message, time, date, alarm type):
506      ;
507 0266  CYCL:  LSG4 = LSG3      ; LSG = message
508 0267      LSG3 = LSG2
509 0268      LSG2 = LSG1
510 0269      LSG1 = LSG0
511 0270      COLOR4 = COLOR3      ; COLOR = type of alarm
512 0271      COLOR3 = COLOR2
513 0272      COLOR2 = COLOR1
514 0273      COLOR1 = COLOR0
515 0274      EPMS4 = EPMS3      ; EPN
516 0275      EPMS3 = EPMS2
517 0276      EPMS2 = EPMS1
518 0277      EPMS1 = STRNG1
519 0278      GRAFX4 = GRAFX3      ; GRAFX = graphic number
520 0279      GRAFX3 = GRAFX2
521 0280      GRAFX2 = GRAFX1
522 0281      GRAFX1 = GRAFX0
523 0282      SDATE4 = SDATE3      ; SDATE = time and date
524 0283      SDATE3 = SDATE2
525 0284      SDATE2 = SDATE1
526 0285      SDATE1 = SDATE0
527      ;
528      ;      ALARM TYPER STRINGS:
529      ;
530 0286  ACKK:  FHOLD = 1
531 0287      ON ILOOP GOTO S1,S2,S3,S4,S5,S6,S7,S8,S9,S10
532 0288      ON ILOOP - 10 GOTO S11,S12,S13,S14,S15,S16,S17,S18,S19,S20
533 0289      ON ILOOP - 20 GOTO S21,S22,S23,S24,S25,S26,S27,S28,S29,S30
534 0290      ON ILOOP - 30 GOTO S31,S32,S33,S34,S35,S36,S37,S38,S39,S40
535 0291      ON ILOOP - 40 GOTO S41,S42,S43,S44,S45,S46,S47,S48,S49,S50
536 0292      ON ILOOP - 50 GOTO S51,S52,S53,S54,S55
537 0293      GOTO AMSG
538 0294  S1:   STRNG0 = "PUMP AW-101 OFF      "
539 0295      STRNG1 = "YS-AW-101"
540 0296      GOTO AMSG
541 0297  S2:   STRNG0 = "AW-101 ANN RD HI/FAIL "
542 0298      STRNG1 = "RSH-AW101"
543 0299      GOTO AMSG
544 0300  S3:   STRNG0 = "AW-01C RAD HI/FAIL    "
545 0301      STRNG1 = "RSH-AW01C"
546 0302      GOTO AMSG
547 0303  S4:   STRNG0 = "AW-102 ANN RD HI/FAIL "
548 0304      STRNG1 = "RSH-AW102"
549 0305      GOTO AMSG
550 0306  S5:   STRNG0 = "AW-02C RAD HI/FAIL    "

```

551	0307		STRING1 = "RSH-AW02C"
552	0308		GOTO AMSG
553	0309	S6:	STRING0 = "PUMP AU-103 OFF "
554	0310		STRING1 = "YS-AW-103"
555	0311		GOTO AMSG
556	0312	S7:	STRING0 = "AW-103 ANN RD HI/FAIL "
557	0313		STRING1 = "RSH-AW103"
558	0314		GOTO AMSG
559	0315	S8:	STRING0 = "AW-03C RADN HIGH/FAIL "
560	0316		STRING1 = "RSH-AW03C"
561	0317		GOTO AMSG
562	0318	S9:	STRING0 = "PUMP AU-104 OFF "
563	0319		STRING1 = "YS-AW-104"
564	0320		GOTO AMSG
565	0321	S10:	STRING0 = "AW-104 ANN RD HI/FAIL "
566	0322		STRING1 = "RSH-AW104"
567	0323		GOTO AMSG
568	0324	S11:	STRING0 = "AW-04C RADN HIGH/FAIL "
569	0325		STRING1 = "RSH-AW04C"
570	0326		GOTO AMSG
571	0327	S12:	STRING0 = "PUMP AU-105 OFF "
572	0328		STRING1 = "YS-AW-105"
573	0329		GOTO AMSG
574	0330	S13:	STRING0 = "AW-105 ANN RD HI/FAIL "
575	0331		STRING1 = "RSH-AW105"
576	0332		GOTO AMSG
577	0333	S14:	STRING0 = "AW-05C RADN HIGH/FAIL "
578	0334		STRING1 = "RSH-AW05C"
579	0335		GOTO AMSG
580	0336	S15:	STRING0 = "PUMP AU-106 OFF "
581	0337		STRING1 = "YS-AW-106"
582	0338		GOTO AMSG
583	0339	S16:	STRING0 = "AW-106 ANN RD HI/FAIL "
584	0340		STRING1 = "RSH-AW106"
585	0341		GOTO AMSG
586	0342	S17:	STRING0 = "AW-06C RADN HIGH/FAIL "
587	0343		STRING1 = "RSH-AW06C"
588	0344		GOTO AMSG
589	0345	S18:	STRING0 = "AW TL-21 LINE LEAK "
590	0346		STRING1 = "LDS-TL021"
591	0347		GOTO AMSG
592	0348	S19:	STRING0 = "AW CIRCUIT TROBL/LEAK "
593	0349		STRING1 = "LDS-TL223"
594	0350		GOTO AMSG
595	0351	S20:	STRING0 = "AW DRAIN ENCASE LEAK "
596	0352		STRING1 = "LDS-AW-DR"
597	0353		GOTO AMSG
598	0354	S21:	STRING0 = "AW PROCESS PITTS LEAK "
599	0355		STRING1 = "LDS-AW-PP"
600	0356		GOTO AMSG
601	0357	S22:	STRING0 = "AW SLURRY ENCASE LEAK "
602	0358		STRING1 = "LDS-AW-SL"
603	0359		GOTO AMSG
604	0360	S23:	STRING0 = "AW SUPRN8 ENCASE LEAK "
605	0361		STRING1 = "LDS-AW-SN"

```

606 0362          GOTO AMSG
607 0363 S24:    STRNG0 = "AW SN-269 LINE LEAK  "
608 0364          STRNG1 = "LDS-SN269"
609 0365          GOTO AMSG
610 0366 S25:    STRNG0 = "AW SN-270 LINE LEAK  "
611 0367          STRNG1 = "LDS-SN270"
612 0368          GOTO AMSG
613 0369 S26:    STRNG0 = "AW CLEAKOUT BOX LEAK  "
614 0370          STRNG1 = "LDS-COBAM"
615 0371          GOTO AMSG
616 0372 S27:    STRNG0 = "AW SN FLSH LN PRES HI  "
617 0373          STRNG1 = "PSH-AWSN1"
618 0374          GOTO AMSG
619 0375 S28:    STRNG0 = "AW FEEDLINE PRESSUR HI  "
620 0376          STRNG1 = "PSH-AWSN2"
621 0377          GOTO AMSG
622 0378 S29:    STRNG0 = "AW FARM GENERAL ALARM  "
623 0379          STRNG1 = "OS-AM-271"
624 0380          GOTO AMSG
625 0381 S30:    STRNG0 = "AW SERPIT RAD HI/FAIL  "
626 0382          STRNG1 = "RSH-SP-AW"
627 0383          GOTO AMSG
628 0384 S31:    STRNG0 = "AW SERVICE PIT LEAK  "
629 0385          STRNG1 = "LDS-SP-AW"
630 0386          GOTO AMSG
631 0387 S32:    STRNG0 = "AW-FRM TNK EXH RAD HI  "
632 0388          STRNG1 = "RSH-AW-K1"
633 0389          GOTO AMSG
634 0390 S33:    STRNG0 = "K1 SK MN SMPL PMP OFF  "
635 0391          STRNG1 = "PSH-AWK11"
636 0392          GOTO AMSG
637 0393 S34:    STRNG0 = "K1 REC SMPL PUMP OFF  "
638 0394          STRNG1 = "PSH-AWK12"
639 0395          GOTO AMSG
640 0396 S35:    STRNG0 = "K2 STK MN SPL PMP OFF  "
641 0397          STRNG1 = "PSH-AWK21"
642 0398          GOTO AMSG
643 0399 S36:    STRNG0 = "K2 REC SMPL PUMP OFF  "
644 0400          STRNG1 = "PSH-AWK22"
645 0401          GOTO AMSG
646 0402 S37:    STRNG0 = "AW TANK EXH FANS OFF  "
647 0403          STRNG1 = "QXS-AWK15"
648 0404          GOTO AMSG
649 0405 S38:    STRNG0 = "AWN EX RD/FAN HI/FAIL  "
650 0406          STRNG1 = "RSH-AW-K2"
651 0407          GOTO AMSG
652 0408 S39:    STRNG0 = "AW TNK EX RADMON FAIL  "
653 0409          STRNG1 = "QXSAMK1A2"
654 0410          GOTO AMSG
655 0411 S40:    STRNG0 = "241-AW RADN HI/FAIL  "
656 0412          STRNG1 = "RSH-AW-12"
657 0413          GOTO AMSG
658 0414 S41:    STRNG0 = "271-AW RADN HI/FAIL  "
659 0415          STRNG1 = "RSHAW2711"
660 0416          GOTO AMSG

```

```

661 0417 S42:  STRNG0 = "271-AW RADN HI/FAIL  "
662 0418      STRNG1 = "RSHAW2712"
663 0419      GOTO AMSG
664 0420 S43:  STRNG0 = "AW AREA RADN BYPASS  "
665 0421      STRNG1 = "YS-AW-ARM"
666 0422      GOTO AMSG
667 0423 S44:  STRNG0 = "271-AW BLD POWER LOSS "
668 0424      STRNG1 = "JXS-AW271"
669 0425      GOTO AMSG
670 0426 S45:  STRNG0 = "EPN DELETED          "
671 0427      STRNG1 = "NO EPN"
672 0428      GOTO AMSG
673 0429 S46:  STRNG0 = "AW-FARM MASTER SHTOWN "
674 0430      STRNG1 = "YS-AW-MSD"
675 0431      GOTO AMSG
676 0432 S47:  STRNG0 = "AW HEAT TRACE OVRTMP  "
677 0433      STRNG1 = "TXS-AW-2"
678 0434      GOTO AMSG
679      ;
680      ; -mct1- abandoned code for white alarms tied to pump devices:
681      ;
682      S48:  ; STRNG0 = "PUMP AW-101 SHUTDOWN "
683      ; STRNG1 = "P-AW-101"
684      ; GOTO AMSG
685      S49:  ; STRNG0 = "PUMP AW-103 SHUTDOWN "
686      ; STRNG1 = "P-AW-103"
687      ; GOTO AMSG
688      S50:  ; STRNG0 = "PLMP AW-104 SHUTDOWN "
689      ; STRNG1 = "P-AW-104"
690      ; GOTO AMSG
691      S51:  ; STRNG0 = "PUMP AW-105 SHUTDOWN "
692      ; STRNG1 = "P-AW-105"
693      ; GOTO AMSG
694      S52:  ; STRNG0 = "PUMP AW-106 SHUTDOWN "
695      ; STRNG1 = "P-AW-106"
696      ; GOTO AMSG
697      ; -mct1-
698      ;
699 0435 S53:  STRNG0 = "AWBATCH PROGRAM START "
700 0436      GOTO AMSG
701 0437 S54:  STRNG0 = "AWBATCH IN WRONG UNIT "
702 0438      GOTO AMSG
703 0439 S55:  STRNG0 = "AWBATCH TERMINATING  "
704 0440      GOTO AMSG
705      ;
706 0441 AMSG:  IF (FPASS) THEN
707      BEGIN
708      PRINT ADI, "%p ALARM ACKNOWLEDGE", STRNG1
709 0443      FPASS = 0
710 0444      GOTO NXT
711      END
712 0446      IF PTYP = 4 THEN COLOR0 = 3
713 0447      ELSE COLOR0 = PTYP
714 0448      EPWS0 = STRNG1
715 0449      IN0 = TIMSTR(SSNG0)

```

```

716 0451 IF INO != 8 THEN SSNGO = "TIMEFAIL"
717 0452 INO = APPEND(LSGO,STRNGO,SSNGO)
718 0453 IF INO < 0 THEN LSGO = "APPEND FAILED!!!"
719 0454 INO = DATSTR(DATED)
720 0455 IF INO != 9 THEN SDATED = "DATE FAIL"
721 0456 IF PSKID = 1 THEN GRAFXO = 303 ; SERVICE PIT
722 0457 IF PSKID = 2 THEN GRAFXO = 303 ; VALVE PIT
723 0458 IF PSKID = 3 THEN GRAFXO = 303 ; HVAC
724 0459 IF PSKID = 4 THEN GRAFXO = 302 ; BLD-271
725 0460 IF PSKID = 5 THEN GRAFXO = 301 ; PUMP
726 0461 IF PSKID = 0 THEN GRAFXO = 300 ; FARM
727 0462 IF IFTYPE = 1 THEN PRINT ADI,"WHITE ALARM %p GRAFICS %i",EPNSO,STRNGO,GRAFXO
728 0463 IF FSTART = 1 THEN
729 0464 BEGIN
730 0465 CGORO = 0
731 0466 GRAFXO = 0
732 0467 STRNG1 = ""
733 0468 FSTART = 0
734 0469 PRINT ADI,"%p",STRNGO
735 0470 END
736 0471 RETURN
737 0472
738 0473 ERRSUB: IF(ERRNUM() = -187)THEN ERSRET
739 0474 ELSE
740 0475 BEGIN
741 0476 ERRN = ERRNUM()
742 0477 ERRL = ERRLIN()
743 0478 PRINT ADI,"ERROR # %i AT LINE %i, ILOOP = %i",errn,errl,ILOOP
744 0479 ERSCLR
745 0480 END
746 0481 GOTO AGAIN
747 0482
748 0483 ; TERM: FADORT = 0
749 0484 STRNGO = "AMACK TERMINATED "
750 0485 ILOOP = 55
751 0486 FSTART = 1
752 0487 TIMEROFF STM#1
753 0488 TIMERCLE STM#1
754 0489 FLDAIM = 0
755 0490 OPEN 'HS-AU-SIL'
756 0491 GOSUB CYCL
757 0492 END

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 9408 Bytes, 588 Pages
116 Forward Reference(s)

XYZBATCH

```

1      :   X Y Z B A T C H . S E Q                               VERSION 2
2      :
3      :   Function:      Sequence program for Lamp function,
4      :                  message display on graphics, and
5      :                  printer messages.
6      :
7      :   Written By    : R. N. Kersey
8      :   Date         : 05-31-92
9      :   Facility     : 242-A Evaporator Bldg.
10     :   Machine      : VAX 4104-A
11     :   Op. System   : VAX/VMS 5.5-2AH
12     :   D/3 Version  : 6.3-2 BASELINE
13     :   Compiler     : SIC
14     :   Build command: @[320,20]XYZBATCH
15
16     UNIT = 4, SKDKEY = 12, 13, 14, 15
17
18     Assoc. UPF: [300,1]_TNKFRM.____ TNKFRM.LST
19     Assoc. RPF: [300,1]_FARM.____ FARM.LST
20     SDB: [300,1]PCM1ASDB.____ PCM1ASDB.CMD
21
22     POINTER      SDB VAR.      FUNCTION
23     -----
24     PSTAT      INTn      Current EPN alarm state is an integer tied to
25                       graphic flashing string of value:
26     .      (n=0-100)      0 = previously not alarmed
27     .      .      1 = previously alarmed, not ack
28     .      .      2 = previously alarmed, is ack
29     .      .      3 = alarm is inhibited
30     .      .      4 = input is off scan
31     .      .      5 = input is bad
32
33     PDIN      'EPN'      ---
34     PTYP      IALMn      alarm type 0 = none, 1 = white, 2 = yellow,
35                       3 = yellow and yellow, 4 = red,
36                       6 = yellow and red
37
38     PREV      FXn      Reverse acting (not fail safe) flag (1=set)
39     PSKID      SKIDn      Tracks which graphic has an alarm
40     PPASS      FPASn      If set (1), turn Lamp off when alarm is ACKnowledged
41     PSN      FSAn      Flag for the graphic to ACKnowledge
42     ---      FHOLD      Flag to update the lamps
43     ---      FNING      Flag for DOG1A.C micro select
44     ---      ACK      Flag for C program to ACK system wide
45
46     MODIFICATION LOG:
47
48     11-JUN-93      Added comments. CCN# 93JUN01-01. M. Teats
49
50     -mct1+ 25-JUN-93      Set QSCONSOLE to no alarms, since the
51                       Gamewell is not connected to the MCS. M. Teats
52                       CCN# 93JUN25-01.
53
54     -mct2+ 19-JUL-93      Delete QSCONSOLE per ECN 190990, since the
55                       Gamewell is not connected to the MCS. CCN
56                       93JUL15-01. Baselined for D/3 version 3.3.

```



```

56      ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
57      ;
58      ;      system. Rebaselined document. Removed
59      ;      obsolete Select() statement.
60      ;      END MOD LOG
61      ;
62      UNIT      XYZFRM
63      COMMON A244      ; Include PSH-SN215, PSH-SN216
64      PUMP == FLSPIT
65      0000      GETUNIT TRIP      ; Which sequence unit is program running in
66      0001      IF TRIP != 4 THEN      ; Abort if running in wrong seq. unit
67      BEGIN
68      0002      FSTART = 1
69      0003      ILOOP = 38
70      0004      GOSUB CYCL
71      0005      FSTART = 1
72      0006      PRINT AD1," >>>> ATTEMPT TO RUN XYZBAT IN WRONG UNIT"
73      0007      GOTO TERM
74      END
75      ;
76      ;      IF !SELECT() THEN GOTO AGAIN
77      ;
78      ;      B E G I N      I N I T I A L I Z I N G :
79      0008      NOP
80      0009      INIT:      LET FABORT = 0      ; Abort program if FABORT = 1
81      0010      LET FSTART = 0      ; Program start flag
82      0011      LET FLDALM = 0
83      0012      LET FSL6 = 0      ; QS-204AR1 flag (mct)
84      0013      LET FSL7 = 0      ; QS-204AR2 flag (mct)
85      0014      LET ERRN = 0      ; Set error number to zero
86      0015      LET ERRL = 0      ; Set error line to zero
87      0016      LAMP 0,15,0      ; Turn off the lamps
88      0017      LAMP 0,14,0
89      0018      LAMP 0,13,0
90      0019      LAMP 0,12,0
91      0020      ASSIGN PDEV = AFARM      ; First of 5 alarm ack devices
92      0021      FOR ILOOP = 1 TO 5
93      0022      PDEV:DEV_TMP0 = 1      ; Set all 5 devices to ack state
94      0023      IF AND(PDEV:DEV_ALMCT,16) THEN      ; If alarm device is inha, remove the inha
95      0024      PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
96      0025      IF AND(PDEV:DEV_ALMCT,32) THEN      ; If alarm device is ack, clear it
97      0026      PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
98      0027      STEP PDEV BY 1
99      0028      NEXT ILOOP
100     ;
101     ;      First, set all flags and integers:
102     ;
103     0029      ASSIGN PDIN = 'PSH-SL100'      ; Assign pointer to first DIN
104     0030      ASSIGN PSTAT = INT4
105     0031      ASSIGN PTYPE = IALN4
106     0032      ASSIGN PREV = FX4
107     0033      ASSIGN PSKID = SKID4
108     0034      ASSIGN PPASS = FPAS4
109     ;
110     0035      FARM = 0      ; Set up values for different graphics

```

```

111 0036      SERPIT = 1                                ; SKID of each EPN indicates graphic
112 0037      VLVPI = 2
113 0038      HVAC = 3
114 0039      BLD271 = 4
115      ;      PUMP = 5
116      ;
117 0040      LEAK0 = 15                                ; XYZ FARM lamp
118 0041      LEAK1 = 14                                ; A FARM lamp
119 0042      LEAK2 = 13                                ; AY FARM lamp
120 0043      LEAK3 = 12                                ; AZ FARM lamp
121      ;
122 0044      FOR ILOOP = 4 TO 29                        ; For D1Ns
123 0045      LET PPASS = 0
124 0046      LET PREV = 0
125 0047      LET PTYPE = 2
126      ;      BLET PSTAT = 0
127 0048      IF ILOOP = 29 THEN                        ; -mct2+
128      BEGIN
129 0049      LET IALM29 = 0                            ; -mct1+
130 0050      GOTO SKP
131      END
132 0051      IF AND(PDIN:DIN_DFLG,3) THEN                ; Check for alarms (must be red or yellow)
133      BEGIN
134 0052      LET PTYPE = 2                                ; Default to yellow alarm
135 0054      IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1    ; Reverse acting
136 0056      IF AND(PDIN:DIN_DFLG,4) THEN LET PTYPE = 4    ; Set to red critical
137      END
138 0058      IF ILOOP = 0 ; ILOOP = 1 THEN LET PSKID = VLVPI
139 0060      IF ILOOP = 2 ; ILOOP = 3 THEN LET PSKID = HVAC
140 0062      IF ILOOP > 3 & ILOOP < 10 THEN LET PSKID = SERPIT
141 0064      IF ILOOP = 10 ; ILOOP = 11 THEN LET PSKID = HVAC
142 0066      IF ILOOP > 11 & ILOOP < 16 THEN LET PSKID = VLVPI
143 0068      IF ILOOP = 16 ; ILOOP = 17 THEN LET PSKID = HVAC
144 0070      IF ILOOP = 18 THEN LET PSKID = VLVPI
145 0072      IF ILOOP > 18 THEN LET PSKID = FARM
146 0074      IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
147 0075      SKP: STEP PSTAT BY 1
148 0076      STEP PDIN BY 1
149 0077      STEP PTYPE BY 1
150 0078      STEP PREV BY 1
151 0079      STEP PSKID BY 1
152 0080      STEP PPASS BY 1
153 0081      NEXT ILOOP
154      ;
155      ;      For White alarms on pump devices:
156      ;
157 0082      FOR ILOOP = 30 TO 33
158      ;      BLET PSTAT = 0
159 0083      LET PTYPE = 1
160 0084      LET PREV = 1
161 0086      IF ILOOP = 30 ; ILOOP = 31 THEN LET PSKID = VLVPI
162 0088      IF ILOOP = 32 ; ILOOP = 33 THEN LET PSKID = HVAC
163 0089      LET PPASS = 1
164 0090      STEP PSTAT BY 1
165 0091      STEP PTYPE BY 1

```

```

166 0092      STEP PREV BY 1
167 0093      STEP PSKID BY 1
168 0094      STEP PPASS BY 1
169 0095      NEXT ILOOP
170          ;
171 0096      FX0 = 0
172 0097      FX1 = 0
173 0098      FX2 = 0
174 0099      FX3 = 0
175          ;
176          ; Set the white alarms:
177          ;
178 0100      IALM0 = 0
179 0101      IALM1 = 0
180 0102      IALM2 = 0
181 0103      IALM3 = 0
182 0104      IALM20 = 1
183 0105      IALM21 = 1
184 0106      IALM30 = 1
185 0107      IALM31 = 1
186 0108      IALM32 = 1
187 0109      IALM33 = 1
188          ;
189 0110      FPAS0 = 1
190 0111      FPAS1 = 1
191 0112      FPAS2 = 1
192 0113      FPAS3 = 1
193          ;
194          ; For PSH-SN215 (#34) and PSH-SN216 (#35) signals in PCMOA:
195          ;
196 0114      FX34 = 0
197 0115      FX35 = 0
198          ;
199          ; INT34 = 0
200 0116      INT35 = 0
201 0117      IALM34 = 2
202 0118      IALM35 = 2
203 0119      SKID34 = SERPIT
204 0120      SKID35 = SERPIT
205 0121      FPAS34 = 0
206 0121      FPAS35 = 0
207          ;
208          ;
209          ; END INITIALIZING
210          ;
211 0122      ILOOP = 37
212 0123      FSTART = 1
213 0124      GOSUB CYCL
214 0125      TIMERON STM2
215 0126      TIMERCLR STM2
216 0127      ON ERROR GOSUB ERRSUB
217          ;
218          ; BEGIN PROGRAM LOOP:
219          ;
220          ; PLEASE NOTE: ALARM ACKNOWLEDGE DEVICES USED IN

```

```

221      ;           GRAPHICS 320 THROUGH 343 START WITH 'AFARM' AND GO
222      ;           TO DEVICE 'XYZ'. GRAPHICS GO FROM XYZFRM TO AZFRM.
223      ;           THIS IS TRANSPARENT TO THE OPERATOR.
224
225 0128 AGAIN: MARK STEP "AGAIN",0,0
226      ;           IF ISELECT() THEN FNING = 0
227      ;           ELSE FNING = 1 ; Flag 480 determines if selected micro for DOG1A.C
228      ;           WAITUNTIL SELECT() ; FABORT ; Wait until micro selected or program aborted
229      ;           IF (FABORT) THEN GOTO TERM ; Terminate program if abort flag is set
230      ;           IF FNING = 0 THEN FHOLD = 1
231 0130      IF STM2 > 30 THEN FHOLD = 1
232 0131      WAIT 1
233      ;
234 0132      DEV0 = 0
235 0133      DEV1 = 0
236 0134      DEV2 = 0
237 0135      DEV3 = 0
238 0136      DEV4 = 0
239 0137      DEV5 = 0
240      ;
241      ; If alarm device is alarmed and its ALMCT indicates alarms are ack'd:
242      ;
243 0138      ASSIGN PDEV = AFARM ; DEV ACCIDENTALLY OFFSET BY 1 (RNK)
244 0139      ASSIGN PSN = FSN0
245 0140      ASSIGN PACK = FLEAK0
246 0141      FOR ILOOP = 0 TO 3
247 0142      PSN = 0
248 0143      IF (PDEV:DEV_TMP0 = 2) & AND(PDEV:DEV_ALMCT,32) THEN
249          BEGIN
250 0144      PDEV:DEV_TMP0 = 1 ; Put device in ack state
251 0145      LET PSN = 1 ; Set flag for graphic to ack
252 0146      LET FLGACK = 1 ; WATCH.C spawns ACKFLG.C, which resets flag
253 0147      LET ACK = 1 ; Set flag for C program to ACK system wide
254 0148      LET PACK = 1 ; ACKFLG.C to acknowledge this graphic
255 0149      ASSIGN PMSG0 = LEAK0 ; Make the appropriate lamp steady:
256 0150      STEP PMSG0 BY ILOOP
257 0152      IF ILOOP != 1 THEN LAMP 0,PMSG0,1 ; Flash lamp
258      END
259 0153      ELSE PSN = 0
260 0154      STEP PSN BY 1
261 0155      STEP PDEV BY 1
262 0156      STEP PACK BY 1
263 0157      NEXT ILOOP
264      ;
265      ; Field Alarm Silence feature:
266      ;
267 0158      IF(ACK)THEN
268          BEGIN
269 0159      TIMECLR STM1
270 0160      TIMEON STM1
271 0161      FLDALM = 1
272      END
273 0163      IF(FLDALM)THEN IF STM1 >= 30 THEN ; If field alarm then ack in 30 sec
274      BEGIN
275 0164      TIMEOFF STM1

```

```

276 0165      TIMERCLR STMR1
277 0166      FLDALM = 0
278 0167      OPEN 'HSAXYZSIL'          ; Silence alarm from PCM
279          END
280          ;
281          ; If field alarm ack then set flag and start timer:
282          ;
283 0169      IF(RESET)THEN IF LMPVAR < 2 THEN
284          BEGIN
285 0170          FLDALM = 1
286 0171          TIMERON STMR1
287 0172          TIMERCLR STMR1
288 0173          RESET = 0
289          END
290 0175      IF LMPVAR = 2 THEN RESET = 1
291          ;
292          ;
293          ;
294 0176      ASSIGN PDIN = 'PSH-SL100'    ; Assign pointer to first DIN
295 0177      ASSIGN PTYPE = IALM4
296 0178      ASSIGN PREV = FX4
297 0179      ASSIGN PSTAT = INT4
298 0180      ASSIGN PSKID = SKID4
299 0181      ASSIGN PPASS = FPAS4
300 0182      CURENT = 0                  ; Used to set lamp status
301 0183      NXT = 0                    ; Temporary INT
302 0184      BITVAL = 0
303          ;
304 0185      LOP:  FOR ILOOP = 5 TO 30
305 0187          IF ILOOP = 30 THEN GOTO SKP1          ; -mct2+
306 0188          IF PTYPE THEN                      ; If some alarm
307              BEGIN
308 0190              IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3      ; be1
309 0192              IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5      ; If EPN is inha
310 0194              IF !AND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4      ; If EPN is ibad
311 0197              IF 1 PSTAT THEN IF PREV = PDIN:DIN_VAL THEN GOSUB ALM ; If EPN is dis
312 0198              IF PSTAT THEN                                ; If not prev alarmed
313                  BEGIN
314 0199                  IF PSTAT < 3 THEN                                ; be2
315                      BEGIN
316 0200                      IF PSTAT = 1 THEN                                ; be3
317                          BEGIN
318 0201                          IF AND(PDIN:DIN_ALMCT,32) THEN                                ; be4
319                              BEGIN
320 0202                              LET PSTAT = 2
321 0203                              F HOLD = 1
322                              END
323 0204                              IF ACK THEN                                ; be5
324                                  BEGIN
325 0205                                  CRIT
326 0206                                  ASSIGN PSN = FSWO
327 0207                                  STEP PSN BY PSKID
328 0208                                  IF PSN THEN                                ; If alarm is ack
329                                      BEGIN
330 0209                                      LET PSTAT = 2

```

```

331 0211          IF PTYPE = 1 THEN PDIN:DIN_ALMCT = OR(PDIN:DIN_ALMCT,32)
332 0212          FPASS = 1
333 0213          GOSUB ACKK
334          END                                ;en7
335          END                                ;en6
336          END                                ;en4
337 0215          IF PREV != PDIN:DIN_VAL THEN GOSUB CLR          ; If alarm has cleared
338 0216          ASSIGN PMSG0 = DEVO          ; Find active alarms & graphics:
339 0217          STEP PMSG0 BY PSKID
340 0218          IF PPASS THEN
341              BEGIN
342 0219              FHOLD = 1
343 0221              IF PSTAT = 1 THEN PMSG0 = OR(PMSG0,PSTAT)
344              END
345 0222              ELSE PMSG0 = OR(PMSG0,PSTAT)
346              END                                ;en3
347 0224          ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8)THEN PSTAT = 0
348              END                                ;en2
349              END                                ;en1
350          ELSE IF PSTAT THEN PSTAT = 0
351 0227  SKP1:    STEP PDIN BY 1
352 0228          STEP PTYPE BY 1
353 0229          STEP PREV BY 1
354 0230          STEP PSTAT BY 1
355 0231          STEP PSKID BY 1
356 0232          STEP PPASS BY 1
357 0233          NEXT ILOOP
358          ;
359          ;    END DINs
360          ;
361 0234          DEV6 = 0
362 0235          DEV7 = 0
363 0236          DEV8 = 0
364 0237          DEV9 = 0
365 0238          DEV10 = 0
366 0239          DEV11 = 0
367          ;
368          ;    BEGIN WHITE ALARMS FOR PUMP DEVICES:
369          ;
370 0240  AUX:    ASSIGN PEPN = 'P-AY-101'
371 0241          FOR ILOOP = 31 TO 34
372 0242          IF PTYPE THEN
373              BEGIN
374 0244              IF AND(PEPN:DEV_ALMCT,16) THEN LET PSTAT = 3      ; If EPN is inha
375 0246              IF AND(PEPN:DEV_ALMCT,4) THEN LET PSTAT = 5      ; If EPN is ibad
376 0248              IF !AND(PEPN:DEV_FLG,8) THEN LET PSTAT = 4      ; If EPN is dis
377 0249              IF PSTAT < 3 THEN                                ; If not inha
378                  BEGIN
379 0250                  IF (PSTAT = 1) & AND(PEPN:DEV_ALMCT,32) THEN      ; If prev unack alarm, and is now ack
380                      BEGIN
381 0251                      LET PSTAT = 2
382 0252                      FHOLD = 1
383                      END
384 0253                  IF ACK THEN
385                      BEGIN

```

```

386 0254          ASSIGN PSN = FSNO
387 0255          STEP PSN BY PSKID
388 0256          IF PSN & PSTAT = 1 THEN
389              BEGIN
390 0257              LET PSTAT = 2
391 0259              IF PTYPE = 1 THEN PEPN:DEV_ALMCT = OR(PEPN:DEV_ALMCT,32)
392 0260              PPASS = 1
393 0261              GOSUB ACKK
394              END
395          END
396 0262          IF !PSTAT THEN                                ; If prev not alarmed
397              BEGIN
398 0263              IF AND(PEPN:DEV_STAT,16)THEN                ; If shutdown
399                  BEGIN
400 0264                  LET PSTAT = 1
401 0266                  IF AND(PEPN:DEV_ALMCT,32)THEN PEPN:DEV_ALMCT = XOR(PEPN:DEV_ALMCT,32)
402 0267                  ASSIGN PDEV = AFARM
403 0268                  STEP PDEV BY PSKID
404 0269                  PDEV:DEV_TMPO = 2
405 0270                  GOSUB CYCL
406                  END
407              END
408 0271          ELSE IF !AND(PEPN:DEV_STAT,16)THEN
409              BEGIN
410 0272              LET PSTAT = 0
411 0273              FHOLD = 1
412 0275              IF AND(PEPN:DEV_ALMCT,32)THEN PEPN:DEV_ALMCT = XOR(PEPN:DEV_ALMCT,32)
413              END
414 0276          IF PSTAT THEN
415              BEGIN
416 0277              ASSIGN PMSGO = DEV6
417 0278              STEP PMSGO BY PSKID
418 0279              IF PPASS THEN
419                  BEGIN
420 0281                  IF PSTAT = 1 THEN PMSGO = OR(PMSGO,PSTAT)
421                  END
422 0282              ELSE PMSGO = OR(PMSGO,PSTAT)
423              END
424          END
425 0284          ELSE IF !AND(PEPN:DEV_ALMCT,16)&!AND(PEPN:DEV_ALMCT,4)&AND(PEPN:DEV_FLG,8)THEN PSTAT = 0
426              END
427 0285          STEP PEPN BY 1
428 0286          STEP PTYPE BY 1
429 0287          STEP PREV BY 1
430 0288          STEP PSTAT BY 1
431 0289          STEP PSKID BY 1
432 0290          STEP PPASS BY 1
433 0291          NEXT ILOOP
434          ;
435          ;
436          ;
437          ;
438          ;
439 0292          PCMOA: ILOOP = 35
440 0293          IF IALM34 THEN

```

```

441          BEGIN
442 0295      IF AND('PSH-SN215':DIN_ALMCT,16) THEN LET INT34 = 3 ;be1
443 0297      IF AND('PSH-SN215':DIN_ALMCT,4) THEN LET INT34 = 5 ; If EPN is inha
444 0299      IF !AND('PSH-SN215':DIN_FLG,8) THEN LET INT34 = 4 ; If EPN is ibad
445 0302      IF ! INT34 THEN IF FX34 = 'PSH-SN215':DIN_VAL THEN GOSUB ALM ; If EPN is dis
446 0303      IF INT34 THEN
447          BEGIN ;be2
448 0304      IF INT34 < 3 THEN ; If not inha
449          BEGIN ;be3
450 0305      IF INT34 = 1 THEN
451          BEGIN ;be4
452 0306      IF FSN1 THEN
453          BEGIN ;be5
454 0307      LET INT34 = 2
455 0309      IF IALM34 = 1 THEN 'PSH-SN215':DIN_ALMCT = OR('PSH-SN215':DIN_ALMCT,32)
456 0310      FPASS = 1
457 0311      GOSUB ACKK
458          END
459          END ;en5
460          END ;en4
461 0313      IF FX34 != 'PSH-SN215':DIN_VAL THEN GOSUB CLR
462 0314      IF FPAS34 THEN
463          BEGIN
464 0316      IF INT34 = 1 THEN DEV1 = OR(DEV1,INT34)
465          END
466          ELSE DEV1 = OR(DEV1,INT34)
467 0319      ELSE IF !AND('PSH-SN215':DIN_ALMCT,16)&!AND('PSH-SN215':DIN_ALMCT,4)&AND('PSH-SN215':DIN_FLG,8) THEN INT34 = 0
468          END ;en2
469          END ;en1
470 0321      ELSE IF INT34 THEN INT34 = 0
471 0322      STEP PTYP BY 1
472 0323      STEP PREV BY 1
473 0324      STEP PSTAT BY 1
474 0325      STEP PSKID BY 1
475 0326      STEP PPASS BY 1
476      ;
477 0327      ILOOP = 36
478 0328      IF IALM35 THEN
479          BEGIN ;be1
480 0330      IF AND('PSH-SN216':DIN_ALMCT,16) THEN LET INT35 = 3 ; If EPN is inha
481 0332      IF AND('PSH-SN216':DIN_ALMCT,4) THEN LET INT35 = 5 ; If EPN is ibad
482 0334      IF !AND('PSH-SN216':DIN_FLG,8) THEN LET INT35 = 4 ; If EPN is dis
483 0337      IF ! INT35 THEN IF FX35 = 'PSH-SN216':DIN_VAL THEN GOSUB ALM
484 0338      IF INT35 THEN
485          BEGIN ;be2
486 0339      IF INT35 < 3 THEN ; If not inha
487          BEGIN ;be3
488 0340      IF INT35 = 1 THEN
489          BEGIN ;be4
490 0341      IF FSN1 THEN
491          BEGIN ;be5
492 0342      LET INT35 = 2
493 0344      IF IALM35 = 1 THEN 'PSH-SN216':DIN_ALMCT = OR('PSH-SN216':DIN_ALMCT,32)
494 0345      FPASS = 1
495 0346      GOSUB ACKK

```


WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:38:41 24-Feb-95

Page 10

```

496             END                                     ;en5
497             END                                     ;en4
498 0348         IF FX35 I= 'PSH-SN216':DIN_VAL THEN GOSUB CLR
499 0349         IF FPAS35 THEN
500             BEGIN
501 0351         IF INT35 = 1 THEN DEV1 = OR(DEV1,INT35)
502             END
503 0352         ELSE DEV1 = OR(DEV1,INT35)
504             END                                     ;en3
505 0354         ELSE IF !AND('PSH-SN216':DIN_ALMCT,16)&!AND('PSH-SN216':DIN_ALMCT,4)&AND('PSH-SN216':DIN_FLG,8)THEN INT35 = 0
506             END                                     ;en2
507             END                                     ;en1
508 0356         ELSE IF INT35 THEN INT35 = 0
509             ;
510             ;
511             ;
512 0357 S2ND:    ACK = 0
513 0358         SORT = 0
514             ;
515 0359         DEV12 = OR(DEVO,DEV6)
516 0360         DEV13 = OR(DEV1,DEV7)
517 0361         DEV14 = OR(DEV2,DEV8)
518 0362         DEV15 = OR(DEV3,DEV9)
519 0363         DEV16 = OR(DEV4,DEV10)
520 0364         DEV17 = OR(DEV5,DEV11)
521             ;
522 0365         ASSIGN PMSG0 = DEV12
523 0366         ASSIGN PMSG1 = FARM
524 0367         FOR LLOOP = 1 TO 6
525 0368         IF PMSG0 THEN
526             BEGIN
527 0370         IF AND(PMSG0,1) THEN PMSG1 = 2
528 0371         ELSE PMSG1 = 1
529             END
530 0372         ELSE PMSG1 = 0
531 0373         STEP PMSG0 BY 1
532 0374         STEP PMSG1 BY 1
533 0375         NEXT LLOOP
534             ;
535 0376         IF FHOLD THEN
536             BEGIN
537 0377         FHOLD = 0
538 0378         TIMERCLR STM2
539 0379         TIMERON STM2
540 0380         LAMP 0,15,FARM
541 0381         LAMP 0,14,SERPIT
542 0382         LAMP 0,13,VLPIT
543 0383         LAMP 0,12,HVAC
544             END
545 0384         MARK STEP "BOTTOM",0,0
546 0385         NOCRIT
547             ;
548             ;
549             ;
550             ; If QS-204AR1 alarm and unack:

```

```

551      ;
552 0386      IF AND('QS-204AR1':DIN_ALMCT,2) & !AND('QS-204AR1':DIN_ALMCT,32) THEN
553      BEGIN
554 0387          FSL6 = 1 ; Alarm not ack
555 0388          LAMP 0,23,2 ; Blink lamp
556      END
557 0389      IF AND('QS-204AR1':DIN_ALMCT,32) & FSL6=1 THEN ; If ACK & 1st pass
558      BEGIN
559 0390          FSL6 = 0 ; Alarm is ack
560 0392          IF FSL7=0 THEN LAMP 0,23,1 ; Lamp steady on
561      END
562 0394      IF !AND('QS-204AR1':DIN_ALMCT,2) & FSL6=1 & AND('QS-204AR2':DIN_ALMCT,2) THEN LAMP 0,23,1
563 0396      IF !AND('QS-204AR1':DIN_ALMCT,2) THEN FSL6 = 0 ; If no alarm
564      ;
565      ; If QS-204AR2 alarm and unack:
566      ;
567 0397      IF AND('QS-204AR2':DIN_ALMCT,2) & !AND('QS-204AR2':DIN_ALMCT,32) THEN
568      BEGIN
569 0398          FSL7 = 1 ; Alarm not ack
570 0399          LAMP 0,23,2 ; Blink lamp
571      END
572 0400      IF AND('QS-204AR2':DIN_ALMCT,32) & FSL7=1 THEN ; IF ACK
573      BEGIN
574 0401          FSL7 = 0 ; Alarm is ack
575 0403          IF FSL6=0 THEN LAMP 0,23,1 ; Lamp steady on
576      END
577 0405      IF !AND('QS-204AR2':DIN_ALMCT,2) & FSL7=1 & AND('QS-204AR1':DIN_ALMCT,2) THEN LAMP 0,23,1
578 0407      IF !AND('QS-204AR2':DIN_ALMCT,2) THEN FSL7 = 0 ; If no alarm
579      ;
580      ; If No Alarms:
581      ;
582 0409      IF !AND('QS-204AR1':DIN_ALMCT,2) & !AND('QS-204AR2':DIN_ALMCT,2) THEN LAMP 0,23,0
583      ;
584      ;
585      ;
586 0410      NOCRIT
587 0411      GOTO AGAIN
588      ;
589      ;
590      ;
591      ;
592      ;
593 0412      CLR: LET PSTAT = 0
594 0414      IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
595 0415      Fhold = 1
596 0416      RETURN
597      ;
598 0417      ALM: LET PSTAT = 1
599 0419      IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
600 0420      ASSIGN PDEV = AFARM
601 0421      STEP PDEV BY PSKID
602 0422      PDEV:DEV_TMO = 2
603 0423      CRIT
604 0424      ASSIGN PMSGO = LEAKO
605 0425      STEP PMSGO BY PSKID

```

```

606 0426      LAMP 0,PMSG0,2
607 0428      IF AND(PTYE,3)THEN FALMON = 1
608
609 0429      CYCL:  FHold = 1
610 0430      LSG4 = LSG3
611 0431      LSG3 = LSG2
612 0432      LSG2 = LSG1
613 0433      LSG1 = LSG0
614 0434      COLOR4 = COLOR3
615 0435      COLOR3 = COLOR2
616 0436      COLOR2 = COLOR1
617 0437      COLOR1 = COLOR0
618 0438      EPNS4 = EPNS3
619 0439      EPNS3 = EPNS2
620 0440      EPNS2 = EPNS1
621 0441      EPNS1 = STRNG1
622 0442      GRAFX4 = GRAFX3
623 0443      GRAFX3 = GRAFX2
624 0444      GRAFX2 = GRAFX1
625 0445      GRAFX1 = GRAFX0
626 0446      SDATE4 = SDATE3
627 0447      SDATE3 = SDATE2
628 0448      SDATE2 = SDATE1
629 0449      SDATE1 = SDATE0
630
631 ;
632 ;
633      ALARM TYPER STRINGS:
634 0450      ACKK:  ON ILOOP GOTO S1,S2,S3,S4,S5,S6,S7,S8,S9,S10
635 0451      ON ILOOP - 10 GOTO S11,S12,S13,S14,S15,S16,S17,S18,S19,S20
636 0452      ON ILOOP - 20 GOTO S21,S22,S23,S24,S25,S26,S27,S28,S29,S30
637 0453      ON ILOOP - 30 GOTO S31,S32,S33,S34,S35,S36,S37,S38,S39
638 0454      GOTO AMSG
639 0455      S1:  STRNG0 = "PUMP AY-101 MCC OFF "
640 0456      STRNG1 = "YS-AY-101"
641 0457      GOTO AMSG
642 0458      S2:  STRNG0 = "PUMP AY-102 MCC OFF "
643 0459      STRNG1 = "YS-AY-102"
644 0460      GOTO AMSG
645 0461      S3:  STRNG0 = "PUMP AZ-101 MCC OFF "
646 0462      STRNG1 = "YS-AZ-101"
647 0463      GOTO AMSG
648 0464      S4:  STRNG0 = "PUMP AZ-102 MCC OFF "
649 0465      STRNG1 = "YS-AZ-102"
650 0466      GOTO AMSG
651 0467      S5:  STRNG0 = "AX-B SL-100 LINE LEAK "
652 0468      STRNG1 = "PSH-SL100"
653 0469      GOTO AMSG
654 0470      S6:  STRNG0 = "AX-A SL-101 LINE LEAK "
655 0471      STRNG1 = "PSH-SL101"
656 0472      GOTO AMSG
657 0473      S7:  STRNG0 = "A-B SL-113 LINE LEAK "
658 0474      STRNG1 = "PSH-SL113"
659 0475      GOTO AMSG
660 0476      S8:  STRNG0 = "A-A SL-114 LINE LEAK "
661 0477      STRNG1 = "PSH-SL114"

```

```

661 0478      GOTO AMSG
662 0479      S9:   STRNG0 = "AX-B SN-200/213 LEAK  "
663 0480          STRNG1 = "PSH200213"
664 0481      GOTO AMSG
665 0482      S10:  STRNG0 = "AX-A SN-201/214 LEAK  "
666 0483          STRNG1 = "PSH201214"
667 0484      GOTO AMSG
668 0485      S11:  STRNG0 = "AZ SL-500 LINE LEAK  "
669 0486          STRNG1 = "LDS-SL500"
670 0487      GOTO AMSG
671 0488      S12:  STRNG0 = "AZ SL-501 LINE LEAK  "
672 0489          STRNG1 = "LDS-SL501"
673 0490      GOTO AMSG
674 0491      S13:  STRNG0 = "AY SL-502 LINE LEAK  "
675 0492          STRNG1 = "LDS-SL502"
676 0493      GOTO AMSG
677 0494      S14:  STRNG0 = "AY SL-503 LINE LEAK  "
678 0495          STRNG1 = "LDS-SL503"
679 0496      GOTO AMSG
680 0497      S15:  STRNG0 = "AY SL-504 LINE LEAK  "
681 0498          STRNG1 = "LDS-SL504"
682 0499      GOTO AMSG
683 0500      S16:  STRNG0 = "AY SL-505 LINE LEAK  "
684 0501          STRNG1 = "LDS-SL505"
685 0502      GOTO AMSG
686 0503      S17:  STRNG0 = "AZ SN-600 LINE LEAK  "
687 0504          STRNG1 = "LDS-SN600"
688 0505      GOTO AMSG
689 0506      S18:  STRNG0 = "AZ SN-601 LINE LEAK  "
690 0507          STRNG1 = "LDS-SN601"
691 0508      GOTO AMSG
692 0509      S19:  STRNG0 = "AY DR-700 LINE LEAK  "
693 0510          STRNG1 = "LDS-DR700"
694 0511      GOTO AMSG
695 0512      S20:  STRNG0 = "AX-A/B SELPOT HI FAIL  "
696 0513          STRNG1 = "LXS-AXPOT"
697 0514      GOTO AMSG
698 0515      S21:  STRNG0 = "AX-A/B SEAL POT HIGH  "
699 0516          STRNG1 = "LSH-AX-1"
700 0517      GOTO AMSG
701 0518      S22:  STRNG0 = "AX-A/B SEAL POT LOW  "
702 0519          STRNG1 = "LSL-AX-1"
703 0520      GOTO AMSG
704 0521      S23:  STRNG0 = "A/AX COB LEAK  "
705 0522          STRNG1 = "LSCOBAAAX"
706 0523      GOTO AMSG
707 0524      S24:  STRNG0 = "A/AX AREA RADN BYPASS  "
708 0525          STRNG1 = "YS-AAXARM"
709 0526      GOTO AMSG
710 0527      S25:  STRNG0 = "A/AX MASTER SHUTDOWN  "
711 0528          STRNG1 = "YS-AAXMSD"
712 0529      GOTO AMSG
713 0530      S26:  STRNG0 = "A/AX FARM RADN HIGH  "
714 0531          STRNG1 = "RSHAAX3/4"
715 0532      GOTO AMSG

```

```

716 0533 S27:  STRNG0 = "A/AX/AY/AZ RD HI/FAIL "
717 0534      STRNG1 = "RSH-AXZY1"
718 0535      GOTO AMSG
719 0536 S28:  STRNG0 = "A/AX/AZ PPT AZ COB LK "
720 0537      STRNG1 = "LOSPPCOB1"
721 0538      GOTO AMSG
722 0539 S29:  STRNG0 = "A3/AX3/AX4 RAD MON FA "
723 0540      STRNG1 = "RXSAAX3/4"
724 0541      GOTO AMSG
725      S30:  STRNG0 = "LEAK-DET CNSL GEN ALM "      -mct2+
726      ;
727      ;
728 0542 S31:  STRNG0 = "PUMP AY-101 SHUTDOWN "
729 0543      STRNG1 = "P-AY-101"
730 0544      GOTO AMSG
731 0545 S32:  STRNG0 = "PUMP AY-102 SHUTDOWN "
732 0546      STRNG1 = "P-AY-102"
733 0547      GOTO AMSG
734 0548 S33:  STRNG0 = "PUMP AZ-101 SHUTDOWN "
735 0549      STRNG1 = "P-AZ-101"
736 0550      GOTO AMSG
737 0551 S34:  STRNG0 = "PUMP AZ-102 SHUTDOWN "
738 0552      STRNG1 = "P-AZ-102"
739 0553      GOTO AMSG
740 0554 S35:  STRNG0 = "A SN-215 LINE LEAK "
741 0555      STRNG1 = "PSH-SN215"
742 0556      GOTO AMSG
743 0557 S36:  STRNG0 = "A SN-216 LINE LEAK "
744 0558      STRNG1 = "PSH-SN216"
745 0559      GOTO AMSG
746 0560 S37:  STRNG0 = "XYZBAT PROGRAM START "
747 0561      GOTO AMSG
748 0562 S38:  STRNG0 = "XYZBAT IN WRONG UNIT "
749 0563      GOTO AMSG
750 0564 S39:  STRNG0 = "XYZBAT TERMINATING "
751      ;
752 0565 AMSG:  IF (FPASS)THEN
753      BEGIN
754      PRINT AD1,"%p ALARM ACKNOWLEDGE",STRNG1
755      FPASS = 0
756      GOTO NXT
757      END
758 0570      IF PTYPE = 4 THEN COLOR0 = 3
759 0571      ELSE COLOR0 = PTYPE
760 0572      EPNS0 = STRNG1
761 0573      IN0 = TIMSTR(SSNG0)
762 0575      IF IN0 != 8 THEN SSNG0 = "TIMEFAIL"
763 0576      IN0 = APPEND(LSG0,STRNG0,SSNG0)
764 0578      IF IN0 < 0 THEN LSG0 = "APPEND FAILED!!"
765 0579      IN0 = DATSTR(SDATED0)
766 0581      IF IN0 != 9 THEN SDATED0 = "DATE FAIL"
767 0583      IF PSKID = 1 THEN GRAFX0 = 272
768 0585      IF PSKID = 2 THEN GRAFX0 = 272
769 0587      IF PSKID = 3 THEN GRAFX0 = 273
770 0589      IF PSKID = 4 THEN GRAFX0 = 274

```

```

771 0591      IF PSKID = 5 THEN GRAFX0 = 270
772 0593      IF PSKID = 0 THEN GRAFX0 = 270
773 0595      IF PTYPE = 1 THEN PRINT ADI,"WHITE ALARM %p %p GRAPHICS %i",EPNSO,STRNGO,GRAFX0
774 0596  HXT:  IF FSTART = 1 THEN
775              BEGIN
776 0597              COLORO = 0
777 0598              GRAFX0 = 0
778 0599              STRNG1 =" "
779 0600              FSTART = 0
780 0601              PRINT ADI,"%p",STRNGO
781              END
782 0602      RETURN
783      ;
784 0603  ERRSUB:IF ERRN != ERRNUM() THEN
785              BEGIN
786 0604              LET ERRN = ERRNUM()
787 0605              LET ERRL = ERRLIN()
788 0606              PRINT ADI,"ERROR # %i AT LINE %i, ILOOP = %i",errn,errl,ILOOP
789              END
790 0607      ERSCLR
791 0608      GOTO AGAIN
792      ;
793      ; TERM: LET FABORT = 0
794 0609  TERM:  LET STRNGO = "XYZBAT TERMINATED  "
795 0610          LET ILOOP = 39
796 0611          LET FSTART = 1
797 0612          TIMEOFF STMRI
798 0613          TIMECLR STMRI
799 0614          LET FLDALM = 0
800 0615          OPEN 'HSAXYZSIL'
801 0616          GOSUB CYCL
802          END

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 12032 Bytes, 752 Pages
95 Forward Reference(s)

242BATCH

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:31:43 24-Feb-95

Page 1

```

1      : 2 4 2 B A T C H . S E Q                                VERSION 2
2      :
3      : Function:      Sequence program for Lamp function,
4      :               message display on Graphic number 023,
5      :               and printer messages.
6      :
7      : Written By    : R. W. Kersey
8      : Date         : 05-31-92
9      : Facility     : 242-A Evaporator Bldg.
10     : Machine      : VAX 4104-A
11     : Op. System   : VAX/VMS 5.5-2AH
12     : D/3 Version  : 6.3-2 BASELINE
13     : Compiler     : SIC
14     : Build command : @I320,20I242BATCH
15     :
16     : UNIT = 5, SKDKEY = 47
17     :
18     : Three AIs (RI-K1-1-1, RI-K1-1-2, and FI-K1-1) are found in unit ROOMS in PCMOA.
19     :
20     : Assoc. UPF:    [300,1] TNKFRM._____ TNKFRM.LST
21     : Assoc. RPF:    [300,1] FARM._____ FARM.LST
22     : SDB:          [300,1]PCM1ASDB._____ PCM1ASDB.CMD
23     :
24     : POINTER      SDB VAR.      FUNCTION
25     : -----
26     : PSTAT        INTn          Current EPN alarm state is an integer tied to
27     :               (n=0-100)    graphic flashing string of value:
28     :               -             0 = previously not alarmed
29     :               -             1 = previously alarmed, not ack
30     :               -             2 = previously alarmed, is ack
31     :               -             3 = alarm is inhibited
32     :               -             4 = input is off scan
33     :               -             5 = input is bad
34     : PDIN          'EPN'         ---
35     : PTYPE         IALMn         alarm type 0 = none, 1 = white, 2 = yellow,
36     :               -             3 = yellow and yellow, 4 = red,
37     :               -             6 = yellow and red
38     : PREV          FXn          Reverse acting (not fail safe) flag (1=set)
39     : PSKID         SKIDn        Tracks which graphic has an alarm
40     : PPASS         FPASn        If set (1), turn lamp off when alarm is ACKnowledged
41     : PSN          FSNn         Flag for the graphic to ACKnowledge
42     : ---          FHOLD        Flag to update the lamps
43     : ---          FNING        Flag for DOG1A.C micro select
44     : ---          ACK          Flag for C program to ACK system wide
45     :
46     : MODIFICATION LOG:
47     :
48     : 10-DEC-92      Modified STRNG0 length.                      M. Teats
49     :
50     : 15-DEC-92      Modified for Rad Systems Upgrade.           M. Teats
51     :
52     : 30-DEC-92      Fixed white alarm print message.           M. Teats
53     :
54     : 15-JUN-93      Added comments and baselined for           M. Teats
55     :                 D/3 version 3.3. CCN# 93JUN01-01.

```



```

56      ;
57      ;      15-DEC-94      Revised header to indicate the new VMS      J. Berger
58      ;      system. Rebaselined document. Removed
59      ;      obsolete Select() statement.
60      ;
61      ;      END MOD LOG
62      ;
63      UNIT      BLDNG
64      COMMON ROOMS
65      ;
66      0000      NOP
67      0001      LET STAR = 45      ; Start message
68      0002      LET SKDKEY = 47      ; SKID Key
69      0003      LET UNITT = 5      ; SEQ Unit number
70      0004      ASSIGN PDEV = 'BUILDING' ; Alarm device
71      0005      ASSIGN PTYPE = IALMD
72      0006      LET ADI = 0      ; Alarm destination index
73      0007      LET FSN0 = 0      ; No alarm yet
74      0008      TIMEROFF STMRO      ; Alarm timer off
75      0009      TIMERCLR STMRO      ; Alarm timer clear
76      0010      LET STRNG0 = " "      ; Blank 30 character string
77      0011      LET ERRN = 0      ; Set error number to zero
78      0012      LET ERRL = 0      ; Set error line to zero
79      0013      LET ACK = 0
80      0014      GETUNIT TRIP      ; Which sequence unit is program running in
81      0015      IF TRIP != UNITT THEN      ; Abort if running in wrong seq. unit
82      BEGIN
83      0016      FSTART = 1
84      0017      ILOOP = STAR + 1
85      0018      PTYPE = 0
86      0019      GOSUB ACKK
87      0020      GOTO TERM
88      END
89      ;      IF !SELECT() THEN GOTO AGAIN
90      ;
91      ;      BEGIN INITIALIZING:
92      ;
93      ;      0-27 are PCM1 DIN, 28-41 are PCM1 AIN, 42-44 are PCMD AIN
94      ;
95      0021      LET SSWG0 = "INIT"
96      0022      INIT:      LET FABORT = 0      ; If FABORT = 1 then abort program
97      0023      LET FSTART = 0      ; Program start flag = 0
98      0024      LAMP ADI,SKDKEY,0      ; Turn off the annunciator key
99      0025      LET LSG0 = " "      ; Clear the string variables
100     0026      LET LSG1 = " "
101     0027      LET LSG2 = " "
102     0028      LET LSG3 = " "
103     0029      LET LSG4 = " "
104     0030      LET PDEV:DEV_TMPO = 1      ; Set alarm device to non-alarm state
105     0031      IF AND(PDEV:DEV_ALMCT,16) THEN      ; If alarm device is inha, remove the inha
106     0032      LET PDEV:DEV_ALMCT=XOR(PDEV:DEV_ALMCT,16)
107     0033      IF AND(PDEV:DEV_ALMCT,32) THEN      ; If alarm device is ack, clear it
108     0034      LET PDEV:DEV_ALMCT = XOR(PDEV:DEV_ALMCT,32)
109     ;
110     ;      INITIALIZE PCM1 DINs AND AINs:

```

```

111      ;
112 0035      ASSIGN PDIN = 'RSH-ARAMU'
113 0036      ASSIGN PSTAT = INTO
114 0037      ASSIGN PTTYPE = IALM0
115 0038      ASSIGN PREV = FX0
116 0039      ASSIGN PSKID = SKID0
117 0040      ASSIGN PPASS = FPAS0
118      ;
119 0041      FOR ILOOP = 0 TO 41                ; 0-27 are PCM1 DINs, 28-41 are PCM1 AINs
120 0042          LET PPASS = 0
121 0043          LET PREV = 0
122 0045          IF PSTAT > 5 THEN PSTAT = 0      ; Reset current status if undefined
123 0046          LET PSKID = 0
124 0047          LET PTTYPE = 2                  ; Default to yellow alarm
125 0048          IF ILOOP <= 27 THEN              ; For PCM1 DINs
126              BEGIN
127 0049              IF (AND(PDIN:DIN_DFLG,3)) THEN ; Check for alarms (Must be red or yellow)
128                  BEGIN
129 0050                  LET PTTYPE = 2            ; Default to yellow alarm
130 0052                  IF AND(PDIN:DIN_DFLG,1) THEN LET PREV = 1 ; Reverse acting
131 0054                  IF AND(PDIN:DIN_DFLG,4) THEN LET PTTYPE = 4 ; Set to red critical
132                      END
133 0056                  IF ILOOP < 27 THEN STEP PDIN BY 1 ; Point to next DIN
134                      END
135 0057                  ELSE LET PTTYPE = 4        ; For PCM1 AINs
136 0058                  STEP PTTYPE BY 1
137 0059                  STEP PREV BY 1
138 0060                  STEP PSTAT BY 1
139 0061                  STEP PSKID BY 1
140 0062                  STEP PPASS BY 1
141 0063                  NEXT ILOOP
142      ;
143      ;
144      ;
145      ;
146      ;
147 0064          LET IALM42 = 4                ; RI-K1-1-1
148 0065          LET SKID42 = 0
149 0066          LET IALM43 = 4                ; RI-K1-1-2
150 0067          LET SKID43 = 0
151 0068          LET IALM44 = 0                ; FI-K1-1
152 0069          LET SKID44 = 0
153      ;
154      ;
155      ;
156      ;
157      ;
158 0070          LET PTTYPE = 0
159 0071          LET ILOOP = STAR
160 0072          LET FSTART = 1
161 0073          GOSUB ACKK
162 0074          ON ERROR GOSUB ERRSUB
163      ;
164      ;
165      ;
166      ;
167      ;
168      ;
169      ;
170      ;
171      ;
172      ;
173      ;
174      ;
175      ;
176      ;
177      ;
178      ;
179      ;
180      ;
181      ;
182      ;
183      ;
184      ;
185      ;
186      ;
187      ;
188      ;
189      ;
190      ;
191      ;
192      ;
193      ;
194      ;
195      ;
196      ;
197      ;
198      ;
199      ;
200      ;
201      ;
202      ;
203      ;
204      ;
205      ;
206      ;
207      ;
208      ;
209      ;
210      ;
211      ;
212      ;
213      ;
214      ;
215      ;
216      ;
217      ;
218      ;
219      ;
220      ;
221      ;
222      ;
223      ;
224      ;
225      ;
226      ;
227      ;
228      ;
229      ;
230      ;
231      ;
232      ;
233      ;
234      ;
235      ;
236      ;
237      ;
238      ;
239      ;
240      ;
241      ;
242      ;
243      ;
244      ;
245      ;
246      ;
247      ;
248      ;
249      ;
250      ;
251      ;
252      ;
253      ;
254      ;
255      ;
256      ;
257      ;
258      ;
259      ;
260      ;
261      ;
262      ;
263      ;
264      ;
265      ;
266      ;
267      ;
268      ;
269      ;
270      ;
271      ;
272      ;
273      ;
274      ;
275      ;
276      ;
277      ;
278      ;
279      ;
280      ;
281      ;
282      ;
283      ;
284      ;
285      ;
286      ;
287      ;
288      ;
289      ;
290      ;
291      ;
292      ;
293      ;
294      ;
295      ;
296      ;
297      ;
298      ;
299      ;
300      ;
301      ;
302      ;
303      ;
304      ;
305      ;
306      ;
307      ;
308      ;
309      ;
310      ;
311      ;
312      ;
313      ;
314      ;
315      ;
316      ;
317      ;
318      ;
319      ;
320      ;
321      ;
322      ;
323      ;
324      ;
325      ;
326      ;
327      ;
328      ;
329      ;
330      ;
331      ;
332      ;
333      ;
334      ;
335      ;
336      ;
337      ;
338      ;
339      ;
340      ;
341      ;
342      ;
343      ;
344      ;
345      ;
346      ;
347      ;
348      ;
349      ;
350      ;
351      ;
352      ;
353      ;
354      ;
355      ;
356      ;
357      ;
358      ;
359      ;
360      ;
361      ;
362      ;
363      ;
364      ;
365      ;
366      ;
367      ;
368      ;
369      ;
370      ;
371      ;
372      ;
373      ;
374      ;
375      ;
376      ;
377      ;
378      ;
379      ;
380      ;
381      ;
382      ;
383      ;
384      ;
385      ;
386      ;
387      ;
388      ;
389      ;
390      ;
391      ;
392      ;
393      ;
394      ;
395      ;
396      ;
397      ;
398      ;
399      ;
400      ;
401      ;
402      ;
403      ;
404      ;
405      ;
406      ;
407      ;
408      ;
409      ;
410      ;
411      ;
412      ;
413      ;
414      ;
415      ;
416      ;
417      ;
418      ;
419      ;
420      ;
421      ;
422      ;
423      ;
424      ;
425      ;
426      ;
427      ;
428      ;
429      ;
430      ;
431      ;
432      ;
433      ;
434      ;
435      ;
436      ;
437      ;
438      ;
439      ;
440      ;
441      ;
442      ;
443      ;
444      ;
445      ;
446      ;
447      ;
448      ;
449      ;
450      ;
451      ;
452      ;
453      ;
454      ;
455      ;
456      ;
457      ;
458      ;
459      ;
460      ;
461      ;
462      ;
463      ;
464      ;
465      ;
466      ;
467      ;
468      ;
469      ;
470      ;
471      ;
472      ;
473      ;
474      ;
475      ;
476      ;
477      ;
478      ;
479      ;
480      ;
481      ;
482      ;
483      ;
484      ;
485      ;
486      ;
487      ;
488      ;
489      ;
490      ;
491      ;
492      ;
493      ;
494      ;
495      ;
496      ;
497      ;
498      ;
499      ;
500      ;
501      ;
502      ;
503      ;
504      ;
505      ;
506      ;
507      ;
508      ;
509      ;
510      ;
511      ;
512      ;
513      ;
514      ;
515      ;
516      ;
517      ;
518      ;
519      ;
520      ;
521      ;
522      ;
523      ;
524      ;
525      ;
526      ;
527      ;
528      ;
529      ;
530      ;
531      ;
532      ;
533      ;
534      ;
535      ;
536      ;
537      ;
538      ;
539      ;
540      ;
541      ;
542      ;
543      ;
544      ;
545      ;
546      ;
547      ;
548      ;
549      ;
550      ;
551      ;
552      ;
553      ;
554      ;
555      ;
556      ;
557      ;
558      ;
559      ;
560      ;
561      ;
562      ;
563      ;
564      ;
565      ;
566      ;
567      ;
568      ;
569      ;
570      ;
571      ;
572      ;
573      ;
574      ;
575      ;
576      ;
577      ;
578      ;
579      ;
580      ;
581      ;
582      ;
583      ;
584      ;
585      ;
586      ;
587      ;
588      ;
589      ;
590      ;
591      ;
592      ;
593      ;
594      ;
595      ;
596      ;
597      ;
598      ;
599      ;
600      ;
601      ;
602      ;
603      ;
604      ;
605      ;
606      ;
607      ;
608      ;
609      ;
610      ;
611      ;
612      ;
613      ;
614      ;
615      ;
616      ;
617      ;
618      ;
619      ;
620      ;
621      ;
622      ;
623      ;
624      ;
625      ;
626      ;
627      ;
628      ;
629      ;
630      ;
631      ;
632      ;
633      ;
634      ;
635      ;
636      ;
637      ;
638      ;
639      ;
640      ;
641      ;
642      ;
643      ;
644      ;
645      ;
646      ;
647      ;
648      ;
649      ;
650      ;
651      ;
652      ;
653      ;
654      ;
655      ;
656      ;
657      ;
658      ;
659      ;
660      ;
661      ;
662      ;
663      ;
664      ;
665      ;
666      ;
667      ;
668      ;
669      ;
670      ;
671      ;
672      ;
673      ;
674      ;
675      ;
676      ;
677      ;
678      ;
679      ;
680      ;
681      ;
682      ;
683      ;
684      ;
685      ;
686      ;
687      ;
688      ;
689      ;
690      ;
691      ;
692      ;
693      ;
694      ;
695      ;
696      ;
697      ;
698      ;
699      ;
700      ;
701      ;
702      ;
703      ;
704      ;
705      ;
706      ;
707      ;
708      ;
709      ;
710      ;
711      ;
712      ;
713      ;
714      ;
715      ;
716      ;
717      ;
718      ;
719      ;
720      ;
721      ;
722      ;
723      ;
724      ;
725      ;
726      ;
727      ;
728      ;
729      ;
730      ;
731      ;
732      ;
733      ;
734      ;
735      ;
736      ;
737      ;
738      ;
739      ;
740      ;
741      ;
742      ;
743      ;
744      ;
745      ;
746      ;
747      ;
748      ;
749      ;
750      ;
751      ;
752      ;
753      ;
754      ;
755      ;
756      ;
757      ;
758      ;
759      ;
760      ;
761      ;
762      ;
763      ;
764      ;
765      ;
766      ;
767      ;
768      ;
769      ;
770      ;
771      ;
772      ;
773      ;
774      ;
775      ;
776      ;
777      ;
778      ;
779      ;
780      ;
781      ;
782      ;
783      ;
784      ;
785      ;
786      ;
787      ;
788      ;
789      ;
790      ;
791      ;
792      ;
793      ;
794      ;
795      ;
796      ;
797      ;
798      ;
799      ;
800      ;
801      ;
802      ;
803      ;
804      ;
805      ;
806      ;
807      ;
808      ;
809      ;
810      ;
811      ;
812      ;
813      ;
814      ;
815      ;
816      ;
817      ;
818      ;
819      ;
820      ;
821      ;
822      ;
823      ;
824      ;
825      ;
826      ;
827      ;
828      ;
829      ;
830      ;
831      ;
832      ;
833      ;
834      ;
835      ;
836      ;
837      ;
838      ;
839      ;
840      ;
841      ;
842      ;
843      ;
844      ;
845      ;
846      ;
847      ;
848      ;
849      ;
850      ;
851      ;
852      ;
853      ;
854      ;
855      ;
856      ;
857      ;
858      ;
859      ;
860      ;
861      ;
862      ;
863      ;
864      ;
865      ;
866      ;
867      ;
868      ;
869      ;
870      ;
871      ;
872      ;
873      ;
874      ;
875      ;
876      ;
877      ;
878      ;
879      ;
880      ;
881      ;
882      ;
883      ;
884      ;
885      ;
886      ;
887      ;
888      ;
889      ;
890      ;
891      ;
892      ;
893      ;
894      ;
895      ;
896      ;
897      ;
898      ;
899      ;
900      ;
901      ;
902      ;
903      ;
904      ;
905      ;
906      ;
907      ;
908      ;
909      ;
910      ;
911      ;
912      ;
913      ;
914      ;
915      ;
916      ;
917      ;
918      ;
919      ;
920      ;
921      ;
922      ;
923      ;
924      ;
925      ;
926      ;
927      ;
928      ;
929      ;
930      ;
931      ;
932      ;
933      ;
934      ;
935      ;
936      ;
937      ;
938      ;
939      ;
940      ;
941      ;
942      ;
943      ;
944      ;
945      ;
946      ;
947      ;
948      ;
949      ;
950      ;
951      ;
952      ;
953      ;
954      ;
955      ;
956      ;
957      ;
958      ;
959      ;
960      ;
961      ;
962      ;
963      ;
964      ;
965      ;
966      ;
967      ;
968      ;
969      ;
970      ;
971      ;
972      ;
973      ;
974      ;
975      ;
976      ;
977      ;
978      ;
979      ;
980      ;
981      ;
982      ;
983      ;
984      ;
985      ;
986      ;
987      ;
988      ;
989      ;
990      ;
991      ;
992      ;
993      ;
994      ;
995      ;
996      ;
997      ;
998      ;
999      ;
1000     ;

```

```

166 0075 AGAIN: MARK STEP "AGAIN",0,0
167 ; IF (FABORT) THEN GOTO TERM ; Terminate program if abort flag set
168 0076 SNGO = "WAIT"
169 ; JH01 WAITUNTIL SELECT() ; Wait until micro is selected
170 0077 WAIT 1
171 0078 IF (FSNO) THEN ; If alarm flag set
172 0079 IF STMRO > 60 THEN ; If alarm flag set for 1 minute
173 BEGIN ; Reset timer and turn off flag
174 0080 TIMEROFF STMRO
175 0081 TIMERCLR STMRO
176 0082 FSNO = 0
177 0083 ERRN = 0
178 0084 ERRL = 0
179 END
180 ;
181 ; IF BUILDING $T1 is alarmed and it's ALMCT indicates alarms are acknowledged:
182 ;
183 0085 IF AND(PDEV:DEV_TMPO,2) & AND(PDEV:DEV_ALMCT,32) THEN
184 BEGIN
185 0086 LET PDEV:DEV_TMPO = 1 ; Put BUILDING in ACK state
186 0087 LET FTRIP0 = 1 ; Set for Unit alarm acknowledgement, system to graphip
187 0088 LET ACK = 1 ; Set for PCMO Unit alarm acknowledgement, graphip to system
188 0089 LET FLGACK = 1 ; Set for PCM1 Unit alarm acknowledgement, graphip to system
189 0090 CRIT
190 0091 LAMP ADI,SKDKEY,1 ; Turn lamp on
191 END
192 ;
193 ;
194 ;
195 0092 ASSIGN PDIN = 'RSH-ARAMU'
196 0093 ASSIGN PTYPE = IALMO
197 0094 ASSIGN PREV = FXO
198 0095 ASSIGN PSTAT = INTO
199 0096 ASSIGN PPASS = FPASO
200 0097 LET CURENT = 0 ; Used to set lamp status
201 0098 LET LAST = 0
202 0099 LET NXT = 0
203 0100 LET LMPVAR = 0
204 0101 LET BITVAL = 0
205 ;
206 0102 LOP: FOR ILOOP = 0 TO 27
207 ; IF ISELECT() THEN GOTO AGAIN
208 0103 IF ( PTYPE > 0 ) THEN ; If some type of alarm
209 BEGIN ;be1
210 0105 IF AND(PDIN:DIN_ALMCT,16) THEN LET PSTAT = 3 ; If EPN is inhibited
211 0107 IF AND(PDIN:DIN_ALMCT,4) THEN LET PSTAT = 5 ; If EPN is bad input
212 0109 IF IAND(PDIN:DIN_FLG,8) THEN LET PSTAT = 4 ; If EPN is not disabled
213 0110 IF ( PSTAT < 3 ) THEN ; If not inhibited
214 BEGIN ;be2
215 0111 IF PSTAT = 1 THEN ;be3
216 BEGIN
217 0112 IF PTYPE != 1 THEN
218 0114 IF AND(PDIN:DIN_ALMCT,32) THEN GOSUB ACK1
219 0116 IF (FTRIP4) THEN GOSUB ACK1
220 END ;en3

```

```

221 0119      IF PSTAT = 0 THEN IF(PDIN:DIN_VAL = PREV) THEN GOSUB ALM
222 0120      IF (PSTAT) THEN
223          BEGIN                                ;be4
224 0122          IF PDIN:DIN_VAL != PREV THEN GOSUB CLR
225 0123          NXT = PSTAT
226 0125          IF (PPASS) & (PSTAT = 2 ) THEN NXT = 0
227 0126          LAST = OR(LAST,NXT)
228          END                                ;en4
229          END                                ;en2
230 0128      ELSE IF !AND(PDIN:DIN_ALMCT,16)&!AND(PDIN:DIN_ALMCT,4)&AND(PDIN:DIN_FLG,8)THEN LET PSTAT = 0
231          END                                ;en1
232 0130      IF ILOOP < 27 THEN STEP PDIN BY 1
233 0131      STEP PTYPE BY 1
234 0132      STEP PREV BY 1
235 0133      STEP PSTAT BY 1
236 0134      STEP PPASS BY 1
237 0135      NEXT ILOOP
238      ;
239 0137      IF (INT5 + INT6) > 0 THEN LET FTRIP1 = 1 ; For K1-6-1 combined dp
240 0138      ELSE LET FTRIP1 = 0
241 0140      IF (INT7 + INT8) > 0 THEN LET FTRIP2 = 1 ; For K1-6-2 combined dp
242 0141      ELSE LET FTRIP2 = 0
243      ;
244      ;
245      ;
246      BEGIN PCM1 ANALOGS:
247      ;
248 0142      ANALOG = 0
249 0143      SNGO = "RUN"
250 0144      ASSIGN PSKID = SKID28
251 0145      ASSIGN PTYPE = IALN28
252 0146      ASSIGN PAI = 'RI-AR-AMU'
253 0147      FOR ILOOP = 28 TO 41
254      ;
255 0148      IF !SELECT() THEN GOTO AGAIN
256          IF(PTYPE) THEN
257              BEGIN                                ;be1
258 0149              BITVAL = PAI:AI_ALMCT
259 0150              IF !AND(BITVAL,16) THEN
260                  BEGIN                                ;be2
261 0151                  IF AND(BITVAL,15) THEN
262                      BEGIN                                ;be3
263 0153                      IF AND(BITVAL,32) THEN NXT = 2
264                      ELSE
265                          BEGIN                                ;be4
266 0154                          IF PSKID != BITVAL THEN
267                              BEGIN                                ;be5
268                                  LET PSKID = BITVAL
269                                  IF BITVAL < 3 THEN FALMON = 1
270                                  PDEV:DEV_TMPO = 2
271                                  NXT = 1
272                                  END                                ;en5
273                              ELSE NXT = 1
274                              END                                ;en4
275                          ELSE

```

WHC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:31:43 24-Feb-95

Page 6

```

276          BEGIN
277 0161      NXT = 0
278 0162      LET PSKID = 0
279          END
280          END ;en2
281      ELSE
282          BEGIN
283 0163      NXT = 0
284 0164      LET PSKID = 0
285          END
286 0165      ANALOG = OR(ANALOG,NXT)
287          END ;en1
288 0166      STEP PSKID BY 1
289 0167      STEP PTYPE BY 1
290 0169      IF ILOOP < 41 THEN STEP PAI BY 1
291 0170      NEXT ILOOP
292      ;
293      ;
294      ;
295      ;
296      ;
297 0171      ILOOP = 42
298 0172      ASSIGN PAI = 'RI-K1-1-1' ; 42
299 0173      GOSUB ANLG1
300 0174      ASSIGN PAI = 'RI-K1-1-2' ; 43
301 0175      GOSUB ANLG1
302 0176      ASSIGN PAI = 'FI-K1-1' ; 44
303 0177      GOSUB ANLG1
304      ;
305      ;
306      ;
307 0178      NOCRIT
308 0180      IF LAST > 3 THEN PRINT AD1," LAMP VARIABLE IS GREATER THAN 3!"
309 0181      CURENT = OR(LAST,ANALOG)
310 0183      IF AND(CURENT,1) THEN LMPVAR = 2
311 0185      ELSE IF AND(CURENT,2) THEN LMPVAR = 1
312 0186      ELSE LMPVAR = 0
313 0187      LAMP AD1,SKOKEY,LMPVAR
314 0188      MARK STEP "BOTTOM",0,0
315 0190      IF AND(BUILDING:DEV_FLG,64) THEN PUTMANL BUILDING
316 0191      LET ACK = 0
317 0192      GOTO AGAIN
318      ;
319      ;
320      ;
321      ;
322      ;
323 0193  ANLG1:  BITVAL = PAI:AI ALMCT
324      ;
325 0194      IF !SELECT() THEN GOTO AGAIN
326      IF !AND(BITVAL,16) THEN
327          BEGIN ;be2
328          IF AND(BITVAL,15) THEN
329              BEGIN ;be3
330              IF AND(BITVAL,32) THEN NXT = 2
331              ELSE

```

```

331          BEGIN                                ;be4
332 0198      IF PSKID != BITVAL THEN
333          BEGIN                                ;be5
334 0199      LET PSKID = BITVAL
335 0200      LET PDEV:DEV_TMPO = 2
336 0201      NXT = 1
337          END                                ;en5
338 0202      ELSE NXT = 1
339          END                                ;en4
340          END                                ;en3
341      ELSE
342      BEGIN
343 0203      NXT = 0
344 0204      LET PSKID = 0
345          END
346          END                                ;en2
347      ELSE
348      BEGIN
349 0205      NXT = 0
350 0206      LET PSKID = 0
351          END
352 0207      ANALOG = OR(ANALOG,NXT)
353 0208      STEP PSKID BY 1
354 0209      STEP PTYPE BY 1
355 0210      ILOOP = ILOOP + 1
356 0211      RETURN
357      ;
358      ;
359      ;
360      ;
361      ;
362 0212 CLR:  LET PSTAT = 0
363 0215      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
364 0216      LET FCLEAR = 1
365 0217      GOSUB ACKK
366 0218      RETURN
367      ;
368 0219 ALM:  LET PSTAT = 1
369 0222      IF PTYPE != 1 THEN IF AND(PDIN:DIN_ALMCT,32) THEN PDIN:DIN_ALMCT = XOR(PDIN:DIN_ALMCT,32)
370 0224      IF AND(PTYPE,3) THEN FALMON = 1
371 0225      LET PDEV:DEV_TMPO = 2
372 0226      LAMP ADI,SKDKEY,2
373 0227      CRIT
374 0228      GOSUB ACKK
375 0229      RETURN
376      ;
377 0230 ACK1: LET PSTAT = 2
378 0231      LET FACK = 1
379 0232      LAMP ADI,SKDKEY,1
380      ;
381      ;
382      ;
383 0234 ACKK: IF ILOOP = 0 THEN STRNG0 = "AMU AREA RAD HI "
384 0236      IF ILOOP = 1 THEN STRNG0 = "AMU AIRSMP CAM RAD HI"
385 0238      IF ILOOP = 2 THEN STRNG0 = "242A K1 ALPHA RAD HI "

```

```

386 0240      IF ILOOP = 3 THEN STRNGO = "242A K1 8/GAMA RAD HI"
387 0242      IF ILOOP = 4 THEN STRNGO = "242A K1 RAD MON TROUB"
388 0244      IF ILOOP = 5 THEN STRNGO = "242A K1-6-1 FLT DP HI"
389 0246      IF ILOOP = 6 THEN STRNGO = "242A K1-6-1 FLT DP LO"
390 0248      IF ILOOP = 7 THEN STRNGO = "242A K1-6-2 FLT DP HI"
391 0250      IF ILOOP = 8 THEN STRNGO = "242A K1-6-2 FLT DP LO"
392 0252      IF ILOOP = 9 THEN STRNGO = "242A K1-6-3 FLT DP LO"
393 0254      IF ILOOP = 10 THEN STRNGO = "242A K1-6-4 FLT DP LO"
394 0256      IF ILOOP = 11 THEN STRNGO = "242A TURBINE FAN ON "
395 0258      IF ILOOP = 12 THEN STRNGO = "HPT AIRSMP CAM RAD HI"
396 0260      IF ILOOP = 13 THEN STRNGO = "CND AIRSMP CAM RAD HI"
397 0262      IF ILOOP = 14 THEN STRNGO = "HVAC AREA RAD HI  "
398 0264      IF ILOOP = 15 THEN STRNGO = "CND LVL4 AREA RAD HI "
399 0266      IF ILOOP = 16 THEN STRNGO = "CND BSMT AREA RAD HI "
400 0268      IF ILOOP = 17 THEN STRNGO = "EVAP RM AREA RAD HI "
401 0270      IF ILOOP = 18 THEN STRNGO = "LOADOUT AREA RAD HI "
402 0272      IF ILOOP = 19 THEN STRNGO = "CND AIR SAMP CAM FAIL"
403 0274      IF ILOOP = 20 THEN STRNGO = "AMU AIR SAMP CAM FAIL"
404 0276      IF ILOOP = 21 THEN STRNGO = "HPT AIR SAMP CAM FAIL"
405 0278      IF ILOOP = 22 THEN STRNGO = "EVAP RM AREA RAD FAIL"
406 0280      IF ILOOP = 23 THEN STRNGO = "AMU AREA RAD FAIL  "
407 0282      IF ILOOP = 24 THEN STRNGO = "HVAC AREA RAD FAIL  "
408 0284      IF ILOOP = 25 THEN STRNGO = "LOADOUT AREA RAD FAIL"
409 0286      IF ILOOP = 26 THEN STRNGO = "CND LVL4 AREA RAD FAIL"
410 0288      IF ILOOP = 27 THEN STRNGO = "CND BAS AREA RAD FAIL"
411 0290      IF ILOOP = 45 THEN STRNGO = "242BATCH PGM START  "
412 0292      IF ILOOP = 46 THEN STRNGO = "244BATCH WRONG UNIT  "
413 0294      IF ILOOP = 47 THEN STRNGO = "244BATCH TERMINATING "
414
415 0295      ;
416      AMSG:  IF PTYPE = 1 THEN
417              BEGIN
418              IF (FACK = 0) & (FCLEAR = 0) & (FSTART = 0) THEN PRINT ADI,"WHITE ALARM %p",STRNGO
419              IF (FACK) THEN
420              BEGIN
421              PRINT ADI,"%p ALARM ACKNOWLEDGE",STRNGO
422              LET FACK = 0
423              GOTO NEX
424              END
425              IF (FCLEAR) THEN
426              BEGIN
427              PRINT ADI,"%p ALARM CLEAR",STRNGO
428              LET FCLEAR = 0
429              GOTO NEX
430              END
431              IF (FSTART) THEN
432              BEGIN
433              LET FSTART = 0
434              PRINT ADI,"%p",STRNGO
435              GOTO NEX
436              END
437              COLORO = PTYPE
438              INO = TIMSTR(SSNGO)
439              IF INO != 8 THEN SSNGO = "TIMEFAIL"
440              INO = APPEND(LSGO,STRNGO,SSNGO)

```

```

441 0316 IF INO < 0 THEN LSGO = "APPEND FAILED!!"
442 0317 INO = DATSTR(SDATED)
443 0319 IF INO != 9 THEN SDATED = "DATE FAIL"
444 0320 RETURN
445
446 0321 ; ERRSUB: IF ERRL = ERRNUM() THEN
447 0322 BEGIN
448 0322 ERRL = ERRNUM()
449 0323 ERRL = ERRNUM()
450 0324 PRINT AD1,"ERROR # %I AT LINE %I, iLoop = %I",errn,errL,iLoop
451 0325 FSNO = 1
452 0326 TIMERON STARO
453 0327 END
454 0327 ERSCLR
455 0329 IF ERRL = -187 THEN WAIT 5
456 0330 GOTO AGAIN
457
458 ;
459 ; TERM: LET FABORT = 0
460 0331 TERM: LET STRINGO = "242BATCH TERMINATED "
461 0332 LET ILOOP = STAR + 2
462 0333 LET FSARY = "TERM"
463 0334 LET SSTARO = "TERM"
464 0335 GOSUB ACKK
465 0336 END

```

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 6784 Bytes, 424 Pages
16 Forward Reference(s)

```


PGM1A

```

1      ;          P G M 1 A . S E Q                      VERSION 2
2      ;
3      ;
4      ;          Program to monitor sequence programs running in PCM1
5      ;          PGMSTAT sets bits in an integer to give the status of a program.
6      ;          It indicates if the program is running, is critical, trapped etc.
7      ;          This program fetches this value then, by using a bit mask, it
8      ;          checks specific statuses and converts these to a single integer value
9      ;          used by the string editor. SED translates the integer to strings
10     ;          in graphic 043, titled "PROGRAM STATUS".
11     ;
12     ;          Written By      : R. N. Kersey
13     ;          Date           : 05-31-92
14     ;          Facility       : 242-A Evaporator Bldg.
15     ;          Machine        : VAX 4104-A
16     ;          Op. System     : VAX/VMS 5.5-2AH
17     ;          D/3 Version    : 6.3-2 BASELINE
18     ;          Compiler       : SIC
19     ;          Build command  : @([320,20]PGM1A
20     ;
21     ;          Assoc. UPF:     [300,1]_BLANKK.____ BLANKK.LST
22     ;          Assoc. RPF:     [300,1]_EMPTY.____ EMPTY.LST
23     ;          SDB:           [300,1]PCM1ASDB.CMD  PCM1ASDB.LST
24     ;
25     ;          MODIFICATION LOG:
26     ;
27     ;          29-JUL-93      Added comments and baselined for      M. Teats
28     ;
29     ;
30     ;          15-DEC-94      Revised header to indicate th enew VMS      J. Berger
31     ;          system. Rebase line document.
32     ;
33     ;          END MOD LOG
34     ;
35     ;          UNIT          BLANK1
36     ;
37     ;          PSTAT == PALARM
38     ;          PLIST == PSPARO
39     0000      TIMERCLR STM1
40     0001      TIMERON STM1
41     0002      FABORT = 0
42     0003      ASSIGN PLIST = INTO
43     0004      FOR ILOOP = 0 TO 11
44     0005          PLIST = 0
45     0006          NEXT ILOOP
46     ;
47     0007      AGAIN:  LSG6 = "WAIT"
48     0008          ON ERROR GOSUB ERRSUB
49     0009          IF (FSPARO) THEN
50     0010              IF STMRO > 60 THEN
51     ;                  BEGIN
52     0011                  TIMEROFF STMRO
53     0012                  TIMERCLR STMRO
54     0013                  FSPAR5 = 0
55     0014                  ERRN = 0

```

```

56 0015      ERRL = 0
57      END
58
59 0016      WAIT 1
60      IF FERROR THEN GOTO TERM
61      LSG6 = "RUNNING"
62      WAIT 1
63      TIMERCLR STM1
64      INT6 = PGMSTAT("WHITE1", "WHOLE1")
65      INT7 = PGMSTAT("AMBATCH", "ANFARM")
66      INT8 = PGMSTAT("APBATCH", "ANPFARM")
67      INT9 = PGMSTAT("AMBATCH", "ANFARM")
68      INT10 = PGMSTAT("XYZBATCH", "XYZFRM")
69      INT11 = PGMSTAT("242BATCH", "BLDNG")
70      ASSIGN PLIST = INTO
71      ASSIGN PSTAT = INTO6
72      FTRIPO = 0
73      FOR ILOOP = 0 TO 5
74      BVAL = 1
75      REMAIN = 0
76      FOR INDEX = 1 TO 4
77      IF AND(BVAL, PSTAT) THEN REMAIN = INDEX
78      BVAL = BVAL(1)
79      NEXT INDEX
80      IF PSTAT = -1 THEN
81      BEGIN
82      PLIST = 6
83      FTRIPO = 1
84      END
85      ELSE
86      BEGIN
87      PLIST = REMAIN
88      IF AND(4096, PSTAT) THEN PLIST = 5
89      END
90      TRIP = OR(TRIP, PSTAT)
91      STEP PLIST BY 1
92      STEP PSTAT BY 1
93      NEXT ILOOP
94      IF AND(TRIP, 2) | FTRIPO THEN FTRIPO = 1
95      ELSE FTRIPO = 0
96      GOTO AGAIN
97
98      =====
99      SUBROUTINES
100
101      ERRSUB:  TIMERON STM1
102      0052      TIMERCLR STM1
103      0053      FSPARS = 1
104      0054
105      ; errors are handled differently depending on their severity
106      ;
107      ; 19 PROGRAM ALREADY IN KICK SUBROUTINE (rekick)
108      ; 32 UNDEFINED COMMUNICATIONS PACKET TYPE (ignore)
109      ; 41 PCM DB HEADER NOT LOADED (ignore)
110      ; 69 IPN NOT DEFINED (ignore)

```

WMC-SD-534-CSWL-001 REV 3

Sequence Instruction Compiler Listing

15:36:22 24-Feb-95

Page 3

```

111      ; 162 NO SUCH PROGRAM IN THIS PCM (ignore)
112      ; 165 SEQUENCE PROGRAM ALREADY RUNNING (ignore)
113      ; 170 SERIAL NUMBER MISMATCH (wait forever)
114      ; 179 ILLEGAL SUBROUTINE STACK CLEAR (ignore)
115      ; 187 BACKUP PCM IS DOWN (ignore)
116      ; 193 NO SUCH PROGRAM IN THIS PCM (ignore)
117      ; 198 ILLEGAL SUBROUTINE STACK NESTING ERROR (ignore)
118      ; 199 ILLEGAL ROF NAME (wait forever)
119      ; 200 ROF NOT LOADED IN MEMORY (wait forever)
120      ; 201 RECIPE LOAD NOT ENABLED (wait forever)
121      ; 222 BOARDS DEAD (ignore)
122      ; 226 BOARD ERROR E.G. broken skid panel etc... (ignore)
123      ; 245 BOTH PCMS DOWN, SENT REDUNDANT HIGHWAY ACCESS (ignore)
124      ;
125 0055      ERRN = ABS(ERRNUM())      ; get error number
126 0056      ERRL = ERRLIN()          ; get error line number
127      ;
128      ; if error is not serious, clear and return to program
129      ;
130 0057      IF (ERRN=187) | (ERRN=32) | (ERRN=69) | (ERRN=222) THEN
131 0058          ERSRET
132      ;
133 0059      IF (ERRN=41) | (ERRN=165) | (ERRN=162) | (ERRN=179) THEN
134 0060          ERSRET
135      ;
136 0061      IF (ERRN=226) | (ERRN=198) | (ERRN=245) THEN
137 0062          ERSRET
138      ;
139 0063      IF (ERRN=193) | (ERRN=19) THEN
140 0064          ERSRET
141      ;
142      ; else report the error, and wait for corrective action ....
143 0065      PRINT 0,"NUMBER %i AT LINE %i ****",ERRN,ERRL
144      ;
145 0066      WAITUNTIL(0)              ; wait for engineer corrective
146      ;                             ; action
147 0067      GOTO AGAIN                ; when corrective action taken,
148      ;                             ; goto done and cleanup
149      ;
150      ;=====
151      ;
152      ;
153 0068      TERM:  ASSIGN PLIST = INTO
154 0069          FOR ILOOP = 0 TO 5
155 0070              PLIST = 6
156 0071              STEP PLIST BY 1
157 0072              NEXT ILOOP
158 0073              TIMEROFF STMRO
159 0074              TIMERCLR STMRO
160 0075              TIMEROFF STMRI
161 0076              TIMERCLR STMRI
162      END

```

Total Errors Detected 0 Total Warnings 0
Sequence Program Size = 1456 Bytes, 91 Pages
1 Forward Reference(s)

APPENDIX F - "C" PROGRAMS FOR ALARMS

DOGOA

WHC-SD-534-CSWL-001 REV 3

```

/*      D O G O A . C                               VERSION 2

DOGOA (DOGOA.LTK) runs in micro PCMOA.

DOGOA scans the FALMON and ACK flags on all sequence units in PCMOA.
When FALMON is set by any sequence program, DOGOA turns on audible
alarm device AUDALARMO and resets the FALMON flag to zero. FALMON
is set by the sequence programs if alarms found by sequence programs
are yellow or white.

When the flag ACK is set by any sequence program, DOGOA sets FSLO
(Flag 168) in unit BLANK0, which is scanned every 2 seconds by
WATCH.C (installed as task DOG running in the CDCM). WATCH.C resets
the FSLO flag to zero and spawns task ACKFLG (in the CDCM). Task
ACKFLG acknowledges all alarms in that unit and resets the ACK flag
to zero in the sequence program.

*/
/* Written by : R. N. Kersey                               */
/* Date       : 12-May-92                                   */
/* Modified by: John Hayden T.I. 6.3-2 Upgrade            */
/* Facility   : 242-A Evaporator Bldg.                    */
/* Machine    : VAX 4105-A                                 */
/* Compiler   : DEC VAX C V3.3                             JH01 */
/* VMS DCL .cmd use : SET DEF MICRO:[200,10]              JH01 */
/* C .cmd use : BLD DOGOA DOGOA.C -L -MOV                  JH01 */
/* Op. System : VAX/VMS 5.5-2AH                             JH01 */
/* D/3 Version: 6.3-2                                       JH01 */
/*
/* MODIFICATION LOG:
/*
/* 16-Jul-92      Added description, format changes.      M. Teats
/*
/* 04-Sep-92      Added/increased sleep to prevent M. Teats
/*                  FALMON getting stuck on (mt!).
/*
/* 19-JAN-93      Added 30 tic sleep before put_vseq.      R. Kersey
/*
/* 02-AUG-93      Baseline for D/3 version 3.3              M. Teats
/*
/* 02-JUL-94      Baseline for D/3 version 6.3              J. Hayden
/*
/* 01-SEP-94      Sleeps now seconds instead of ticks      J. Hayden
/*
/* 02-SEP-94      %i now %d, #includes different            J. Hayden
/*
/* 20-SEP-94      Paramaters for put_vseq calls changed     J. Hayden
/*
/* 22-Sep-94      Included RSI fuction to put in infinite
/*                  loop                                     J. Hayden
/*
/* END OF MOD LOG
/*

#include <d3sys.h>
#include <dba.h>
#include <micstd.h>
#include <seqdba.h>
#include <conf.h>
#include <almmsg.h>
#include <err.h>
#include <comm.h>
#include <comcodes.h>
#include <mic.h>
#include <comstat.h>
#include <pcmii.h>

main()
(
char  net, unit, type, field, cmd, pcm;
int   access, k, vprr, alm_on, flg_on, value, select,

```

WHC-SD-534-CSWL-001 REV 3

```

ipn, extra, zero, err, crid, ack, ackflg, index ;

access = UNITREL ; net = 0 ; zero = 0 ; cmd = 1 ;
vprrt = 0 ; alrm_on = 0 ; flg_on = 0 ; value = 0 ; select = 0 ;
ipn = 0 ; extra = 0 ; err = 0 ; ack = 0 ; ackflg = 0 ; index = 0 ;
checkerr(1, mopen());

FOREVER
(
    /*
    RSI(100) ;
    ackflg = 0 ; alrm_on = 0 ;
    for ( k = 1; k < 22; k++ )
    */
    /* 21 Units, skip 19 (ROOMS). 14 handled outside Loop
    {
        index = 364 ;
        /* FALMON offset 204 for UPF
        SHORTO /* if ( k < 5 ) index = 204 ;
        /* FALMON offset 364 for UPF
        TNKFRM /* if ( k != 19 )
        {
            checkerr ( 3, get_vseq ( net, 2, FG, access, k, index, &vprrt ) ;
            /* Get FALMON
            if ( vprrt == 1 )
            {
                alrm_on = 1 ;
                sleep (1) ;
                /*
            added
            checkerr ( 4, put_vseq ( net, 2, FG, access, k, index, &zero ));
            /* Reset FALMON
            Flag /* sleep(1);
            /* added
            (mt1)
            }
            checkerr ( 4, get_vseq ( net, 2, FG, access, k, index+3, &ack ) ;
            /* Get ACK Flag
            /*
            if ( ack == 1 ) ackflg = 1 ;
            sleep(1);
            }
            /* End FOR Loop
            */
        }

        /* If ACKFLG == 1 then set Flag 168 in sequence unit BLANKO in PCMDA
        */
        /* If ALM_ON = 1 then turn on device AUDALARMO in PCMDA
        */

        if ( ackflg == 1 ) checkerr ( 8, put_vseq ( net, 2, FG, access, 25, 168, &ackflg ) ;
        if ( alrm_on == 1 ) checkerr ( 9, devcmd ( 0, 2, DEV, 171, &cmd ) ;
        /* TAKE A REST
        */
        sleep (3) ;
        /* changed from 10 to 100 (mt1)
        */
    }
    /* End FOREVER
    */
)
/* End MAIN
*/

checkerr(x,ierr)
int ierr,x ;
{
    int r2,r3 ;
    if (ierr < 0 )
    {
        r2 = lnkrq("CONFIG");
        r3 = lnkrq("ALMQUE");
        sndmsg ( 0, 0, r3, r2, 3, MSG_TEXT, CRIT," DOGOB.LTK err %d at %d, DOGOB IS TERMINATING!!!!!!\n",ierr,x)
    ;
        sndmsg ( 0, 0, r3, r2, 3, MSG_TEXT, CRIT," DOGOB.LTK err %d at %d, DOGOB IS TERMINATING!!!!!!\n",ierr,x)
    ;
        exit();
    }
}

```

RMIS View/Print Document Cover Sheet

This document was retrieved from the Documentation and Records Management (DRM) ISEARCH System. It is intended for Information only and may not be the most recent or updated version. Contact a Document Service Center (see Hanford Info for locations) if you need additional retrieval information.

Accession #: D196079553

Document #: SD-534-CSWL-001

Title/Desc:

SEQUENCE & BATCH LANGUAGE PROGRAMS & ALARM RELATED
C PROGRAMS FOR THE 242A MCS [SEC 6 OF 6 PAGE 637
TO 731]

Pages: 97

This document was too large to scan as a whole document, therefore it required breaking into smaller sections.

Document number: SD-534-CSWL-001

Section 6 of 6

Title: Sequence and Batch Language programs
and Alarm Related "C" Programs for the
242-A MCS

Date: 4/15/96 Revision: 3

Originator: Berger, J.F
Co: _____

Recipient: _____
Co: _____

References: ECN- 627504

DOGOB

WHC-SD-534-CSWL-001 REV 3

```
/* DOG0B.C
```

VERSION 2

DOG0B (DOG0B.LTX) runs in micro PCMOA.

DOG0B scans the FALMON and ACK flags on all sequence units in PCMOA. When FALMON is set by any sequence program, DOG0B turns on audible alarm device AUDALARM0 and resets the FALMON flag to zero. FALMON is set by the sequence programs if alarms found by sequence programs are yellow or white.

When the flag ACK is set by any sequence program, DOG0B sets FSLO (Flag 168) in unit BLANK0, which is scanned every 2 seconds by WATCH.C (installed as task DOG running in the CDCM). WATCH.C resets the FSLO flag to zero and spawns task ACKFLG (in the CDCM). Task ACKFLG acknowledges all alarms in that unit and resets the ACK flag to zero in the sequence program.

```
/*
/* Written by : R. N. Kersey
/* Date : 12-May-92
/* Facility : 242-A Evaporator Bldg.
/* Modified by: John Hayden T.I. 6.3-2 Upgrade
/* Facility : 242-A Evaporator Bldg.
/* Machine : VAX 4105-A
/* Compiler : DEC VAX C V3.3
/* VMS DCL .cmd use : SET DEF MICRO:[200,10]
/* C .cmd use : BLD DOG0B DOG0B.C -L -MOV
/* Op. System : VAX/VMS 5.5-2AH
/* D/3 Version: 6.3-2
/*
/* MODIFICATION LOG:
/*
/* 16-JUL-92 Added description, format changes. M. Teats
/*
/* 04-Sep-92 Added/increased sleep to prevent M. Teats
/* FALMON getting stuck on (mt1).
/*
/* 19-JAN-93 Added 30 tic sleep before put_vseq. R. Kersey
/*
/* 02-AUG-93 Baseline for D/3 version 3.3 M. Teats
/*
/* 02-JUL-94 Baseline for D/3 version 6.3 J. Hayden
/*
/* 01-SEP-94 Sleeps now seconds instead of ticks J. Hayden
/*
/* 02-SEP-94 %i now %d, #includes different J. Hayden
/*
/* 20-SEP-94 Paramaters for put_vseq calls changed J. Hayden
/*
/* 22-Sep-94 Included RSI fuction to put in infinite
/* loop J. Hayden
/* END OF MOD LOG
```

```
#include <d3sys.h>
#include <dba.h>
#include <micstd.h>
#include <seqdba.h>
#include <conf.h>
#include <almsg.h>
#include <err.h>
#include <comm.h>
#include <comcodes.h>
#include <mic.h>
#include <comstat.h>
#include <pcmii.h>
```

```
main()
{
char net, unit, type, field, cmd ;
int access, k, vprt, alm_on, flg_on, value, select,
```

WHC-SD-534-CSWL-001 REV 3

```

ipn, extra, zero, err, crid, ack, ackflg, index ;

access = UNITREL ; net = 0 ; zero = 0 ; cmd = 1 ;
vprrt = 0 ; alm_on = 0 ; flg_on = 0 ; value = 0 ; select = 0 ;
ipn = 0 ; extra = 0 ; err = 0 ; ack = 0 ; ackflg = 0 ; index = 0 ;
checkerr(1, mopen());

FOREVER
(
    /*
    RSI(100);
    ackflg = 0 ; alm_on = 0 ;
    for ( k = 1; k < 22; k++ )
    */
    /* Infinite Loop
    */
    /* 21 Units, skip 19 (ROOMS). 14 handled outside Loop
    */
    (
        index = 364 ;
        /* FALMON offset 204 for UPF
        */
        SHORTO /* if ( k < 5 ) index = 204 ;
        /* FALMON offset 364 for UPF
        */
        TNKFRM /*
        if ( k != 19 )
        (
            checkerr ( 3, get_vseq ( net, 3, FG, access, k, index, &vprrt ) );
            /* Get FALMON
            */
            if ( vprrt == 1 )
            (
                alm_on = 1 ;
                checkerr ( 4, put_vseq ( net, 3, FG, access, k, index, &zero ));
                /* Reset FALMON
                */
                sleep(1);
                /* added
                */
            )
            checkerr ( 4, get_vseq ( net, 3, FG, access, k, index+3, &ack ) );
            /* Get ACK Flag
            */
            if ( ack == 1 ) ackflg = 1 ;
            sleep(1);
        )
        /* End FOR Loop
        */
    )
    /* If ACKFLG = 1 then set Flag 168 in sequence unit BLANKO in PCMOA
    */
    /* If ALM_ON = 1 then turn on device AUDALARMO in PCMOA
    */
    if ( ackflg == 1 ) checkerr ( 8, put_vseq ( net, 3, FG, access, 25, 168, &ackflg ) );
    if ( alm_on == 1 ) checkerr ( 9, devCmd ( 0, 3, DEV, 171, &cmd ) );

    /* TAKE A REST
    */
    sleep (3) ;
    /* changed from 10 to 100 (mt1)
    */
)
/* End FOREVER
*/
/* End MAIN
*/

checkerr(x,ierr)
int ierr,x ;
{
    int r2,r3 ;
    if (ierr < 0 )
    {
        r2 = lnkrq("CONFIG");
        r3 = lnkrq("ALMQUE");
        sndmsg ( 0, 0, r3, r2, 3, MSG_TEXT, CRIT," DOGOB.LTK err %d at %d, DOGOB IS TERMINATING!!!!\n",ierr,x)
        ;
        sndmsg ( 0, 0, r3, r2, 3, MSG_TEXT, CRIT," DOGOB.LTK err %d at %d, DOGOB IS TERMINATING!!!!\n",ierr,x)
        ;
        exit();
    }
}

```

WHC-SD-534-CSWL-001 REV 3

DOG1A

WHC-SD-534-CSWL-001 REV 3

/* DOG1A.C

VERSION 2

DOG1A (DOG1A.LTK) runs in micro PCM1A.

DOG1A scans the FALMON and FLGACK flags on all sequence units in PCM1A. When FALMON is set by any sequence program, DOG1A turns on audible alarm device AUDALARMO and resets the FALMON flag to zero. FALMON is set by the sequence programs if alarms found by sequence programs are yellow or white.

When the flag FLGACK is set by any sequence program, DOG1A sets FSLO (Flag 168) in unit BLANK1, which is scanned every 2 seconds by WATCH.C (installed as task DOG running in the CDCM). WATCH.C resets the FSLO flag to zero and spawns task ACKFLG (in the CDCM). Task ACKFLG acknowledges all alarms in that unit and resets the FLGACK flag to zero in the sequence program.

```

/*
/* Written by : R. N. Kersey          */
/* Date       : 27-May-92             */
/* Modified by: John Hayden T.I.      6.3-2 Upgrade */
/* Facility   : 242-A Evaporator Bldg. */
/* Machine    : VAX 4104-A           */
/* Compiler   : DEC VAX C V3.3        JH01 */
/* VMS DCL .cmd use : SET DEF MICRO:[200,10] JH01 */
/* C .cmd use : BLD DOG1A DOG1A.C -L -MOV JH01 */
/* Op. System : VAX/VMS 5.5-2AH      JH01 */
/* D/3 Version: 6.3-2                JH01 */
/*
/*
/*
/* MODIFICATION LOG:
/*
/* 13-Jul-92      Removed LNKRG('CONFIG') after mopen.      M. Teats */
/*                Reformatted brackets.                     */
/*
/* 16-Jul-92      Added description, format changes.        M. Teats */
/*
/* 21-Jul-92      Implemented micro select feature.          M. Teats */
/*                Changed 'err = ' to 'checkerr'.            */
/*                Added CHECKERR subroutine.                 */
/*                Deleted CONTINUE if err < 0.              */
/*                Changed 'cmd' from int to char to allow    */
/*                DEVCMO to turn on horn.                   */
/*                Initialize 'cmd = ' at program start and   */
/*                removed 'cmd = ' from IF #8 when           */
/*                AUDALARMO turned on.                       */
/*                Rewrote last half of code.                 */
/*
/* 12-FEB-93      - Added sleep as noted before and          M. Teats */
/*                after put_vseq                             */
/*                - Removed phasing (& put_vseq) on          */
/*                FHOLD (done by sequence programs)          */
/*                These are to prevent program lock-up.      */
/*
/* 02-AUG-93      Baseline for D/3 version 3.3               */
/*
/* 03-JUL-93      Baseline for D/3 version 6.3-2            J. Hayden */
/*
/* 01-SEP-94      Sleeps now seconds instead of ticks        J. Hayden */
/*
/* 02-SEP-94      %i now %d, #includes different             J. Hayden */
/*
/* 20-SEP-94      Parameters for put_vseq calls changed      J. Hayden */
/*
/* 22-Sep-94      Included RSI fuction to put in infinite    J. Hayden */
/*                loop                                        */
/*
/* END OF MOD LOG
/*

```

```

#include <d3sys.h>
#include <dba.h>

```

WHC-SD-534-CSWL-001 REV 3

```

#include <micstd.h>
#include <seqdba.h>
#include <conf.h>
#include <almmshg.h>
#include <err.h>
#include <comm.h>
#include <comcodes.h>
#include <mic.h>
#include <comstat.h>
#include <pcmi1.h>

main()
(
char    net, unit, type, field, cmd ;
int     access, k, vpvt, alm_on, one, select,
        flg_on, value, ipn, extra, phase,
        zero, err, crid, total ;

access = UNITREL ; net = 0 ; zero = 0 ; phase = 0 ; one = 1 ; cmd = 1 ;
vpvt = 0 ; alm_on = 0 ; select = 0 ; flg_on = 0 ; value = 0 ; ipn = 0 ;
extra = 0 ; err = 0 ; total = 0 ;
checkerr (1, mopen());

FOREVER
(
    RSI(100) ;
    alm_on = 0 ;
    total = 0 ;

/*      A L A R M :
/*      Get FALMON Flag 492, then reset it to zero:
for ( k = 1 ; k < 6 ; k++ )
(
    checkerr ( 3, get_vseq ( 0, 4, FG, access, k, 492, &vpvt ) ) ;
    if (vpvt==1)
    (
        alm_on = 1 ;
        checkerr ( 4, put_vseq ( 0, 4, FG, access, k, 492, &zero ) ) ;
        sleep (1) ;
        sndmsg(0,0,-1,crid,1,MSG_TEXT,NONCRIT,"D O G 1 A . C - FALMON FLAG SET ON UNIT %d\n",k) ;
    )

/*      A C K N O W L E D G E :
/*      Get FLGACK Flag 495, then reset it to zero:
checkerr ( 5, get_vseq ( 0, 4, FG, access, k, 495, &vpvt ) ) ;
if (vpvt == 1 )
(
    total = 1 ;
    if ( k != 5 )
    (
        checkerr ( 6, put_vseq ( 0, 4, FG, access, k, 495, &zero ) ) ;
        sleep (1) ;
    )
)

/* If FALMON = 1, then turn on device AUDALARMO in PCMOA:
if (alm_on == 1 )
(
    checkerr ( 7, devcmd ( 0, 2, DEV, 171, &cmd ) ) ;

/* If FLGACK = 1, then set Flag 168 (FSLO) in unit BLANK1 in PCMI1A:
if (total == 1 ) checkerr ( 8, put_vseq ( 0, 4, FG, access, 6, 168, &total ) ) ;

/* Phase the FHOLD flag 479 to turn off Lamp:
/* if (phase == 1) put_vseq ( 0, 4, FG, access, 1, 479, &one ) ;
/* if (phase == 7) put_vseq ( 0, 4, FG, access, 2, 479, &one ) ;
/* if (phase == 14) put_vseq ( 0, 4, FG, access, 3, 479, &one ) ;

```

WHC-SD-534-CSWL-001 REV 3

```

/*      if (phase == 21) put_vseq ( 0, 4, FG, access, 4, 479, &one ) ;      deleted      */
/*      phase++ ; if (phase >= 22) phase = 1 ;      deleted      */

    )      sleep (3) ;      /* en2, End FOREVER      */
)      /* en1, End main      */

/* CHECKERR SUBROUTINE:      */
checkerr(x,ierr)
int ierr,x ;
{
    int r2,r3 ;
    if (ierr < 0 )
    {
        r2 = lnkrq("CONFIG");
        r3 = lnkrq("ALMOQUE");
        sndmsg ( 0, 0, r3, r2, 3, MSG_TEXT, CRIT," DOG1A.LTK err %d at %d\n",ierr,x) ;
        exit();
    }
}

```


WHC-SD-534-CSWL-001 REV 3

D0618

WHC-SD-534-CSWL-001 REV 3

```
/* DOG1B.C
```

```
VERSION 2
```

```
DOG1b (DOG1b.LTK) runs in micro PCM1b.
```

```
DOG1b scans the FALMON and FLGACK flags on all sequence units in PCM1b.
When FALMON is set by any sequence program, DOG1b turns on audible
alarm device AUDALARMO and resets the FALMON flag to zero. FALMON
is set by the sequence programs if alarms found by sequence programs
are yellow or white.
```

```
When the flag FLGACK is set by any sequence program, DOG1b sets FSL0
(flag 168) in unit BLANK1, which is scanned every 2 seconds by
WATCH.C (installed as task DOG running in the CDCM). WATCH.C resets
the FSL0 flag to zero and spawns task ACKFLG (in the CDCM). Task
ACKFLG acknowledges all alarms in that unit and resets the FLGACK flag
to zero in the sequence program.
```

```

/*
/* Written by : R. N. Kersey
/* Date : 27-May-92
/* Modified by: John Hayden T.I. 6.3-2 Upgrade
/* Facility : 242-A Evaporator Bldg.
/* Machine : VAX 4104-A
/* Compiler : DEC VAX C V3.3 JH01
/* VMS COMMAND use : SET DEF MICRO:[200,10] JH01
/* C.cmd use : BLD DOG1B DOG1B.C -L -MOV JH01
/* Op. System : VAX/VMS 5.5-2AH JH01
/* D/3 Version: 6.3-2
/*
/*
/* MODIFICATION LOG:
/*
/* 13-Jul-92 Removed LNKRG('CONFIG') after mopen. M. Teats
/* Reformatted brackets.
/*
/* 16-Jul-92 Added description, format changes. M. Teats
/*
/* 21-Jul-92 Implemented micro select feature. M. Teats
/* Changed 'err = ' to 'checkerr'.
/* Added CHECKERR subroutine.
/* Deleted CONTINUE if err < 0.
/* Changed 'cmd' from int to char to allow
/* DEVCHD to turn on horn.
/* Initialize 'cmd = 1' at program start and
/* removed 'cmd = 1' from IF #8 when
/* AUDALARMO turned on.
/* Rewrote last half of code.
/*
/* 12-FEB-93 - Added sleep as noted before and M. Teats
/* after put_vseq
/* - Removed phasing (& put_vseq) on
/* FHOLD (done by sequence programs)
/* These are to prevent program lock-up.
/*
/* 02-AUG-93 Baseline for D/3 version 3.3
/*
/* 02-JUL-94 Baseline for D/3 version 6.3-2 J. Hayden
/*
/* 01-SEP-94 Sleeps now seconds instead of ticks J. Hayden
/*
/* 02-SEP-94 %i now %d, #includes different J. Hayden
/*
/* 20-SEP-94 Parameters for put_vseq calls changed J. Hayden
/*
/* 22-Sep-94 Included RSI fuction to put in infinite
/* loop J. Hayden
/*
/* END OF MOD LOG
/*
#include <cd3sys.h>
#include <cdba.h>

```

```

#include <micstd.h>
#include <seqdba.h>
#include <conf.h>
#include <almmsg.h>
#include <err.h>
#include <comm.h>
#include <comcodes.h>
#include <mic.h>
#include <comstat.h>
#include <pcmil.h>

main()
(
char net, unit, type, field, cmd ;
int access, k, vpvt, alm_on, one, select,
fig_on, value, ipn, extra, phase,
zero, err, crid, total ;

access = UNITREL ; net = 0 ; zero = 0 ; phase = 0 ; one = 1 ; cmd = 1 ;
vpvt = 0 ; alm_on = 0 ; select = 0 ; fig_on = 0 ; value = 0 ; ipn = 0 ;
extra = 0 ; phase = 0 ; err = 0 ; total = 0 ;
checkerr (1, mopen());

FOREVER
(
RSI(100) ;
alm_on = 0 ;
total = 0 ;

/* A L A R M :
/* Get FALMON Flag 492, then reset it to zero:
for ( k = 1 ; k < 6 ; k++ )
(
checkerr ( 3, get_vseq ( 0, 5, FG, access, k, 492, &vpvt )) ;
if (vpvt==1)
(
alm_on = 1 ;
checkerr ( 4, put_vseq ( 0, 5, FG, access, k, 492, &zero )) ;
sleep (1) ;
sndmsg(0,0,-1,crid,1,MSG_TEXT,NONCRIT,"D O G 1 A . C - FALMON FLAG SET ON UNIT %d\n",k) ;
)
)

/* A C K N O W L E D G E :
/* Get FLGACK Flag 495, then reset it to zero:
checkerr ( 5, get_vseq ( 0, 5, FG, access, k, 495, &vpvt )) ;
if (vpvt == 1)
(
total = 1 ;
if ( k == 5 )
(
checkerr ( 6, put_vseq ( 0, 5, FG, access, k, 495, &zero )) ;
sleep (1) ;
)
)
)

/* If FALMON = 1, then turn on device AUDALARM0 in PCMDA:
if (alm_on == 1)
(
checkerr ( 7, devcmd ( 0, 2, DEV, 171, &cmd )) ;
)

/* If FLGACK = 1, then set Flag 168 (FSLO) in unit BLANK1 in PCMIb:
if (total == 1) checkerr ( 8, put_vseq ( 0, 5, FG, access, 6, 168, &total )) ;

/* Phase the FHOOLD flag 479 to turn off Lamp:
if (phase == 1) put_vseq ( 0, 4, FG, access, 1, 479, &one ) ;
if (phase == 7) put_vseq ( 0, 4, FG, access, 2, 479, &one ) ;
if (phase == 14) put_vseq ( 0, 4, FG, access, 3, 479, &one ) ;
)

```

WHC-SD-534-CSWL-001 REV 3

```

/*      if (phase == 21) put_vseq ( 0, 4, FG, access, 4, 479, &one ) ;      deleted      */
/*      phase++ ; if (phase >= 22) phase = 1 ;      deleted      */

    sleep (3) ;                                /* en2, End FOREVER      */
)                                              /* en1, End main        */

/* CHECKERR SUBROUTINE:      */
checkerr(x,ierr)
int ierr,x ;
{
    int r2,r3 ;
    if (ierr < 0 )
    {
        r2 = lnkrq("CONFIG");
        r3 = lnkrq("ALMQUE");
        sndmsg ( 0, 0, r3, r2, 3, MSG_TEXT, CRIT," DOG1b.LTK err %d at %d\n",ierr,x) ;
        exit();
    }
}

```

WHC-SD-534-CSWL-001 REV 3

WATCH

PAGE 648

WHC-SD-534-CSWL-001 REV 3

```

/*      WATCH.C      VERSION 2      */
/*      Written by : R. N. Kersey      */
/*      Date       : 12-May-92      */
/*      Facility   : 242-A Evaporator Bldg.      */
/*      Machine    : VAX 4104-A      */
/*      Op. System  : VAX/VMS 5.5-2AH      */
/*      D/3 Version : 6.3-1 BASELINE      */
/*      Compiler   : VAX C      */
/*      C .cmd use : BLD -L -ALL -MOV -LIS WATCH.SCR      */

```

```

/*      Program to acknowledge, from a graphic, all alarms in that graphic (and
/*      the corresponding sequence unit - since there is one sequence unit per
/*      graphic). Resets flag FSLO to zero in Sequence Units BLANK0 and
/*      BLANK1, and runs function ACKFLG().

```

```

/*      How WATCH.C works: WATCH.C is installed as task DOG in the CDCM.
/*      It works in conjunction with DOG0A and DOG1A running in the micros.
/*      These three programs communicate via sequence variable flags in the
/*      various sequence units to turn on device AUDALARMO for yellow and
/*      white alarms and acknowledge (from a single graphic) all analog and
/*      digital alarms in that graphic.

```

```

/*      Each sequence program sets flag FALMON to 1 when a new yellow or white
/*      alarm occurs. DOG0A and DOG1A scan the sequence units for FALMON flags
/*      that are set to 1. When the flag is found, DOG0A or DOG1A resets
/*      FALMON to zero. Any FALMON flag set to 1 causes AUDALARMO to be
/*      commanded on, which turns on the yellow/white audible alarm horn.

```

```

/*      Also, DOG0A looks at flag ACK (in PCMO sequence units) and DOG1A looks
/*      at flag FLGACK (in PCM1 sequence units). If ACK is set to 1, DOG0A
/*      sets flag FSLO in unit BLANK0. If FLGACK is set to 1, DOG1A sets flag
/*      FSLO in unit BLANK1. WATCH scans the FSLO flags in these two units.
/*      If FSLO is set to 1, WATCH resets them to zero and runs function
/*      ACKFLG() contained within the WATCH.C program. This function looks
/*      at all the sequence units, finds the ACK and FLGACK flags, resets
/*      them to zero, and acknowledges all EPNs only in the unit corresponding
/*      to the currently displayed graphic.

```

```

/*      NOTE: The redundant program for DOG0A in PCMOB is DOG0B.C, and
/*      for DOG1A in PCM1B is DOG1B.C

```

```

MODIFICATION LOG:
*/
*/ 17-JUL-92 Added description, flow chart, M. Teats */
*/ reformatted. */
*/
*/ 20-JUL-92 Commented-out code (abandoned) to M. Teats */
*/ spawn ALMON (which is unused). */
*/ AUDALARM0 is now turned on by */
*/ DOGGA and DOG1A. See also ALMON.C. */
*/
*/ 15-DEC-92 Changed for AUDALARM0 now being M. Teats */
*/ turned off by SILENCE Key. */
*/
*/ 14-JUL-93 Commented. M. Teats */
*/
*/ 03-AUG-93 Baseline for D/3 Version 3.3 M. Teats */
*/
*/ 03-JAN-95 Combined the program ACKFLG.C as J. Berger */
*/ a function within WATCH.C. Also */
*/ rebaselined the documentation after */
*/ the up grade. Added PDI-CA1-3 to */
*/ ACKFLG() function matrix. */
*/
*/ 26-MAR-95 Added PC Recycle EPNs to Group Ack */
*/ per EE-95-00036. */
*/
*/ END OF MOD LOG
*/

```

```
#include <emcstd.h>
#include <d3sys.h>
#include <dba.h>
#include <seqdba.h>
#include <almsg.h>
#include <recipe.h>
```

```
main()
{
void ackflg(void) ;

char net, unit, type, field ;
int access, ipn, alarm, flg_on, value, ipn, extra, zero, err ;
int pcm0, pcm1, crid, rid ;
access = UNITREL ; net = 0 ; zero = 0 ; pcm0 = 0 ; pcm1 = 0 ; vpvt = 0 ;
err = mopen() ;
```

```
FOREVER
(
    alm_on = 0 ;
    err = get_vseq ( net, 2, FG, access, 25, 168, &pcm0 ) ;
        if ( err < 0 ) continue ;

    err = get_vseq ( net, 4, FG, access, 6, 168, &pcm1 ) :
```

WHC-SD-534-CSWL-001 REV 3

```

if (err < 0 ) continue ;

if ( pcm0==1 || pcm1==1 ) ackflg();

if (pcm0 == 1 ) err = put_vseq ( net, 2, FG, access, 25, 168, &zero ) ;

/* Reset flags */
if (pcm1 == 1 ) err = put_vseq ( net, 4, FG, access, 6, 168, &zero ) ;

err = get_vseq ( net, 4, FG, access, 6, 181, &vprrt ) ;
err = put_vseq ( net, 2, FG, access, 25, 183, &vprrt ) ;
/* Flag FTRIP2, Unit(25) Blank0 */

/*
*/
/* ABANDONED CODE (may be used for testing) TO TURN ON AUDALARM0:
*/
/*
*/
/* err = get_vseq ( net, 2, FG, 25, access, 184, &extra ) ; Flag FTRIP3, Unit (25) Blank0
*/
/* if ( extra == 1)
*/
/* < Test for AUDALARM0 using FTRIP3
in SSD */
/* err = put_vseq ( net, 2, FG, 25, access, 184, &zero ) ; Reset FTRIP3
*/
/*
*/
/* END ABANDONED CODE
*/
/*
*/

sleep (2) ;

)

/* End FOREVER */

)
/* End Main */

void ackflg(void)
(
    struct s1
    (
        char pcm;
        int sunit,
            index,
            bloks,
            offset;
    );

    static struct s1 seqrec [] =
    (
        /* k      struct s1 seqrec []      seq unit name      */
        /* --      -----      -----      */
        /* 0 */      2, 1, 207, 2, 0, /* A244 */
        /* 1 */      2, 2, 207, 1, 2, /* HVC244 */
        /* 2 */      2, 3, 207, 2, 3, /* WASTE */
        /* 3 */      2, 4, 207, 1, 5, /* A350 */
        /* 4 */      2, 5, 367, 1, 6, /* FEED */
        /* 5 */      2, 6, 367, 1, 7, /* CA1 */
        /* 6 */      2, 7, 367, 1, 8, /* PB1 */
        /* 7 */      2, 8, 367, 1, 9, /* PB2 */
        /* 8 */      2, 9, 367, 1, 10, /* EA1 */
        /* 9 */      2, 10, 367, 1, 11, /* EC1 */
        /* 10 */      2, 11, 367, 2, 12, /* RWATER */
        /* 11 */      2, 12, 367, 1, 14, /* URW */
        /* 12 */      2, 13, 367, 2, 15, /* VVENT
    
```


WHC-SD-534-CSWL-001 REV 3

```

/* 13 */      2, 14, 367, 1, 17, /* PC */
/* 14 */      2, 15, 367, 1, 18, /* IX */
/* 15 */      2, 16, 367, 1, 19, /* SC */
/* 16 */      2, 17, 367, 1, 20, /* BASINS */
/* 17 */      2, 18, 367, 1, 21, /* UTIL */
/* 18 */      2, 20, 367, 1, 22, /* AMU */
/* 19 */      2, 21, 367, 1, 23, /* LERF */
/* 20 */      4, 1, 447, 1, 24, /* ANFARM */
/* 21 */      4, 1, 450, 1, 25, /* ANFARM - ANHVC */
/* 22 */      4, 1, 449, 1, 26, /* ANFARM - ANVLV */
/* 23 */      4, 1, 451, 1, 27, /* ANFARM - AN271 */
/* 24 */      4, 2, 447, 2, 28, /* APFARM */
/* 25 */      4, 2, 450, 2, 30, /* APFARM - APHVC */
/* 26 */      4, 2, 449, 2, 31, /* APFARM - APVLV */
/* 27 */      4, 2, 451, 2, 34, /* APFARM - AP271 */
/* 28 */      4, 2, 452, 2, 36, /* APFARM - AP102 */
/* 29 */      4, 3, 447, 1, 40, /* AWFARM */
/* 30 */      4, 3, 450, 2, 41, /* AWFARM - AWHVC */
/* 31 */      4, 3, 449, 2, 43, /* AWFARM - AWWLV */
/* 32 */      4, 3, 451, 3, 45, /* AWFARM - AW271 */
/* 33 */      4, 4, 447, 1, 48, /* XYZFRM */
/* 34 */      4, 4, 448, 2, 49, /* XYZFRM - AFRM */
/* 35 */      4, 4, 449, 2, 51, /* XYZFRM - AYFRM */
/* 36 */      4, 4, 450, 2, 53, /* XYZFRM - AY, AZFRM */
/* 37 */      4, 5, 495, 2, 55, /* BLDNG -mct3+ */
    };

    struct s2
    {
        char pcm;
        int fstdIN,
            totDIN,
            fstAI,
            totAI,
            fstDEV,
            totDEV;
    };

    static struct s2 detblk [] =
    {
/* Added fstdEV and totDEV columns to struct s2 - */
/* 0 */      2, 0, 15, 0, 11, 0, 0, /* A244 */
/* 1 */      2, 17, 3, 0, 0, 0, 0, /* A244 */
/* 2 */      2, 20, 5, 11, 3, 0, 0, /* HVC244 */
/* 3 */      2, 25, 18, 0, 0, 0, 0, /* WASTE */
/* 4 */      4, 243, 1, 0, 0, 0, 0, /* WASTE -mct3+ */
/* 5 */      2, 44, 3, 14, 7, 0, 0, /* A350 */
/* 6 */      2, 50, 7, 30, 10, 0, 0, /* FEED */
/* 7 */      2, 60, 4, 45, 30, 0, 0, /* CA1 */
/* 8 */      2, 70, 13, 80, 18, 0, 0, /* PB1 -jfb3+ */
/* 9 */      2, 85, 17, 100, 15, 61, 1, /* PB2 -mct1+ */
/* 10 */      2, 105, 3, 120, 21, 0, 0, /* EA1 -mct1+ */
/* 11 */      2, 110, 5, 150, 24, 0, 0, /* EC1 -mct1+ */
/* 12 */      2, 120, 9, 180, 5, 0, 0, /* RWATER */
/* 13 */      4, 242, 1, 0, 0, 0, 0, /* RWATER */
/* 14 */      2, 130, 4, 190, 11, 102, 1, /* URW -mct2+ */
/* 15 */      2, 140, 13, 205, 16, 0, 0, /* VVENT */
/* 16 */      2, 154, 3, 0, 0, 0, 0, /* VVENT */
/* 17 */      2, 160, 6, 225, 15, 0, 0, /* PC */
/* 18 */      2, 170, 14, 240, 21, 133, 1, /* IX -mct2+ */
/* 19 */      2, 190, 13, 265, 11, 143, 1, /* SC -mct2+ */
/* 20 */      2, 210, 17, 280, 2, 151, 1, /* BASINS -mct2+ */
/* 21 */      2, 230, 11, 290, 4, 0, 0, /* UTIL */
/* 22 */      2, 260, 6, 310, 14, 0, 0, /* AMU */
/* 23 */      2, 270, 16, 330, 7, 0, 0, /* LERF */
/* 24 */      4, 0, 21, 0, 0, 0, 0, /* ANFARM */
/* 25 */      4, 21, 8, 0, 0, 0, 0, /* ANFARM - ANHVC */
/* 26 */      4, 29, 8, 0, 0, 0, 0, /* ANFARM - ANVLV */
/* 27 */      4, 37, 8, 0, 0, 0, 0, /* ANFARM - AN271 */
    };

```

WMC-SD-534-CSWL-001 REV 3

```

/* 28 */      4, 60, 3, 0, 0, 0, 0, /* APFARM */
/* 29 */      4, 64, 25, 0, 0, 0, 0, /* APFARM */
/* 30 */      4, 114, 13, 0, 0, 0, 0, /* APFARM - APHVC */
/* 31 */      4, 111, 3, 0, 0, 0, 0, /* APFARM - APVLV */
/* 32 */      4, 127, 5, 0, 0, 0, 0, /* APFARM - APVLV */
/* 33 */      4, 136, 6, 0, 0, 0, 0, /* APFARM - APVLV */
/* 34 */      4, 98, 13, 0, 0, 0, 0, /* APFARM - AP271 */
/* 35 */      4, 134, 2, 0, 0, 0, 0, /* APFARM - AP271 */
/* 36 */      4, 63, 1, 0, 0, 0, 0, /* APFARM - AP102 */
/* 37 */      4, 89, 9, 0, 0, 0, 0, /* APFARM - AP102 */
/* 38 */      4, 132, 2, 0, 0, 0, 0, /* APFARM - AP102 */
/* 39 */      4, 142, 1, 0, 0, 0, 0, /* APFARM - AP102 */
/* 40 */      4, 150, 17, 0, 0, 0, 0, /* AMFARM */
/* 41 */      4, 181, 1, 0, 0, 0, 0, /* AMFARM - AWHVC */
/* 42 */      4, 186, 3, 0, 0, 0, 0, /* AMFARM - AWHVC */
/* 43 */      4, 167, 11, 0, 0, 0, 0, /* AMFARM - AWHVLV */
/* 44 */      4, 179, 2, 0, 0, 0, 0, /* AMFARM - AWHVLV */
/* 45 */      4, 178, 1, 0, 0, 0, 0, /* AMFARM - AW271 */
/* 46 */      4, 189, 5, 0, 0, 0, 0, /* AMFARM - AW271 */
/* 47 */      4, 195, 2, 0, 0, 0, 0, /* AMFARM - AW271 */
/* 48 */      4, 229, 10, 0, 0, 0, 0, /* XYZFRM */
/* 49 */      4, 214, 6, 0, 0, 0, 0, /* XYZFRM - AFRM */
/* 50 */      2, 15, 2, 0, 0, 0, 0, /* A244 - AFRM */
/* 51 */      4, 222, 4, 0, 0, 0, 0, /* XYZFRM - AYFRM */
/* 52 */      4, 228, 1, 0, 0, 0, 0, /* XYZFRM - AYFRM */
/* 53 */      4, 220, 2, 0, 0, 0, 0, /* XYZFRM - AY,AZ */
/* 54 */      4, 226, 2, 0, 0, 0, 0, /* XYZFRM - AY,AZ */
/* 55 */      4, 250, 28, 80, 12, 0, 0, /* BLDNG */
/* 56 */      2, 0, 0, 300, 3, 0, 0, /* BLDNG */

```

{

```

    char net, unit;
    int access, almreg, conreg, dbreg, j, k, vpert, extra, value;

    access = UNITREL;
    net = 0; almreg = 0; conreg = 0; dbreg = 0; j = 0;
    k = 0; vpert = 0; extra = 0; value = 0;
    /* opens network for communication */
    checkerr (1, mopen());
    for (k = 0; k < 38; k++)
    {
        /* get value from Sequence Database to check if flag is set */
        checkipn ( k, 2, seqrec[k].pcm, get_vseq(net,seqrec[k].pcm,FG,access,
            seqrec[k].sunIt, seqrec[k].index,&vpert) );
        if(vpert==1)
        {
            vpert = 0;
            checkerr (3, put_vseq(net,seqrec[k].pcm,FG,access,
                seqrec[k].sunIt,seqrec[k].index,&vpert) );
            checkerr (4, dbreg = lnrg ("DBASE" ));
            checkerr (5, almreg = lnrg ("ALMOQUE" ));
            checkerr (6, conreg = lnrg ("CONFIGN" ));

            for ( j = seqrec[k].offset;
                j < seqrec[k].offset + seqrec[k].bloks;
                j++)
            {
                if (datblk[j].totDIN > 0 )
                    acknow (net, datblk[j].pcm, datblk[j].fstDIN,
                        datblk[j].totDIN, DIN, DIN_ALMCT,
                        almreg, conreg, dbreg);
                if (datblk[j].totAI > 0 )
                    acknow (net, datblk[j].pcm, datblk[j].fstAI,
                        datblk[j].totAI, AI, AI_ALMCT,
                        almreg, conreg, dbreg);

                if (datblk[j].totDEV > 0 )
                    acknow (net, datblk[j].pcm, datblk[j].fstDEV,

```

WHC-SD-534-CSWL-001 REV 3

```

        datblk[j].totDEV, DEV, DEV_ALMCT,
        almreg, conreg, dbreg);

    } /* end for */

    checkerr ( 11, conreg = unlrg ( conreg ) );
    checkerr ( 12, almreg = unlrg ( almreg ) );
    checkerr ( 13, dbreg = unlrg ( dbreg ) );

} /* end if */

) /* end for loop */
    checkerr (14, get_vseq(net,4,FG,access, 6, 181, &vprt) );
    checkerr (15, put_vseq(net,2,FG,access,25, 183, &vprt) ); /* +mct2- */
    checkerr (16, get_vseq(net,2,FG,access,25, 184, &extra) ); /* +mct2- */
    if ( extra == 1 )
    {
        value = 1 ;
        extra = 0 ;
        checkerr ( 4, devcmd( net, 2, DEV, 171, &value)); /* AUDALARMO, +mct2- */
        checkerr (17, put_vseq(net,2,FG,access,25, 184, &extra) ); /* +mct2- */
    }
} /* end main */
}
/*****
/* function acknow */
*****/
acknow (net, pcm, first, last, type, field, almreg, conreg, dbreg)

    char field, net, pcm, type;
    int almreg, conreg, dbreg, first, last;

(
    char buff[MAX_DB_RECSIZ],
    dbuff[MAX_DB_RECSIZ],
    epn[EPNSIZ + 1];
    int adi, devunt, ipn, tot_val, temp_val, err;

    devunt = 4;

/* loop for ipns */
for (ipn = first; ipn < first + last; ipn++)
(
    /* transfers every field in one PCMDB into buffer */
    err = grb(net,pcm,type,ipn,buff);
    if ( err < 0 ) continue ;

    /* transfers one field from buffer into user location */
    checkerr (8, gfb(type,field,&tot_val,buff));

    temp_val = (tot_val & 15);
    if (temp_val > 0)
    {
        if ( ! ( tot_val & 32 ) )
        {
            /* transfers an acknowledgement */
            checkipn (9,ipn, pcm, dback(net,pcm,type,ipn));

            /* identifies epn */
            checkerr (10, idepn(dbreg,epn,net,pcm,type,ipn));
            sndmsg (net,devunt,almreg,conreg,adi,MSG TEXT,
                NONCRIT,"\\t\\t%s ALARM ACKNOWLEDGED",epn);

        } /* end if */
    } /* end if */
} /* end for loop */

```

```

) /* end function acknow */

/*****
/* function checkerr */
*****/
checkerr(x,ierr)
    int ierr[],x ;
    {
        int r2,r3 ;
        if (ierr < 0 )
        {
            printf("\n checkerr problem on ACKFLG.C at %d, error = %d\n",x,ierr);
/*            perror(ierr); */
            r2 = lnkrq("CONFIG");
            r3 = lnkrq("ALMQUE");
/*            sndmsg(0,0,r3,r2,3,MSG_TEXT,NOCRIT,
"\n ALARM ACKNOWLEDGE PROGRAM --- ERROR ROUTINE FAILURE AT %d, ERROR %d\n",
                x,ierr); */
            exit();
        }
    }

checkipn(x,y,p,ierr) /* number, ipn, error */
    int x, y, p, ierr ;
    {
        int r2,r3 ;
        if (ierr < 0 )
        {
            printf("\n checkipn problem on ackflg.c at %d, error = %d, ipn = %d, pcm %d\n",x,ierr,y,p);
/*            perror(ierr); */
            r2 = lnkrq("CONFIG");
            r3 = lnkrq("ALMQUE");
/*            sndmsg(0,0,r3,r2,3,MSG_TEXT,NOCRIT,
"\n ALARM ACKNOWLEDGE PROGRAM --- ERROR ROUTINE FAILURE AT %d, ERROR %d\n",
                x,ierr); */
            exit();
        }
    }

```

INHA

WHC-SD-534-CSWL-001 REV 3

```

/*          I N H A . C                      VERSION 3
*
* Alarm Inhibit program for graphics 80, 81, 82, 83 and 84. Scans
* EPNs and lists all inhibited EPNs and descriptors. Batch variables
* FTRIP0, FTRIP1, FTRIP2, FTRIP3 display *MORE on GRA080, 81, 82, 83 when
* more than 20 found on the page. FSTOP displays TOO MANY INHIBITED
* ALARMS!.... when more than 96 inhibited alarms are found. FSTOP also
* terminates program. INHA.C is to be run with sequence program
* TIMEINHA.SEQ in unit WHOLED which fetches the correct time and date.
* INHA.C is installed as task INHALM and runs from skid key 53.
*/

/* Written by : R. M. Kersey                */
/* Date       : 11-SEP-91                   */
/* Facility   : 242-A Evaporator Bldg.      */
/* Machine    : VAX 4000-105                */
/* D/3 Version: 6.3                         */
/* Compiler   : VAX C                       */
/* Op. System : VMS 5                       */

/* MODIFICATION LOG:                        */
/*
/* 20-JUL-92      Added description, header.      M. Teats */
/*
/* 22-JUL-93      Baselined for D/3 Version 3.3.   M. Teats */
/*
/* 26-MAR-95      Changed struct S1 so that all    J. Berger */
/*                  EPNs will get displayed. Per    EE-95-00036. */
/*
/* END OF MOD LOG
*/

#include <emcstd.h>
#include <d3sys.h>
#include <dba.h>
#include <seqdba.h>
#include <almsg.h>

struct s1{
    int   pcm ;
    char  type ;
    char  field ;
    int   tot_ipn ;
} ;

/* number of analog, digital and device EPNS */
/* Tank farm signals are first                */

static struct s1 point[] =
{
    4,      AI,      AI_ALMCT,      135,
    4,      DIN,     DIN_ALMCT,     300,
    4,      DEV,     DEV_ALMCT,     100,
    4,      DGR,     DGR_ALMCT,      3,
    4,      DOT,     DOT_ALMCT,      3,
    2,      AI,      AI_ALMCT,     380,
    2,      DIN,     DIN_ALMCT,     348,
    2,      DEV,     DEV_ALMCT,     217,
    2,      DGR,     DGR_ALMCT,      7,
    2,      DOT,     DOT_ALMCT,      4
} ;

static char fil[]="09" ;

main()
{
    char variab[EPNSIZ+3], epn[EPNSIZ+1];
    char buff [ MAX_DB_RECISZ ], net;
    int descrip[(NUMDESC*DESCSZ)+1] ; int seq_pcm = 2 ;

```

WHC-SD-534-CSWL-001 REV 3

```

int descrip1[8], descrip2[8], descrip3[8], descrip4[8];
int string[NUMDESC*DESCSZ+1], bottom[NUMDESC*DESCSZ+3];
int u_rel = UNITREL; int sequit = 0;
int access, conreg, dbreg, i, j, index, index;
int fstart, fstop, ftrip0, ftrip1, ftrip2, ftrip3, count;
int ipn, err, tot_val, sunit;
unsigned long epoch;
int tt;

net = 0;
access = UNITREL; sunit = 0;

checkerr (1, mopen());
checkerr (2, dbreg = lnrg ("DBASE" ));
checkerr (3, conreg = lnrg ("CONFIG" ));

/* clear screen */

for ( j = 0; j < 100; j++ )
{
    index = j;
    checkerr ( 4, put_vseq ( net, 2, SG, access, sunit, index, fil ) );
    index = 100 + j;
    checkerr ( 5, put_vseq ( net, 2, SG, access, sunit, index, fil ) );
}

/* initialize batch variables */

fstart = 1; fstop = 0; ftrip0 = 0; ftrip1 = 0; ftrip2 = 0; ftrip3 = 0;
put_vseq ( net, 2, FG, access, sunit, 425, &fstart ); /* sets fstart */
put_vseq ( net, 2, FG, access, sunit, 426, &fstop ); /* clear fstop */
put_vseq ( net, 2, FG, access, sunit, 427, &ftrip0 ); /* clear ftrip0 */
put_vseq ( net, 2, FG, access, sunit, 428, &ftrip1 ); /* clear ftrip1 */
put_vseq ( net, 2, FG, access, sunit, 429, &ftrip2 ); /* clear ftrip2 */
put_vseq ( net, 2, FG, access, sunit, 430, &ftrip3 ); /* clear ftrip3 */
fstart = 0; fstop = 1; ftrip0 = 1; ftrip1 = 1; ftrip2 = 1; ftrip3 = 1;

sleep(1); /* give sequence program chance to run */

index = 0;
jindex = 100;
count = 0;

for (i = 0; i < 10; i++)
{
    for (ipn = 0; ipn < point[i].tot_ipn; ipn++)
    {
        err = grb ( net, point[i].pcm, point[i].type, ipn, buff );
        if ( err < 0 ) continue;
        checkerr ( 6, gfb ( point[i].type, point[i].field, &tot_val, buff ) );
        if ( tot_val & 16) /* if alm
        inhibited...*/
        {
            err = idepn ( dbreg, epn, net, point[i].pcm, point[i].type, ipn ); /* get EPN and
            descriptor */
            if ( err < 0 ) continue;
            cpystr(variab1,"09", epn);
            checkerr ( 8, put_vseq ( net, seq_pcm, SG, u_rel, sequit, index++, variab1));
            sleep(1);
        }
        descrip[ squeeze ( descrip, desc ( dbreg, descrip, net, point[i].pcm, point[i].type, ipn), NULL) ] = 0;
    }

    descrip1[ squeeze ( descrip1, dstring(dbreg, descrip1, net, point[i].pcm, point[i].type, ipn, 0), NULL)];
    descrip2[ squeeze ( descrip2, dstring(dbreg, descrip2, net, point[i].pcm, point[i].type, ipn, 1), NULL)];
    descrip3[ squeeze ( descrip3, dstring(dbreg, descrip3, net, point[i].pcm, point[i].type, ipn, 2), NULL)];

```

WHC-SD-534-CSWL-001 REV 3

```

descrip4[squeeze(descrip4,dstring(dbrg, descrip4, net, point[i].pcm, point[i].type, ipn, 3), NULL)];
    cpystr ( bottom,"09", descrip1, " ", descrip2, " ", descrip3, " ", descrip4, NULL) ;

    checkerr ( 10, put_vseq ( net, seq_pcm, SG, u_rel, sequint,index++, bottom));
    if ( count > 20 ) put_vseq ( net, 2, FG, access, sunit, 427, &ftrip0 ) ; /* more on page
0 */
    if ( count > 40 ) put_vseq ( net, 2, FG, access, sunit, 428, &ftrip1 ) ; /* more on page 1 */
    if ( count > 60 ) put_vseq ( net, 2, FG, access, sunit, 429, &ftrip2 ) ; /* more on page 2 */
    if ( count > 80 ) put_vseq ( net, 2, FG, access, sunit, 430, &ftrip3 ) ; /* more on page 3 */
    if (++count > 99)
    {
        /* exceeds number of strings
*/
        put_vseq ( net, 2, FG, access, sunit, 425, &fstart ) ; /* clears 'running' message
*/
        put_vseq ( net, 2, FG, access, sunit, 426, &fstop ) ; /* lights 'too many inhibited
alarms' */
    }
    exit();
    /* count */
    /* tot_val */
    /* ipn */
    /* i */
    put_vseq ( net, 2, FG, access, sunit, 425, &fstart ) ; /* clears 'running' message if normal term.
*/
}
/* main */

checkerr(x,ierr)
{
    int ierr,x ;
    {
        int r2,r3 ;
        if (ierr < 0 )
        {
            printf("\n checkerr problem on test.c at %d, error = %d\n",x,ierr);
            r2 = lnkrg("CONFIG");
            r3 = lnkrg("ALMQUE");
            sndmsg(0,0,r3,r2,3,MSG_TEXT,CRIT,
"\n ALARM INHIBIT TEST PROGRAM --- ERROR ROUTINE FAILURE AT %d, ERROR %d\n",
x,ierr);
            exit();
        }
    }
}

checkipn(x,y,ierr)
{
    int ierr,x,y ;
    {
        int r2,r3 ;
        if (ierr < 0 )
        {
            printf("\n checkerr problem on ackflg.c at %d, error = %d, ipn = %d\n",x,ierr,y);
            r2 = lnkrg("CONFIG");
            r3 = lnkrg("ALMQUE");
            sndmsg(0,0,r3,r2,3,MSG_TEXT,CRIT,
"\n ALARM ACKNOWLEDGE PROGRAM --- ERROR ROUTINE FAILURE AT %d, ERROR %d\n",
x,ierr);
            exit();
        }
    }
}

```


WHC-SD-534-CSWL-001 REV 3

UKF14

PAGE 660

```

/* +
*   U K F 1 4 . C
*
*   .IDENT    /33MF00/
*+
*   Disk = SD0:    UIC = [261,10]
*
*   Copyright (C) 1992 - Texas Instruments
*   ALL RIGHTS RESERVED
*
*   This program is a confidential trade secret
*   of Texas Instruments which may be used only
*   with the system with which it was supplied
*   and which may not be disclosed, published,
*   disseminated, sold or in any way divulged
*   without advance express written permission
*   of Texas Instruments.
*
*
*   Program transcribed by :   Mike Fowler
*   Facility   : 242-A Evaporator Bldg.
*   Machine    : VAX 4104-A
*   Compiler    : DEC VAX C V3.3
*   C .cmd use : SET DEF D3SOURCE::[261,10]
*   C .cmd use : BLD UKF14 UKF14.C -L -MOV
*   Op. System : VAX/VMS 5.5-2AH
*   D/3 Version: 6.3
*
*   Reactivate Alarm Summary Pages by clearing the alarm
*   condition of all I/O points in each PCM. This causes
*   the PCM's to regenerate alarms for I/O points that are
*   still active.
*   This Program can be initiated from a console via
*   general function 14.
*
*+
*   PROGRAM MODIFICATION LOG
*
*+
*33MF00    31-JAN-92    DEFINE PROGRAM FOR D/3 VERSION 3.3
*
*   01-SEP-94          Sleeps now seconds instead of ticks          J. Hayden
*
*   02-SEP-94          %i now %d, #includes different                J. Hayden
*
*   10-SEP-94          Some functions changed now wanting address of
*                      Variable instead of variable and vice-versa
*                      J. Hayden
*
*   11-SEP-94          Ppragma compiler directives to
*                      initialize dlist variables                    J. Hayden
*
*   26-SEP-95          Changed program so multiple general
*                      functions do not have to be run.
*                      See EDN 623938 & EE-95-00006.
*
*   #includes changed
*   parameters to functions changed, glf,plf
*   pragma for dlist declarations, initialize
*   sleep now seconds instead of ticks
*   %p not %s, putfmt-> printf,%i->%d
*   mopen() now net_saps
*
*   END OF MOD LOG
*
*/
#include <omcstd.h>
#include <file.h>
#include <d3sys.h>
#include <grinnell.h>
#include <conkey.h>
#include <keyequ.h>

```

WHC-SD-534-CSWL-001 REV 3

```

#include <err.h>
#include <video.h>
#include <almmsg.h>
#include <conf.h>
#include <unitdb.h>
#include <dbaal.h>
#include <dbaal.h>
#include <dbadot.h>
#include <dbadgr.h>
#include <devdba.h>
#include <digdef.h>
#include <func.h>

#define UDF14      14
#define NOBTYPES   5
#define IPNSMAX    4681      /* unusual max count for e/ipns */
#define DONE      1

#pragma nostandard

PUBLIC int      dlrd      = 0;      /* DLIST descriptor */
PUBLIC struct dlist dlist  = 0;      /* DLIST structure */
PUBLIC struct gcb  gcb    = 0;      /* Struct for Grinnel Control Block */
PUBLIC short con = 0;

#pragma standard

/*
 * Define struct for Node Status'
 */
struct unitstat ustat      = 0;      /* for Micro Status determination */

/*
 * Define field for micro data
 */
unsigned short fldbuf[4] = 0;

/*
 * Define the field list
 */
static char fldlst[] = (
    AI_ALMCT, AI_CURAL,
    DIN_ALMCT,
    DOT_ALMCT,
    DGR_ALMCT,
    DEV_ALMCT, DEV_FLG, DEV_STAT, DEV_DFLG,
);

/*
 * Database types
 */
/* 0, 1, 2, 3, 5 */
static char dbtype[] = ( AI, DIN, DOT, DGR, DEV, );
static int  nflds[]  = ( 2, 1, 1, 1, 4, );

static char net = 0;
static char type = 0;

static char dbase[] = "DBASE";
static char config[] = "CONFIG";

/*
 * M A I N ( UKF14 )
 */
main( )
(
    int dbrd;      /* DBASE region id */

```

WHC-SD-534-CSWL-001 REV 3

```

int rdc;                /* Config region descriptor */
int ipn;                /* current ipn */
char fldx;              /* fldlst index */
char uname[ UNAMSIZ+1 ]; /* unit name buffer */
char epn[ EPNSIZ+1 ];   /* epn buffer */
char status[36];        /* mf - temp string for status */
char pcm_node[64];       /* table of PCM's completed or failed */
int unit;               /* PCM unit number */
char backup;            /* PCM backup unit number */
char buffer[80];
char anydone;           /* Any Nodes regenerated ? */
int err;                /* error codes */
int i, exit_code;       /* misc variables */
long alarms = 0;        /* total alarms regenerated */

con = gtnum();
if( con < 0 || con > 3 )
{
    con = 0;
    printf("\n-- UKF14 - REGENERATING ALL ALARMS ( Invoked - Interactive Mode )\n" );
}
else
    /* Else - Invoked by Gen Func 44 at console */

{
    printf("\n-- UKF14 - REGENERATING ALL ALARMS Console %d \n", con );
    wrt_con1( con );
    /* Send message that 44 will be done on both */
    /* End else from CONSOLE */

sprintf(buffer, "@d3source:[261010]reset.com");
exit_code = d3spawn(buffer, 0, 0);
if(exit_code == ER_GEN)
    printf("Error spawing RESET.COM\n");

sleep(3);
sprintf(buffer, "@d3source:[261010]dcmreset.com");
exit_code = d3spawn(buffer, 0, 0);
if(exit_code == ER_GEN)
    printf("Error spawing DCMRESET.COM\n");

/*      system("@d3source:[261010]reset.com");
*/      system("@d3source:[261010]dcmreset.com");

sleep(120);          /* Wait for 44 to get done */

con = gtnum();
if( con < 0 || con > 3 )
{
    con = 0;
    printf("\n-- UKF14 - REGENERATING ALL ALARMS ( Invoked - Interactive Mode )\n" );
}
else
    /* Else - Invoked by Gen Func 44 at console */

{
    printf("\n-- UKF14 - REGENERATING ALL ALARMS Console %d \n", con );
    wrt_con2( con );
    /* Send message that 44 will be done on both */
    /* End else from CONSOLE */

/*
*      Open the communication lines.
*/
if( net_saps(FALSE,TRUE) < 0 ) {
    printf("UKF14 - cant open communication, mopen()\n");
    exit();
}

/*
*      Link to the regions (config, dbase, )

```

```

/*
  if( (rdc = lnkrq("config")) < 0 ) {    /* to CONFIG table */
      printf("UKF14 - Can't link to CONFIG, task exiting \n");
      exit();
  }

  if( (dbrd = lnkrq(dbase)) < 0 ) {      /* link to database */
      printf("UKF14 - can't link to DBASE, task exiting \n");
      unlrq( rdc );
      exit();
  }

/*
 * Initialization
 */
  anyone = 0;
  unit = -1;
  for( i=0; i<sizeof(pcm_node); i++ )
      pcm_node[i] = 0;

/*
 * MAIN LOOP - While each pcm on network 0.
 */
  while( ++unit < NUMPCM ) {
      /*
       * Get unitname for next pcm.
       */
      for( type=0; type != UPCM; unit++ ) {

          if(unit >= NUMPCM) goto quit;          /* Maxed out */
          fill( uname,sizeof(uname),'0' );

          err = idpcm( rdc,uname,net,unit );
          if( err < 0 ) {

              if( err == ER_SND || err == ER_UNF || err == ER_UNL )
                  continue;

              printf("UKF14 - Can't get unit name for unit #%d, err=%d\n",unit,err);
              goto quit;
          }

          if( err = pcmid( rdc,uname,&net,&unit,&type,&backup) < 0 ) {

              printf("UKF14 - Can't get unit type for unit# %d, err=%d\n",unit,err);
              goto quit;
          }

          if( type == UPCM ) {

              /* Has the Primary unit been done already ? */

              if( (unit>backup) && (backup!=-1) && (pcm_node[backup] == DONE) ) {
                  printf( "UKF14 - %s is a backup (BYPASSED) \n", uname );
                  type = 0;
                  continue;
              }

              err = gsstat( rdc, net, unit, &ustat );

              if( ustat.us_pro != STOK )
                  type = 0;

              /*
               * Processor status
               */
              switch( ustat.us_pro ) {

```

MHC-SD-534-CSWL-001 REV 3

```

case( STOFF ):  cpystr( status, "offline", 0 );
                break;
case( STLOD ):  cpystr( status, "load request", 0 );
                break;
case( STIDL ):  cpystr( status, "idle", 0 );
                break;
case( STOK ):   cpystr( status, "running", 0 );
                break;
default:        cpystr( status, "UNKNOWN", 0 );
                break;
    }

    if( ustat.us_pro != STOK )
        printf( "UKF14 - %s is %s (BYPASSED) \n", uname, status );
    else printf( "UKF14 - %s is %s \n", uname, status );

} /* end if */
} /* end for( unit++ ) loop */

unit--;
anydone = 1;
if( backup == -1 ) backup = unit;
pcm_node[ unit ] = pcm_node[ backup ] = DONE;    /* Now do it ! */

/*
 * For each database type
 */
fldx = 0;
for( i=0; i< NDBTYPES; i++ ) (
    printf( "ndbtype is %d\n",i);

    /*
     * For every ipn in a given pcm (unit).
     */
    for( ipn=0; ipn <= IPNSMAX; ipn++) (
        printf( " ipn is %d\n",ipn);

        /*
         * Get the field data to determine the
         * uninhibited points that are in alarm.
         */
        err = glf( net, unit, dbtype[i], 1, &ipn,
                    nflds[i], &fldlst[fldx], fldbuf);

        if( err < 0 ) (
            if( err == ER_IPN || err == E_PWPF )
                break;
            else continue;
        ) /* next ipn */

    /*
     * Uninhibited and Alarm conditions ?
     */
    if( !(fldbuf[0] & BIT_ALLIN) && (fldbuf[0] & BIT_DISAL) ) (

        /*
         * Clear current Alarm status condition
         * and for DIN's and DOT's save old
         * CFN or COS using MSB
         */
        if( (dbtype[i] == DIN &&
              (fldbuf[0] & (DB_COS | DB_CFN0 | DB_CFN1) )) ||
            (dbtype[i] == DOT && (fldbuf[0] & DB_COS) ))
            fldbuf[0] |= BIT_NOALM;

        fldbuf[0] &= ~BIT_DISAL;

    /*
     * Set doit bit for devices;
     * Clear current Alarm for analog inputs;

```

WHC-SD-534-CSWL-001 REV 3

```

    * other database types this is moot.
    */

    if( dbtype[i] == DEV ) {
        fldbuf[1] != DV_D01T;
        fldbuf[2] = 0;
        fldbuf[3] = 0;
    }
    else fldbuf[1] = 0;

    alarms += 1;
} /* end if( alarms ) */

else { /* Else there is no alarm, or its inhibited */
    printf("UKF14 - NOT ALARMED - unit=%d db=%d ipn=%d BUF=%d fldx=%d fld=%d\n",
        unit,dbtype[i],ipn,fldbuf[0],fldx,fldlst[fldx]);

    continue; /* next ipn - for loop */
}

/*
 * Write fields(s) to P C M - regenerate ALARMS
 */
err = plf( net, unit, dbtype[i], 1, &ipn,
    nfls[i], &fldlst[fldx], fldbuf);
if( err < 0 )
    printf("UKF14 - PLF ERROR=%d (unit=%d dbtype=%d ipn=%d nfls=%d fldx=%d fld=%d)\n",
        err, unit, dbtype[i], ipn, nfls[i], fldx, fldlst[fldx]);

} /* end for( each ipn ) */
printf(" done with ipns\n");

fldx += nfls[i];
sleep(5); /* allow some alarm processing time */

} /* end for( each DBTYPE ) */
printf(" done with type\n");
sleep(5);
} /* end while( units 0->63 ) */
printf(" done with units\n");

/*
 * Clean up shop and getout
 */
quit:

con = gtnum();
if( con < 0 || con > 3 )
{
    con = 0;
    printf("\n-- UKF14 - REGENERATING ALL ALARMS ( Invoked - Interactive Mode )\n" );
}
else /* Else - Invoked by Gen Func 44 at console */
{
    printf("\n-- UKF14 - REGENERATING ALL ALARMS Console %d \n", con );
    wrt_con3( con ); /* Send message that 44 will be done on both */
} /* End else from CONSOLE */

unlrg( rdc );
unlrg( dbrd );
if( anyone ) {
    printf( "UKF14 - Total Alarms Regenerated (%d) \n", alarms );
    printf( "UKF14 - ALARM REGENERATION COMPLETED \n" );
}
else
    printf( "UKF14 - ALARM REGENERATION WARNING - ALL PCM NODES OFFLINE \n" );
return(1);

```

```

) /* end main( UKF14 ) */

/*
 * WRT_CON1( )
 *
 * Draw the General Function 14 - Display
 */
wrt_con1(con)
int con;
{
    int kbfd; /* Key board fd */
    int kboutfd; /* keyboard output fd */
    int err; /* error code */
    static char dlname[] = "DLISTxx";

    dlnam[ 5 + itob( &dlname[5], con, 10 ) ] = 0;
    if( (dlrd = lnrg(dlname)) < 0 ) {
        printf("UKF14 - DLIST LINK FAILURE %d\n",dlrd);
        unlrg(dlrd);
        exit();
    }
    if( (err = ginit(&gcb, con)) < 0 ) {
        printf("UKF14 - CANT ACCESS GRINNELL, ERR %d\n",err );
        unlrg(dlrd);
        exit();
    }
    dlist.act = DMUSER + UDF14;
    putrg(dlrd, &dlist.act, OFFSET(dlist.act), sizeof(dlist.act) );

/*
 *
 */
    CLEAR the OCM

    gsetup(&gcb,BLACK);
    erase(&gcb);
    gflush(&gcb);

/* OCM GRAPHICS */

    gsetup(&gcb,RED);
    cursor(&gcb,10,425); /* SCREEN FORMAT */
    horiz( &gcb,440,3); /* */
    cursor(&gcb,447,425); /* General Function 14 */
    vert( &gcb,-105,3); /* */
    cursor(&gcb,449,320); /* Clearing alarm pages */
    horiz( &gcb,-439,3); /* */
    cursor(&gcb,10,320); /* On both machines */
    vert( &gcb,105,3); /* */
    cursor(&gcb,10,390); /* */
    horiz( &gcb,440,3); /* */

/* OCM TEXT */

    gsetup(&gcb, WHITE );
    dublchar(&gcb);
    cursor(&gcb,HCHAR*3,VCHAR*40);
    g_puts( &gcb, "GENERAL FUNCTION 14" );
    gflush(&gcb);
    gsetup( &gcb, YELLOW );
    widechar(&gcb);
    cursor(&gcb,HCHAR*3,VCHAR*36);
    g_puts( &gcb, "CLEARING ALARM PAGES" );
    gflush(&gcb);
    cursor(&gcb,HCHAR*3,VCHAR*34);
    g_puts( &gcb, "ON BOTH MACHINES" );
    gflush(&gcb);
    gsetup(&gcb, WHITE );
    cursor(&gcb,HCHAR*12,VCHAR*31);
    g_puts( &gcb, "" );
    gflush(&gcb);
}

```


WHC-SD-534-CSWL-001 REV 3

```

*      Release the console and dlist region
*/
    dlist.ukf = -1;
    dlist.act = 0;
    putrg(dlrd,&dlist.ukf,OFFSET(dlist,ukf),sizeof(dlist.ukf));
    putrg(dlrd,&dlist.act,OFFSET(dlist,act),sizeof(dlist.act));
    unlrg( dlrd );
    return(1);
} /* end wrt_con1() */

/*
* WRT _ C O N 2( )
*
* Draw the General Function 14 - Display
*/
wrt_con2(con)
int con;
{
    int kbfd; /* Key board fd */
    int kboutfd; /* keyboard output fd */
    int err; /* error code */
    static char dlname[] = "DLISTxx";

    dlname[ 5 + itob( &dlname[5], con, 10 ) ] = 0;
    if((dlrd = lnkrg(dlname)) < 0 ) {
        printf("UKF14 - DLIST LINK FAILURE %d\n",dlrd);
        unlrg(dlrd);
        exit();
    }
    if((err = ginit(&gcb, con)) < 0 ) {
        printf("UKF14 - CANT ACCESS GRINNELL, ERR %d\n",err );
        unlrg(dlrd);
        exit();
    }
    dlist.act = DMUSER + UDF14;
    putrg(dlrd, &dlist.act, OFFSET(dlist,act), sizeof(dlist.act) );

/*
*
* C L E A R the O C M
*/
    gsetup(&gcb, BLACK);
    erase(&gcb);
    gflush(&gcb);

/* O C M G R A P H I C S */

    gsetup(&gcb, RED);
    cursor(&gcb, 10, 425); /* SCREEN FORMAT */
    horiz( &gcb, 440, 3); /* */
    cursor(&gcb, 447, 425); /* General Function 14 */
    vert( &gcb, -105, 3); /* */
    cursor(&gcb, 449, 320); /* Regenerating Alarm pages */
    horiz( &gcb, -439, 3); /* */
    cursor(&gcb, 10, 320); /* Alarms printing at typer. */
    vert( &gcb, 105, 3); /* */
    cursor(&gcb, 10, 390);
    horiz( &gcb, 440, 3);

/* O C M T E X T */

    gsetup(&gcb, WHITE );
    dublchar(&gcb);
    cursor(&gcb, HCHAR*3, VCHAR*40);
    g_puts( &gcb, "GENERAL FUNCTION 14" );
    gflush(&gcb);
    gsetup( &gcb, YELLOW );
    widechar(&gcb);
    cursor(&gcb, HCHAR*3, VCHAR*36);
    g_puts( &gcb, "REGENERATING ALARM PAGES" );

```

```

gflush(&gcb);
cursor(&gcb, HCHAR*3, VCHAR*34);
g_puts( &gcb, "ALARMS PRINTING AT TYPER." );
gflush(&gcb);
gsetup(&gcb, WHITE );
cursor(&gcb, HCHAR*12, VCHAR*31);
g_puts( &gcb, "" );
gflush(&gcb);

/*
 *
 */
    Release the console and dlist region

    dlist.ukf = -1;
    dlist.act = 0;
    putrg(dlrd, &dlist.ukf, OFFSET(dlist, ukf), sizeof(dlist.ukf));
    putrg(dlrd, &dlist.act, OFFSET(dlist, act), sizeof(dlist.act));
    unlrg( dlrd );
    return(1);

) /* end wrt_con2() */

/*
 * WRT_CON3( )
 *
 * Draw the General Function 14 - Display
 */
wrt_con3(con)
int con;
{
    int kbfd; /* Key board fd */
    int kboutfd; /* keyboard output fd */
    int err; /* error code */
    static char dlname[] = "DLISTxx";

    dlname[ 5 + itob( &dlname[5], con, 10 ) ] = 0;
    if((dlrd = (krgr(dlname)) < 0) {
        printf("UKF14 - DLIST LINK FAILURE %d\n", dlrd);
        unlrg(dlrd);
        exit();
    }
    if((err = ginit(&gcb, con)) < 0) {
        printf("UKF14 - CANT ACCESS GRINNELL, ERR %d\n", err );
        unlrg(dlrd);
        exit();
    }
    dlist.act = DMUSER + UDF14;
    putrg(dlrd, &dlist.act, OFFSET(dlist, act), sizeof(dlist.act) );

/*
 *
 */
    CLEAR the OCM

    gsetup(&gcb, BLACK);
    erase(&gcb);
    gflush(&gcb);

    /* OCM GRAPHICS */

    gsetup(&gcb, YELLOW);
    cursor(&gcb, 10, 425);
    horiz( &gcb, 440, 3);
    cursor(&gcb, 447, 425);
    vert( &gcb, 105, 3);
    cursor(&gcb, 449, 320);
    horiz( &gcb, 439, 3);
    cursor(&gcb, 10, 320);
    vert( &gcb, 105, 3);
    cursor(&gcb, 10, 390);
    horiz( &gcb, 440, 3);

/* SCREEN FORMAT */
/*
 *
 * General Function 14
 *
 * All Alarms processed
 *
 * General Function Complete
 *
 */
    gsetup(&gcb, WHITE );

```

WHC-SD-534-CSWL-001 REV 3

```
dublchar(&gcb);
cursor(&gcb,HCHAR*3,VCHAR*40);
g_puts( &gcb, "GENERAL FUNCTION 14" );
gflush(&gcb);
gsetup( &gcb, YELLOW );
widechar(&gcb);
cursor(&gcb,HCHAR*3,VCHAR*36);
g_puts( &gcb, "ALL ALARMS PROCESSED" );
gflush(&gcb);
cursor(&gcb,HCHAR*3,VCHAR*34);
g_puts( &gcb, "GENERAL FUNCTION COMPLETE." );
gflush(&gcb);
gsetup(&gcb, WHITE );
cursor(&gcb,HCHAR*12,VCHAR*31);
g_puts( &gcb, "" );
gflush(&gcb);

/*
 * Release the console and dlist region
 */
dlist.ukf = -1;
dlist.act = 0;
putrg(dlrd,&dlist.ukf,OFFSET(dlist,ukf),sizeof(dlist.ukf));
putrg(dlrd,&dlist.act,OFFSET(dlist,act),sizeof(dlist.act));
unlrg( dlrd );
return(1);

} /* end wrt_con3() */
```

WHC-SD-534-CSWL-001 REV 3

UKF34

WHC-SD-534-CSWL-001 REV 3

```

/*          U K F 3 4 . C                      VERSION 5
*
*
*
* General Function 34 - To remove inhibited alarms for Plant startup.
*                      Reversal of UKF43. UKF34 is only loaded to the
*                      CDCM, as are UKF43 (alarm inhibiting) and UKF14
*                      (alarm regeneration).
*
*
*
* Written by : M. C. Teats
* Date       : 03-08-93
* Facility   : 242-A Evaporator Bldg.
* Machine    : VAX 4104-A
* Compiler   : DEC VAX C V3.3
* C .cmd use : SET DEF D3SOURCE::[261,10]
* C .cmd use : BLD UKF34 UKF34.C -L -MOV
* Op. System : VAX/VMS 5.5-2AH
* D/3 Version: 6.3
*
* MODIFICATION LOG:
*
* 08-MAR-93      Define program for D/3 version 3.3                M. Teats
*
* 07-MAY-93      Modify ipns per CCN# 93MAR25-01 (search            M. Teats
*                  MCT1).
*
* 05-JAN-95      Rebase lined the program for upgrade to           J. Berger
*                  version 6.3 of the D/3 software. Added
*                  additional EPNs to the inhibit list.
*                  This will be documented on the rebaselined
*                  documentation for the 6.3 upgrade.
*
* 27-APR-95      Added EPNs P1-CA1-20, YS-PB1-1, YS-P-C100         J. Berger
*                  and YS-AW102 to program. See WP EE-95-00006
*                  and ECN 622499. (JFB2)
*
* 26-SEP-95      Revised the screens that are displayed to         J. Berger
*                  the OCMs. See ECN 623938 & EE-95-00006.
*
* END OF MOD LOG
*/

#include <emcstd.h>
#include <cd3sys.h>
#include <unitdb.h>
#include <grinnell.h>
#include <dbatypes.h>
#include <dbaal.h>
#include <dbadin.h>
#include <dbadev.h>
#include <dlgdef.h>
#include <conkey.h>
#include <conf.h>
#include <err.h>
#include <keyeq.h>
#include <func.h>
#include <seqdba.h>
#include <slmsg.h>

#define UDF34      34
#define DONE       1

#pragma nostandard
PUBLIC int dlrld =0; /* DLIST descriptor */
PUBLIC struct dlist dlist =0; /* DLIST structure */
PUBLIC struct gcb gcb =0; /* Struct for Grinnel Control Block */
PUBLIC short con = 0; /* console number */
#pragma standard

```

WHC-SD-534-CSWL-001 REV 3

```

/*
 *
 */
#define STRUCTURE FOR NODE STATUS:

/*
struct unitstat ustat =0; /* For Micro Status determination */
/*
#define STRUCTURES FOR EPN's TO REMOVE INHIBIT FOR:
*/

struct s1
(
    char pcm; /* PCM unit number */
    dbt; /* DBASE type */
    int sbi; /* Scan block ipn */
);

static struct s1 inha [] =
(
/* k pcm dbt sbi epn */
/* -- --- --- --- */
/* 0 */ 2, A1, 45, /* pi-ca1-11 */
/* 1 */ 2, A1, 46, /* pic-ca1-7 */
/* 2 */ 2, A1, 53, /* lic-ca1-1 */
/* 3 */ 2, A1, 57, /* lic-ca1-2 */
/* 4 */ 2, A1, 63, /* li-ca1-3 */
/* 5 */ 2, A1, 68, /* fic-ca1-6 */
/* 6 */ 2, A1, 81, /* FI-CA1-7 */ /* JFB1 */
/* 7 */ 2, A1, 83, /* vi-pb1-2a */
/* 8 */ 2, A1, 84, /* vi-pb1-1a */
/* 9 */ 2, A1, 85, /* vi-pb1-3a */
/* 10 */ 2, A1, 86, /* fi-ca1-1 */
/* 11 */ 2, A1, 87, /* pi-ca1-9 */
/* 12 */ 2, A1, 89, /* fi-ca1-3 */
/* 13 */ 2, A1, 97, /* pi-ca1-20 */ /* JFB2 */
/* 14 */ 2, A1, 102, /* sic-pb2-1 */ /* JFB1 */
/* 15 */ 2, A1, 103, /* vi-pb2-1a */
/* 16 */ 2, A1, 106, /* fi-ca1-2 */
/* 17 */ 2, A1, 107, /* pl-ca1-10 */
/* 18 */ 2, A1, 110, /* fic-ca1-4 */
/* 19 */ 2, A1, 130, /* pi-ec1-16 */ /* JFB1 */
/* 20 */ 2, A1, 131, /* fic-ca1-1 */
/* 21 */ 2, A1, 153, /* pi-ec1-4 */ /* MCT1 */
/* 22 */ 2, A1, 154, /* pi-ec2-2 */ /* MCT1 */
/* 23 */ 2, A1, 162, /* fic-ec3-1 */ /* MCT1 */
/* 24 */ 2, A1, 197, /* pic-ec1-2 */
/* 25 */ 2, A1, 198, /* pi-ec1-9 */ /* JFB1 */
/* 26 */ 2, DIN, 50, /* ys-aw102 */ /* JFB2 */
/* 27 */ 2, DIN, 60, /* psh-ca111 */
/* 28 */ 2, DIN, 73, /* ys-pb1-1 */ /* JFB2 */
/* 29 */ 2, DIN, 87, /* TSH-PB2-1 */ /* JFB1 */
/* 30 */ 2, DIN, 90, /* SXS-PB2-1 */ /* JFB1 */
/* 31 */ 2, DIN, 91, /* YS-VFD-1 */ /* JFB1 */
/* 32 */ 2, DIN, 92, /* SSL-PB2-1 */ /* JFB1 */
/* 33 */ 2, DIN, 107, /* ZS-EA1-1C */ /* JFB1 */
/* 34 */ 2, DIN, 163, /* ys-p-c100 */ /* JFB2 */
/* 35 */ 2, DEV, 102, /* RC2-PIG */ /* JFB1 */
/* 36 */ 2, DEV, 133, /* RC3-PIG */ /* JFB1 */
/* 37 */ 2, DEV, 143, /* RC1-PIG */ /* JFB1 */
);

static char type = 0; /* For PCM processor type, status */

/*****
 *
 * M A I N ( UKF34 )
 *
 *****/

main( )
(
    char net; /* Network number */
    uname[ UNAMSIZ+1 ], /* Unit name buffer */
    epn[ EPNSIZ+1 ], /* EPN buffer */

```

WHC-SD-534-CSWL-001 REV 3

```

status[36],          /* Temp string for status */
pcm_node[64],        /* Table of PCM's completed or failed */
backup,              /* PCM backup unit number */
anydone,             /* Any Nodes done ? */
buff[ MAX_DB_RECSIZ ], /* Largest db block */
field;              /* Alarm count field */
int dbreg,           /* DBASE region id */
    creg,            /* Config region id */
    areg,            /* Alarm que region id */
    unit,            /* PCM unit number */
    npcm,            /* Maximum PCM unit number = 3 */
    value,           /* Alarm count field's bit value */
    k,               /* De-Inhibit for-loop counter */
    err,             /* Error code for dbase access calls */
    i;               /* Miscellaneous variables */
net = 0;
npcm = 3;

con = gtrun();
if( con < 0 || con > 3 )
{
    con = 0;
    printf("\n-- UKF34 - ALARM INHIBIT REMOVAL ( Invoked - Interactive Mode )\n" );
}
else
{
    /* Else - Invoked by Gen Func 34 at console */
    printf("\n-- UKF34 - ALARM INHIBIT REMOVAL Invoked from Console %d \n", con );
    wrt_con( con );
    /* End else from CONSOLE */
}

/*
 * FIRST, LINK TO THE CONFIG TABLE, DATABASE, AND ALARM QUE REGIONS:
 */
if( (creg = lnkrg("config")) < 0 )
{
    /* Link to config */
    printf("\n -- UKF34 -- CAN'T LINK TO CONFIG - TASK EXITING --\n");
    exit();
}
if( (dbreg = lnkrg("DBASE")) < 0 )
{
    /* Link to database */
    printf("\n -- UKF34 -- CAN'T LINK TO DBASE - TASK EXITING --\n");
    unlrg( creg );
    exit();
}
if( (areg = lnkrg("ALMQUE")) < 0 )
{
    /* Link to alarm que */
    printf("\n -- UKF34 -- CAN'T LINK TO ALMQUE - TASK EXITING --\n");
    unlrg( creg );
    unlrg( dbreg );
    exit();
}

/*
 * OPEN A COMMUNICATION CHANNEL TO THE PCM:
 */
if( net_saps(FALSE,TRUE) < 0 )
{
    printf("\n -- UKF34 -- CAN'T OPEN COMMUNICATIONS, MOPEN() - TASK EXITING\n");
    unlrg( creg );
    unlrg( dbreg );
    unlrg( areg );
    exit();
}

/*
 * INITIALIZATION:
 */
anydone = 0;
unit = -1;
for( i=0; i<sizeof(pcm_node); i++ )
    pcm_node[i] = 0;

```

```

*   MAIN LOOP -
*
*   WHILE PCMOA OR PCMOB (unit 2 or 3) ON NETWORK 0:
*/
while( ++unit <= npcm )
{
/*
*   GET UNIT NAME FOR NEXT PCM:
*/
for( type=0; type != UPCM; unit++ )
{
    if( unit > npcm ) goto quit;
    fill( uname,sizeof(uname),'\0' );

    err = idpcm( creg, uname, net, unit );
    if( err < 0 )
    {
        if( err == ER_SND || err == ER_UNF || err == ER_UNL )
            continue;
        printf("UKF34 - CAN'T GET UNIT NAME FOR UNIT #%d, err=%d\n",unit,err);
        goto quit;
    }
    if( err = pcmid( creg, uname, &net, &unit, &type, &backp) < 0 )
    {
        printf("UKF34 - CAN'T GET UNIT TYPE FOR UNIT #%d, err=%d\n",unit,err);
        goto quit;
    }
    if( type == UPCM )
    {
        /*
        *   HAS THE PRIMARY UNIT ALREADY BEEN DONE ?
        */
        if( (unit>backp) && (backp!=-1) && (pcm_node[backp] == DONE) )
        {
            printf( "UKF34 - %s IS BACKUP UNIT (BYPASSED) \n", uname );
            type = 0;
            continue;
        }
        err = gsstat( creg, net, unit, &stat );

        if( ustat.us_pro != STOK )
            type = 0;
        /*
        *   PROCESSOR STATUS:
        */
        switch( ustat.us_pro )
        {
            case( STOFF ):   cpyst( status, "offline", 0 );
                            break;
            case( STLOD ):  cpyst( status, "load request", 0 );
                            break;
            case( STIDL ):  cpyst( status, "idle", 0 );
                            break;
            case( STOK ):   cpyst( status, "running", 0 );
                            break;
            default:       cpyst( status, "UNKNOWN", 0 );
                            break;
        }
        if( ustat.us_pro != STOK )
            printf( "UKF34 - %s IS %s (BYPASSED) \n", uname, status );
        else printf( "UKF34 - %s IS %s \n", uname, status );
        /*
        *   END PROCESSOR STATUS
        */
    }
    /* End If ( type UPCM ) */
    /* End For ( unit++ ) loop */
}
unit--;
anydone = 1;
if( backp == -1 ) backp = unit;
pcm_node[ unit ] = pcm_node[ backp ] = DONE;

```


WHC-SD-534-CSWL-001 REV 3

```

/*
 *
 * EPN INHIBIT REMOVAL LOOP -
 *
 *
 * NOW WE ARE READY TO REMOVE INHIBITED ALARMS:
 *     err = idepn(rid, &epn, net, unit, dbtype, ipn)
 *     err = grb(net, unit, dbtype, ipn, &buff)
 *     err = gfb(dbtype, field, &value, &buff)
 *     err = pfb(dbtype, field, &value, &buff)
 *     err = prb(net, unit, dbtype, ipn, &buff)
 */
for (k = 0; k < 38; k++)      /* JFB1,2 */
{
    field = A1_ALMCT;
    if (k > 25)                /* JFB1,2 */
    {
        field = DIN_ALMCT;
    }
    if (k > 34)                /* JFB1,2 */
    {
        field = DEV_ALMCT;
    }
    err = idepn(dbreg, epn, net, inha[k].pcm, inha[k].dbt, inha[k].sbi);
    if(err < 0)
    {
        printf("\n --UKF34 IDEPN--ERROR = %d\n",err);
        perror(&err); /*
/*
        unlg(c reg );
        unlg( dbreg );
        unlg( areg );
        exit();
    }
    err = grb(net, inha[k].pcm, inha[k].dbt, inha[k].sbi, buff);
    if(err < 0)
    {
        printf("\n --UKF34 %s GRB--ERROR = %d\n", epn, err);
        perror(&err); /*
/*
        unlg(c reg );
        unlg( dbreg );
        unlg( areg );
        exit();
    }
    err = gfb(inha[k].dbt, field, &value, buff);
    if(err < 0)
    {
        printf("\n -- UKF34 %s GFB--ERROR = %d\n", epn, err);
        perror(&err); /*
/*
        unlg(c reg );
        unlg( dbreg );
        unlg( areg );
        exit();
    }
}
/*
 *
 * CHECK IF INHIBITED. IF TRUE, REMOVE IT AND SEND MESSAGE!
 */
if ( value = 16 )
{
    value = 32;
    err = pfb(inha[k].dbt, field, &value, buff); /* Remove inhibit */
    if(err < 0)
    {
        printf("\n -- UKF34 %s PFB--ERROR = %d\n", epn, err);
        perror(&err); /*
/*
        unlg(c reg );
        unlg( dbreg );
        unlg( areg );
        exit();
    }
}
sndmsg (0,0,areg,c reg,3,MSG_TEXT,NONCRIT,"--- UKF34 - INHIBIT REMOVED FOR %s",epn);
sleep (2);
}
/* End If

```

```

/*
 *      GET OUT AND GO TO NEXT EPN:
 */
err = prb(net, inha[k].pcm, inha[k].dbt, inha[k].sbi, buff);
if(err < 0)
{
    printf("\n --UKF34 %s PRB--ERROR = %d\n", epn, err);
    perror(&err); /*
/*
    unlrq( creg );
    unlrq( dbreg );
    unlrq( areg );
    exit();
}
sleep(1);
/* Take a rest
/* End removal for loop, k++
/* END while <= npcm
)
/*
 *      UNLINK REGIONS AND GET OUT:
 */
quit:
unlrq( creg );
unlrq( dbreg );
unlrq( areg );

con = gtnum();
if( con < 0 || con > 3 )
{
    con = 0;
    printf("\n-- UKF34 - ALARM INHIBIT REMOVAL ( Invoked - Interactive Mode )\n" );
}
else
/* Else - Invoked by Gen Func 34 at console
/*
{
    printf("\n-- UKF34 - ALARM INHIBIT REMOVAL Invoked from Console %d \n", con );
    wrt_con2( con );
/* End else from CONSOLE
/*

if( anyone )
{
    printf( "UKF34 - INHIBIT REMOVAL FOR 242-A STARTUP COMPLETED \n" );
}
else
    printf( "UKF34 - INHIBIT REMOVAL WARNING - ALL PCM NODES OFFLINE \n" );
return(1);

}
/* End main */

/*****
 *
 *      END MAIN
 *
 *****/

/*      WRT_CON1( )
 *
 *      Draw the General Function 34 - Display
 */
int wrt_con1( con )
int con;
{
    int kbfd;
    int kboutfd;
    int err;
    static char dname[] = "DLISTxx";

    dname[5 + itob( &dname[5], con ,10 ) ] = 0;
    if((dlrd = lnkrq(dname)) < 0)
    {
        printf("UKF34 - DLIST LINK FAILURE %d\n",dlrd);
    }
}

```

```

        unlrg(dlrd);
        exit();
    }
    if((err = ginit(&gcb, con)) < 0 )
    {
        printf("UKF34 - CANT ACCESS GRINWELL, ERR %d\n",err );
        unlrg(dlrd);
        exit();
    }
    dlist.act = DMUSER + UDF34;
    putrg(dlrd, &dlist.act, OFFSET(dlist.act), sizeof(dlist.act) );

/*
 *
 */
    CLEAR the OCM :

    gsetup(&gcb,BLACK);
    erase(&gcb);
    gflush(&gcb);

/* OCM GRAPHICS : */

    gsetup(&gcb,RED);
    cursor(&gcb,10,425); /* SCREEN FORMAT */
    horiz( &gcb,440,3); /*
    cursor(&gcb,447,425); /* General Function 34
    vert( &gcb,105,3); /*
    cursor(&gcb,449,520); /* Alarm Inhibit Removal
    horiz( &gcb,439,3); /*
    cursor(&gcb,10,320); /* EPN's printing at typer.
    vert( &gcb,105,3); /*
    cursor(&gcb,10,390); /*
    horiz( &gcb,440,3);

/* OCM TEXT : */

    gsetup(&gcb, WHITE );
    dublchar(&gcb);
    cursor(&gcb,HCHAR*3,VCHAR*40);
    g_puts( &gcb, "GENERAL FUNCTION 34" );
    gflush(&gcb);
    gsetup( &gcb, YELLOW );
    widechar(&gcb);
    cursor(&gcb,HCHAR*3,VCHAR*36);
    g_puts( &gcb, "REMOVING INHA ALARMS FOR STARTUP " );
    gflush(&gcb);
    cursor(&gcb,HCHAR*3,VCHAR*34);
    g_puts( &gcb, "EPN'S WILL BE PRINTED AT TYPER " );
    gflush(&gcb);

/*
 *
 */
    RELEASE THE CONSOLE AND DLIST REGION:

    dlist.ukf = -1;
    dlist.act = 0;
    putrg(dlrd,&dlist.ukf,OFFSET(dlist.ukf),sizeof(dlist.ukf));
    putrg(dlrd,&dlist.act,OFFSET(dlist.act),sizeof(dlist.act));
    unlrg( dlrd );
    return(1);

/* End wrt_con1() */

}

/*
 *
 */
    WRT_CON2( )

/*
 */
    Draw the General Function 34 - Display

int wrt_con2( con )
int con;
{
    int kbfd; /* Key board fd */
    int kboutfd; /* Keyboard output fd */
    int err; /* Error code */
    static char dlname[] = "DLISTxx";

```

WHC-SD-534-CSWL-001 REV 3

```

dname[5 + itob ( &dname[5], con ,10 ) ] = 0;
if((dlrd = lnrg(dname)) < 0)
{
    printf("UKF34 - DLIST LINK FAILURE %d\n",dlrd);
    unlrg(dlrd);
    exit();
}
if((err = ginit(&gcb, con)) < 0 )
{
    printf("UKF34 - CANT ACCESS GRINWELL, ERR %d\n",err );
    unlrg(dlrd);
    exit();
}

dlist.act = DMUSER + UDF34;
putrg(dlrd, &dlist.act, OFFSET(dlist.act), sizeof(dlist.act) );

/*
 *
 */
CLEAR the OCM :

gsetup(&gcb,BLACK);
erase(&gcb);
gflush(&gcb);

/* OCM GRAPHICS : */

gsetup(&gcb,YELLOW);
cursor(&gcb,10,425);
horiz( &gcb,440,3);
cursor(&gcb,447,425);
vert( &gcb,-105,3);
cursor(&gcb,449,320);
horiz( &gcb,-439,3);
cursor(&gcb,10,320);
vert( &gcb,105,3);
cursor(&gcb,10,390);
horiz( &gcb,440,3);

/* SCREEN FORMAT */
/*
 *
 * General Function 34
 *
 *
 *
 *
 *
 */

/* OCM TEXT : */

gsetup(&gcb, WHITE );
dublchar(&gcb);
cursor(&gcb,HCHAR*3,VCHAR*40);
g_puts( &gcb, "GENERAL FUNCTION 34" );
gflush(&gcb);
gsetup( &gcb, YELLOW );
widechar(&gcb);
cursor(&gcb,HCHAR*3,VCHAR*36);
g_puts( &gcb, "GENERAL FUNCTION IS COMPLETE" );
gflush(&gcb);
cursor(&gcb,HCHAR*3,VCHAR*34);
g_puts( &gcb, "THE SCREEN CAN BE CLEARED " );
gflush(&gcb);

/*
 *
 */
RELEASE THE CONSOLE AND DLIST REGION:

dlist.ukf = -1;
dlist.act = 0;
putrg(dlrd,&dlist.ukf,OFFSET(dlist.ukf),sizeof(dlist.ukf));
putrg(dlrd,&dlist.act,OFFSET(dlist.act),sizeof(dlist.act));
unlrg( dlrd );
return(1);

/* End wrt_con2() */

```

WHC-SD-534-CSWL-001 REV 3

UKF43

WHC-SD-534-CSWL-001 REV 3

```

/*      U K F 4 3 . C                      VERSION 5
*
*
*
* General Function 43 - For standard alarm inhibiting during Plant
* shutdown. UKF43 is loaded only to the CDCM,
* as are UKF34 (de-inhibiting), and UKF14 (alarm
* regeneration). Due to the way the ALMOMG task
* works, the following user key functions must
* be run after GF 43 in the order presented:
* GF 14 (to regenerate alarms).
*
*
*
* Written by : M. C. Teats
* Date       : 03-08-93
* Facility   : 242-A Evaporator Bldg.
* Run on Unit: CDCM only
* Machine    : VAX 4104-A
* Compiler   : DEC VAX C V3.3
* C .cmd use : SET DEF D3SOURCE::[261,10]
* C .cmd use : BLD UKF14 UKF14.C -L -MOV
* Op. System : VAX/VMS 5.5-2AH
* D/3 Version: 6.3      BASELINE
*
* * MODIFICATION LOG:
*
* * 08-MAR-93      Define program for D/3 version 3.3      M. Teats
* * 07-MAY-93      Modify ipns per CCN# 93MAR25-01 (search M. Teats
* *                  MCT1).
* *
* * 05-JAN-95      Rebaselined the program for version      J. Berger
* *                  6.3 upgrade of the D/3 software. Added
* *                  additional EPN to the list of EPNs to be
* *                  inhibited. See rebaseline documents for
* *                  the 6.3 upgrade.
* *
* * 27-APR-95      Added PI-CA1-20, YS-PB1-1, YS-P-C100      J. Berger
* *                  and YS-AW102. See WP EE-95-00006 and
* *                  ECN 622499. (JFB2)
* *
* * 26-SEP-95      Changed program so multiple general      J. Berger
* *                  functions do not have to be run.
* *                  See ECN 623938 & EE-95-00006.
* *
* *
* * END OF MOD LOG
*/

#include <emcstd.h>
#include <d3sys.h>
#include <unitdb.h>
#include <grinnell.h>
#include <dbatypes.h>
#include <dbaai.h>
#include <dbadin.h>
#include <dbadev.h>
#include <digdef.h>
#include <conkey.h>
#include <conf.h>
#include <err.h>
#include <keyequ.h>
#include <func.h>
#include <seqdba.h>
#include <almmsg.h>
#include <processes.h>

#define UDF43      43
#define DONE      1

```

WHC-SD-534-CSWL-001 REV 3

```
#pragma nostandard
PUBLIC int dldr = 0; /* DLIST descriptor */
PUBLIC struct dlist dlist = 0; /* DLIST structure */
PUBLIC struct gcb gcb = 0; /* Struct for Grinnel Control Block */
PUBLIC short con=0;
#pragma standard

/*
 * DEFINE STRUCTURE FOR NODE STATUS:
 */
struct unitstat ustat =0; /* For Micro Status determination */

/*
 * DEFINE STRUCTURES FOR EPN's TO INHIBIT:
 */
struct s1
(
    char pcm, /* PCM unit number */
    dbt; /* DBASE type */
    int sbi; /* Scan block ipn */
);

static struct s1 inha [] =
(
    /* k pcm dbt sbi epn */
    /* 0 2, AI, 45, pi-ca1-11 */
    /* 1 2, AI, 46, pic-ca1-7 */
    /* 2 2, AI, 53, lic-ca1-1 */
    /* 3 2, AI, 57, lic-ca1-2 */
    /* 4 2, AI, 63, li-ca1-3 */
    /* 5 2, AI, 68, fic-ca1-6 */
    /* 6 2, AI, 81, fi-ca1-7 */ /* jfb1 */
    /* 7 2, AI, 83, vi-pb1-2a */
    /* 8 2, AI, 84, vi-pb1-1a */
    /* 9 2, AI, 85, vi-pb1-3a */
    /* 10 2, AI, 86, fi-ca1-1 */
    /* 11 2, AI, 87, pi-ca1-9 */
    /* 12 2, AI, 89, fi-ca1-3 */
    /* 13 2, AI, 97, pi-ca1-20 */ /* JFB2 */
    /* 14 2, AI, 102, sic-pb2-1 */ /* JFB1 */
    /* 15 2, AI, 103, vi-pb2-1a */
    /* 16 2, AI, 106, fi-ca1-2 */
    /* 17 2, AI, 106, pi-ca1-10 */
    /* 18 2, AI, 110, fic-ca1-4 */
    /* 19 2, AI, 130, pi-ea1-14 */ /* JFB1 */
    /* 20 2, AI, 131, fic-ea1-1 */
    /* 21 2, AI, 153, pi-ec1-4 */ /* MCT1 */
    /* 22 2, AI, 154, pi-ec2-2 */ /* MCT1 */
    /* 23 2, AI, 162, fic-ec3-1 */ /* MCT1 */
    /* 24 2, AI, 197, pic-ec1-2 */
    /* 25 2, AI, 198, pi-ec1-9 */ /* JFB1 */
    /* 26 2, DIN, 50, ys-aw102 */ /* JFB2 */
    /* 27 2, DIN, 60, psh-ca111 */
    /* 28 2, DIN, 73, ys-pb1-1 */ /* JFB2 */
    /* 29 2, DIN, 87, TSH-PB2-1 */ /* JFB1 */
    /* 30 2, DIN, 90, SKS-PB2-1 */ /* JFB1 */
    /* 31 2, DIN, 91, YS-VFD-1 */ /* JFB1 */
    /* 32 2, DIN, 92, SSL-PB2-1 */ /* JFB1 */
    /* 33 2, DIN, 107, ZS-EA1-1C */ /* JFB1 */
    /* 34 2, DIN, 163, YS-P-C100 */ /* JFB2 */
    /* 35 2, DEV, 102, RC2-P1G */ /* JFB1 */
    /* 36 2, DEV, 133, RC3-P1G */ /* JFB1 */
    /* 37 2, DEV, 143, RC1-P1G */ /* JFB1 */
);

static char type = 0; /* For PCM processor type, status */

*****
*
* M A I N ( UKF43 )
*
```

```

*****/

main( )
(
    char net, /* Network number */
    unamet[ UNAMSIZ+1 ], /* Unit name buffer */
    epnt[ EPNSIZ+1 ], /* EPN buffer */
    status[36], /* Temp string for status */
    pcm_node[64], /* Table of PCM's completed or failed */
    backup, /* PCM backup unit number */
    anydone, /* Any Nodes done ? */
    buff[ MAX_DB_RECSIZ ], /* Largest db block */
    field; /* Alarm count field */
    int dbreg, /* DBASE region id */
    creg, /* Config region id */
    areg, /* Alarm que region id */
    unit, /* PCM unit number */
    npcm, /* Maximum PCM unit number = 3 */
    value, /* Alarm count field's bit value */
    k, /* For loop counter */
    err, /* Error code for dbase access calls */
    con, /* Console number */
    i; /* Miscellaneous variables */

    int exit_code;
    char buffer[80];
    net = 0;
    npcm = 3;

    con = gtnum();
    if( con < 0 || con > 3 )
    {
        con = 0;
        printf("\n-- UKF43 - ALARM INHIBIT ( Invoked - Interactive Mode )\n" );
    }
    else /* Else - Invoked by Gen Func 43 at console */
    {
        printf("\n-- UKF43 - ALARM INHIBIT Invoked from Console %d \n", con );
        wrt_con1( con );
    } /* End else from CONSOLE */

/*
 * SECOND, OPEN A COMMUNICATION CHANNEL TO THE PCM:
 */
if ( net_saps(FALSE,TRUE) < 0 )
{
    printf("\n -- UKF43 -- CAN'T OPEN COMMUNICATIONS, net_saps- TASK EXITING\n");
    unlrg( creg );
    unlrg( dbreg );
    unlrg( areg );
    exit();
}

/*
 * FIRST, LINK TO THE CONFIG TABLE, DATABASE, AND ALARM QUE REGIONS:
 */
if( creg = lnrg("CONFIG") < 0 ) /* Link to config */
{
    printf("\n -- UKF43 -- CAN'T LINK TO CONFIG - TASK EXITING --\n");
    exit();
}
if( dbreg = lnrg("DBASE") < 0 ) /* Link to database */
{
    printf("\n -- UKF43 -- CAN'T LINK TO DBASE - TASK EXITING --\n");
    unlrg( creg );
    exit();
}
if( areg = lnrg("ALMQUE") < 0 ) /* Link to alarm que */
{
    printf("\n -- UKF43 -- CAN'T LINK TO ALMQUE - TASK EXITING --\n");
    unlrg( creg );
    unlrg( dbreg );
    exit();
}

```



```

    )
/*
 *   INITIALIZATION:
 */
anydone = 0;
unit = -1;
for( i=0; i<sizeof(pcm_node); i++ )
    pcm_node[i] = 0;

/*
 *   MAIN LOOP -
 *   WHILE PCMOA OR PCMOB (unit 2 or 3) ON NETWORK 0:
 */
while( ++unit <= npcm )
{
    /*
     *   GET UNIT NAME FOR NEXT PCM:
     */
    for( type=0; type != UPCM; unit++ )
    {
        if( unit > npcm ) goto quit;
        fill( uname,sizeof(uname),'\0' );

        err = idpcm( creg, uname, net, unit );
        if( err < 0 )
        {
            if( err == ER_SND || err == ER_UNF || err == ER_UNL )
                continue;
            printf("UKF43 - CAN'T GET UNIT NAME FOR UNIT %d, err=%d\n",unit,err);
            goto quit;
        }
        if( err = pcmid( creg, uname, &net, &unit, &type, &backup) < 0 )
        {
            printf("UKF43 - CAN'T GET UNIT TYPE FOR UNIT %d, err=%d\n",unit,err);
            goto quit;
        }
        if( type == UPCM )
        {
            /*
             *   HAS THE PRIMARY UNIT ALREADY BEEN DONE ?
             */
            if( (unit>backup) && (backup!= -1) && (pcm_node[backup] == DONE) )
            {
                printf( "UKF43 - %s IS BACKUP UNIT (BYPASSED) \n", uname );
                type = 0;
                continue;
            }
            err = gsstat( creg, net, unit, &ustat );

            if( ustat.us_pro != STOK )
                type = 0;

            /*
             *   PROCESSOR STATUS:
             */
            switch( ustat.us_pro )
            {
                case( STOFF ):    cpystr( status, "offline", 0 );
                                break;
                case( STL00 ):    cpystr( status, "load request", 0 );
                                break;
                case( STIDL ):    cpystr( status, "idle", 0 );
                                break;
                case( STOK ):     cpystr( status, "running", 0 );
                                break;
                default:          cpystr( status, "UNKNOWN", 0 );
                                break;
            }
            if( ustat.us_pro != STOK )
                printf( "UKF43 - %s IS %s (BYPASSED) \n", uname, status );
            else printf( "UKF43 - %s IS %s \n", uname, status );
        }
    }
}

```

WHC-SD-534-CSWL-001 REV 3

```

/*
 *      END PROCESSOR STATUS
 */
}
/* End If ( type UPCM )
/* End For ( unit++ ) loop
unit--;
anydone = 1;
if( backup == -1 ) backup = unit;
pcm_node[ unit ] = pcm_node[ backup ] = DONE;
/*
 *
 *
 *      EPN INHIBIT LOOP -
 *
 *      NOW WE ARE READY TO INHIBIT ALARMS:
 *      err = idepn(rid, &epn, net, unit, dbtype, ipn)
 *      err = grb( net, unit, dbtype , ipn, &buff)
 *      err = gfb( dbtype, field, &value, &buff)
 *      err = pfb( dbtype, field, &value, &buff)
 *      err = prb( net, unit, dbtype , ipn, &buff)
 */
for ( k = 0; k < 38; k++)      /* JFB1 */
{
    field = AI_ALMCT;
    if ( k > 25)                /* JFB1 */
    {
        field = DIN_ALMCT;
    }
    if ( k > 34)                /* JFB1 */
    {
        /* JFB1 */
        field = DEV_ALMCT;
        /* JFB1 */
        /* JFB1 */
    }
    err = idepn(dbreg, epn, net, inha[k].pcm, inha[k].dbt, inha[k].sbi);
    if(err < 0)
    {
        printf("\n --UKF43 IDEPN--ERROR = %d\n",err);

        unlg( creg );
        unlg( dbreg );
        unlg( areg );
        exit();
    }
    err = grb(net, inha[k].pcm, inha[k].dbt, inha[k].sbi, buff);
    if(err < 0)
    {
        printf("\n --UKF43 %s GRB--ERROR = %d\n", epn, err);

        unlg( creg );
        unlg( dbreg );
        unlg( areg );
        exit();
    }
    err = gfb(inha[k].dbt, field, &value, buff);
    if(err < 0)
    {
        printf("\n -- UKF43 %s GFB--ERROR = %d\n", epn, err);

        unlg( creg );
        unlg( dbreg );
        unlg( areg );
        exit();
    }
}
/*
 *      CHECK IF INHIBITED. IF NOT, DO IT AND SEND MESSAGE!
 */
if ( value != 16 )
{
    /* If not inhibited
    value = 16;
    err = pfb(inha[k].dbt, field, &value, buff);
    /* Set to inhibit
    if(err < 0)
    {
        printf("\n -- UKF43 %s PFB--ERROR = %d\n", epn, err);
    }
}

```

```

        unlrg( creg );
        unlrg( dbreg );
        unlrg( areg );
        exit();
    }
    sndmsg (0,0,areg,creg,3,MSG_TEXT,NONCRIT,"--- UKF43 INHIBITED EPN %s ",epn);
    sleep (.2);
}                                     /* End If */

/*
 *   GET OUT AND GO TO NEXT EPN:
 */
err = prb(net, inha[k].pcm, inha[k].dbt, inha[k].sbi, buff);
if(err < 0)
{
    printf("\n --UKF43 %s PRB--ERROR = %d\n", epn, err);

    unlrg( creg );
    unlrg( dbreg );
    unlrg( areg );
    exit();
}

sleep (1);                          /* Take a rest */
}                                     /* End inhibit for-loop, k++ */
/* END while <= npcm */

/*
 *   UNLINK REGIONS AND GET OUT:
 */
quit:
unlrg( creg );
unlrg( dbreg );
unlrg( areg );
if( anyone )
{
    sleep ( 5 );

    con = gtnum();
    if( con < 0 || con > 3 )
    {
        con = 0;
        printf("\n-- UKF43 - ALARM INHIBIT ( Invoked - Interactive Mode )\n" );
    }
    else
    {
        /* Else - Invoked by Gen Func 43 at console */
        printf("\n-- UKF43 - ALARM INHIBIT Invoked from Console %d \n", con );
        wrt_con2( con );
        /* End else from CONSOLE */
    }

    /*
    system("run d3source:[004055]ukf14.exe");
    */

    printf( "UKF43 - ALARM INHIBIT COMPLETED \n" );
}
else
    printf( "UKF43 - ALARM INHIBIT WARNING - ALL PCM NODES OFFLINE \n" );
return(1);

}                                     /* End main */

/*****
 *
 *   END MAIN
 *
 *****/

/*
 *   WRT_CONS1 ( )
 *
 *   Draw the General Function 43 - Display
 */
int wrt_con1( con )

```

[illegible]

[illegible]

WHC-SD-534-CSWL-001 REV 3

```
putrg(dlrd,&dlist.ukf,OFFSET(dlist,ukf),sizeof(dlist.ukf));  
putrg(dlrd,&dlist.act,OFFSET(dlist,act),sizeof(dlist.act));  
unlrg( dlrd );  
return(1);
```

```
/* End wrt_con2) */
```

WHC-SD-534-CSWL-001 REV 3

UKF44

WHC-SD-534-CSWL-001 REV 3

```

/*
*
*
*
*
* General Function 44 - General Function 44 is to be used only on
* DCM0 following GF 43 (alarm inhibiting). GF 44
* clears the alarm summary pages on DCM0. The
* alarm summary pages on the CDCM are cleared by
* GF43, which spawns the mcr command file to reset
* the summary pages.
*
*
* After GF 44 is run on DCM0, GF 14 is then run on
* the CDCM to regenerate alarms.
*
*
*
*
* Written by : M. C. Teats
* Date : 03-08-93
* Facility : 242-A Evaporator Bldg. */
/* Modified by: John Hayden T.I. 6.3-2 Upgrade */
/* Facility : 242-A Evaporator Bldg. */
/* Machine : VAX 4104-A */
/* Compiler : DEC VAX C V3.3 JH01 */
/* C .cmd use : SET DEF D3SOURCE::[261,10] JH01 */
/* C .cmd use : BLD UKF14 UKF14.C -L -MOV JH01 */
/* Op. System : VAX/VMS 5.5-2AH JH01 */
/* D/3 Version: 6.3-2 BASELINE JH01 */
/* #INCLUDES CHANGED JH01 */
/* PARAMETERS TO FUNCTIONS CHANGED JH01 */
/* PRAGMA FOR DLIST DECLARATIONS, INITIALIZE JH01 */
/* SLEEP NOW SECONDS INSTEAD OF TICKS JH01 */
/* XP NOT XS, PUTFMT-> PRINTF,XI->XD JH01 */
/* Run on Unit: DCM0 only
*
*
* Build command file : @[261,10]UKF44
*
*
* MODIFICATION LOG:
*
* 08-MAR-93 Define program for D/3 version 3.3 M. Teats
*
* 05-JAN-95 Rebaselined software for upgrade of J. Berger
* D/3 software to version 6.3.
*
* END OF MOD LOG
*/

#include <emcstd.h>
#include <d3sys.h>
#include <unitdb.h>
#include <dbatypes.h>
#include <grinnell.h>
#include <dbaal.h>
#include <dbadln.h>
#include <digdef.h>
#include <conkey.h>
#include <conf.h>
#include <err.h>
#include <keyequ.h>
#include <func.h>
#include <almsg.h>

#define UDF44 44
#define DONE 1
#pragma nostandard
PUBLIC int dlrd =0; /* DLIST descriptor */
PUBLIC struct dlist dlst =0; /* DLIST structure */
PUBLIC struct gcb gcb =0; /* Struct for Grinnel Control Block */
PUBLIC short con = 0;
#pragma standard

```


WHC-SD-534-CSWL-001 REV 3

```

/*****
 *
 * M A I N ( UKF44 )
 *
 *****/

main( )
(
    int err,                                /* Error code */

    con = gtnum();
    if( con < 0 || con > 3 )
    {
        con = 0;
        printf("\n-- UKF44 - ALARM SUMMARY PAGE CLEARING ( Invoked - Interactive Mode )\n" );
    }
    else
        /* Else - Invoked by Gen Func 44 at console */
        {
            printf("\n-- UKF44 - ALARM SUMMARY PAGE CLEARING Invoked from Console %d \n", con );
            wrt_console( con );
        }
        /* End else from CONSOLE */

/*
 * SPAWN RESET.CMD TO CLEAR ALARM SUMMARY PAGES ON THIS DCM
 */

/*
    sprintf(buffer, "d3com:pcmlod %s all", node_name);
    exit_code = d3spawn(buffer, 1, 0);
    if (exit_code == ER_GEN)
        printf("Error spawning PCMLOD\n");
    sleep(3);
    printf("%-80s", " ");

*/

    err = d3spawn("SUBMIT D3SOURCE:[261,10]RESET.COM",1,0);
    if(err == ER_GEN)
        printf("error spawning\n");

    sleep (180);

    con = gtnum();
    if( con < 0 || con > 3 )
    {
        con = 0;
        printf("\n-- UKF44 - ALARM SUMMARY PAGE CLEARING ( Invoked - Interactive Mode )\n" );
    }
    else
        /* Else - Invoked by Gen Func 44 at console */
        {
            printf("\n-- UKF44 - ALARM SUMMARY PAGE CLEARING Invoked from Console %d \n", con );
            wrt_cons( con );
        }
        /* End else from CONSOLE */

)

/*****
 *
 * E N D M A I N
 *
 *****/

/*
 * W R T _ C O N S O L E ( )
 *
 * Draw the General Function 44 - Display
 */
int wrt_console( con )
int con;

```

```

/*
 *
 */
int kbfd; /* Key board fd */
int kboutfd; /* Keyboard output fd */
int err; /* Error code */
static char dlname[] = "DLISTxx";

dlname[5 + itob( &dlname[5], con ,10)] = 0;
if((dlrd = lnrg(dlname)) < 0)
{
    printf("UKF44 - DLIST LINK FAILURE %d\n",dlrd);
    unrg(dlrd);
    exit();
}
if((err = ginit(&gcb, con)) < 0 )
{
    printf("UKF44 - CANT ACCESS GRINNELL, ERR %d\n",err );
    unrg(dlrd);
    exit();
}
dlist.act = DMUSER + UDF44;
putrg(dlrd, &dlist.act, OFFSET(dlist,act), sizeof(dlist.act) );

/*
 *
 */
CLEAR the OCM :

gsetup(&gcb,BLACK);
erase(&gcb);
gflush(&gcb);

/* OCM GRAPHICS : */

gsetup(&gcb,RED); /* SCREEN FORMAT */
cursor(&gcb,10,425); /*
horiz( &gcb,440,3); /* General Function 44 */
cursor(&gcb,447,425); /*
vert( &gcb,-125,3); /* Clearing Alarm Pages.
cursor(&gcb,449,300); /*
horiz( &gcb,-439,3); /* Wait for COMPLETE
cursor(&gcb,10,300); /*
vert( &gcb,125,3); /* Then run GF 14 on CDCM.
cursor(&gcb,10,390); /*
horiz( &gcb,440,3); /*

gsetup(&gcb, WHITE );
dub(char(&gcb);
cursor(&gcb,HCHAR*3,VCHAR*40);
g_puts( &gcb, "GENERAL FUNCTION 44" );
gflush(&gcb);
gsetup( &gcb, YELLOW );
widechar(&gcb);
cursor(&gcb,HCHAR*3,VCHAR*36);
g_puts( &gcb, "CLEARING ALARM PAGES FOR THIS UNIT." );
gflush(&gcb);
g_puts( &gcb, " " );
gflush(&gcb);
gsetup( &gcb, RED );
cursor(&gcb,HCHAR*3,VCHAR*34);
g_puts( &gcb, "WAIT FOR COMPLETE " );
gflush(&gcb);
gsetup( &gcb, GREEN );
cursor(&gcb,HCHAR*3,VCHAR*32);
g_puts( &gcb, "THEN RUN GENL FCTN 14 AT CDCM CONSOLE." );
gflush(&gcb);
gsetup(&gcb, WHITE );
cursor(&gcb,HCHAR*72,VCHAR*29);
g_puts( &gcb, " " );
gflush(&gcb);

/*
 *
 */
RELEASE THE CONSOLE AND DLIST REGION:

dlist.ukf = -1;

```

```

dlist.act = 0;
putrg(dlrd,&dlist.ukf,OFFSET(dlist,ukf),sizeof(dlist.ukf));
putrg(dlrd,&dlist.act,OFFSET(dlist,act),sizeof(dlist.act));
unlrg( dlrd );
return(1);
exit();
)

/* End wrt_console() */

/*
 * W R T _ C O N S ( )
 *
 * Draw the General Function 44 - Display
 */
int wrt_cons( con )
int con;
(
    int kbfd; /* Key board fd */
    int kboutfd; /* Keyboard output fd */
    int err; /* Error code */
    static char dlname[] = "DLISTxx";

    dlname[5 + itob( &dlname[5], con ,10)] = 0;
    if((dlrd = lnkrg(dlname)) < 0)
    (
        printf("UKF44 - DLIST LINK FAILURE %d\n",dlrd);
        unlrg(dlrd);
        exit();
    )
    if((err = ginit(&gcb, con)) < 0 )
    (
        printf("UKF44 - CANT ACCESS GRINNELL, ERR %d\n",err );
        unlrg(dlrd);
        exit();
    )
    dlist.act = DMUSER + UDF44;
    putrg(dlrd, &dlist.act, OFFSET(dlist,act), sizeof(dlist.act) );

/*
 *
 */
    C L E A R the O C M :

    gsetup(&gcb,BLACK);
    erase(&gcb);
    gflush(&gcb);

    /* O C M G R A P H I C S : */

    gsetup(&gcb,RED); /*
    cursor(&gcb,10,425); /*
    horiz( &gcb,440,3); /*
    cursor(&gcb,447,425); /*
    vert( &gcb,-125,3); /*
    cursor(&gcb,449,300); /*
    horiz( &gcb,-439,3); /*
    cursor(&gcb,10,300); /*
    vert( &gcb,125,3); /*
    cursor(&gcb,10,390); /*
    horiz( &gcb,440,3); /*

    SCREEN FORMAT
    General Function 44
    -----
    COMPLETE COMPLETE
    -----
    COMPLETE COMPLETE
    -----
    Now run GF 14 on CDCM.

    gsetup(&gcb, WHITE );
    dublchar(&gcb);
    cursor(&gcb,HCHAR*3,VCHAR*40);
    g_puts( &gcb, "GENERAL FUNCTION 44" );
    gflush(&gcb);
    gsetup( &gcb, YELLOW );
    widechar(&gcb);
    cursor(&gcb,HCHAR*3,VCHAR*36);
    g_puts( &gcb, "COMPLETE COMPLETE " );
    gflush(&gcb);
    g_puts( &gcb, " " );
    gflush(&gcb);
    gsetup( &gcb, RED );

```

WHC-SD-534-CSWL-001 REV 3

```

cursor(&gcb,HCHAR*3,VCHAR*34);
g_puts( &gcb, "COMPLETE COMPLETE " );
gflush(&gcb);
gsetup( &gcb, GREEN );
cursor(&gcb,HCHAR*3,VCHAR*32);
g_puts( &gcb, "NOW RUN GENL FCTN 14 AT CDCM CONSOLE." );
gflush(&gcb);
gsetup(&gcb, WHITE );
cursor(&gcb,HCHAR*72,VCHAR*29);
g_puts( &gcb, " " );
gflush(&gcb);
/*
 *
 */
RELEASE THE CONSOLE AND DLIST REGION:

dlist.ukf = -1;
dlist.act = 0;
putrg(dlrd,&dlist.ukf,OFFSET(dlist,ukf),sizeof(dlist.ukf));
putrg(dlrd,&dlist.act,OFFSET(dlist,act),sizeof(dlist.act));
unlrg( dlrd );
return(1);
exit();
)
/* End wrt_cons() */

```

APPENDIX G - STRING PACKET LISTING

Listing of string.out Created using SED from CDCM D3DATA:[100010]STRING.BIN

WHC-SD-534-CSWL-001 REV 3

247

[0]
 <0>
 <1> {ALARM (UP)
 <2> ALARM
 <3> {ALARM (UP)
 <4> ALARM
 <5> {ALARM (UP)
 <6> ALARM

[1]
 <0>
 <1> {LOW FLOW
 <2> LOW FLOW
 <3> LOW FLOW
 <4> LOW FLOW
 <5> LOW FLOW
 <6> LOW FLOW

[2]
 <0>
 <1> {50 LB INSTRUMENT AIR LOW
 <2> 50 LB INSTRUMENT AIR LOW
 <3> 50 LB INSTRUMENT AIR LOW
 <4> 50 LB INSTRUMENT AIR LOW
 <5> 50 LB INSTRUMENT AIR LOW
 <6> ALARM

[3]
 <0>
 <1> {ERROR
 <2> ERROR
 <3> ERROR
 <4> ERROR
 <5> ERROR
 <6> ALARM

[4]
 <0>
 <1> {LOW
 <2> LOW
 <3> LOW
 <4> LOW
 <5> LOW
 <6> LOW

[5]
 <0>
 <1> {ANNULUS EXH RAD HI/FAIL
 <2> ANNULUS EXH RAD HI/FAIL
 <3> ANN EXH RAD HI/FAIL INHA
 <4> {ANN EXH RAD HI/FAIL OSCN
 <5> ANN EXH RAD HI/FAIL IBAD

[6]
 <0>
 <1> {LEAK PIT RAD HI/FAIL
 <2> LEAK PIT RAD HI/FAIL
 <3> L-PIT RAD HI/FL INHA
 <4> {L-PIT RAD HI/FL OSCN
 <5> L-PIT RAD HI/FL IBAD

[7]
 <0>
 <1> {ALARM
 <2> ALARM
 <3> {ALARM
 <4> ALARM
 <5> {ALARM
 <6> ALARM

[8]
 <0>
 <1> {SERVICE PIT RAD HI/FAIL
 <2> SERVICE PIT RAD HI/FAIL
 <3> SERPIT RAD DET INHA
 <4> {SER PIT RAD DET OSCN
 <5> SER PIT RAD DET IBAD

[9]
 <0>
 <1> {SERVICE PIT LEAK
 <2> SERVICE PIT LEAK
 <3> SERPIT LEAK INHA
 <4> {SERPIT LEAK OSCN
 <5> SERPIT LEAK IBAD

[10]
 <0>
 <1> {PROCESS PITS LEAK
 <2> PROCESS PITS LEAK
 <3> PRO PIT LEAK INHA
 <4> {PRO PIT LEAK OSCN
 <5> PRO PIT LEAK IBAD

[11]
 <0>
 <1> {SL-160 SL-168 LEAK
 <2> SL-160 SL-168 LEAK
 <3> SL-160 SL-168 INHA
 <4> {SL160/168 OFF SCAN
 <5> SL160/168 BAD INPUT

[12]
 <0>
 <1> {SN-260 SN-268 LEAK
 <2> SN-260 SN-268 LEAK
 <3> SN260/268 INHA
 <4> {SN-260/268 OSCN
 <5> SN260/268 BAD INPUT

[13]
 <0>
 <1> {CLEANOUT BOX LEAK
 <2> CLEANOUT BOX LEAK
 <3> COB LEAK INHA
 <4> {COM LEAK DIS
 <5> COB LEAK IBAD

WHC-SD-534-CSWL-001 REV 3

[14]
 <0>
 <1> {SN FLUSH LINE PRESSURE HI
 <2> SN FLUSH LINE PRESSURE HI
 <3> SN FLUSH LINE PRES INHA
 <4> {SN FLUSH LINE PRESS DIS
 <5> SN FLUSH LINE PRESSURE IBAD

[15]
 <0>
 <1> {AN DR-368 ENCASE LEAK
 <2> AN DR-368 ENCASE LEAK
 <3> AN368 ENCASE INHA
 <4> {AN368 ENCASE DIS
 <5> AN368 ENCASE BAD INPUT

[16]
 <0> 271 BUILDING POWER NORM
 <1> {271 BUILDING POWER LOSS
 <2> 271 BUILDING POWER LOSS
 <3> BLD POWER LOSS ALM INHA
 <4> {BLD POWER LOSS DIS
 <5> BLD POW LOSS BAD INPUT

[17]
 <0> 271 BUILDING RADIATION NORMAL
 <1> {271 BUILDING RADIATION HI/FAIL
 <2> 271 BUILDING RADIATION HI/FAIL
 <3> 271 BUILDING RADIATION INHA
 <4> {271 BUILDING RADIATION DIS
 <5> BLD RAD ALARM BAD INPUT

[18]
 <0> 241 AREA RADIATION NORMAL
 <1> {241 AREA RADIATION HI/FAIL
 <2> 241 RADIATION HI/FAIL
 <3> 241 RAD HI/FAIL INHA
 <4> {241 RAD HI/FAIL DIS
 <5> 241 RAD HI/FAIL BAD INPUT

[19]
 <0> GENERAL ALARM OFF
 <1> {GENERAL ALARM ON
 <2> GENERAL ALARM ON
 <3> GENERAL ALM INHA
 <4> {GENERAL ALM DIS
 <5> GENERAL ALM IBAD

[20]
 <0> RADIATION BYPASS OFF
 <1> {AREA RADIATION BYPASS
 <2> AREA RADIATION BYPASS
 <3> AREA RAD BYPASS INHA
 <4> {AREA RAD BYPASS DIS
 <5> AREA RAD BYPASS IBAD

[21]
 <0> MASTER SHUTDOWN OFF
 <1> {FARM MASTER SHUTDOWN
 <2> FARM MASTER SHUTDOWN
 <3> MSTR SHUTDOWN INHA
 <4> {MSTR SHUTDOWN DIS
 <5> MSTR SHUTDN BAD INPUT

[22]
 <0>
 <1> {TANK EXH RAD HI
 <2> TANK EXH RAD HI
 <3> TANK EXHRD INHA
 <4> {TANK EXHRD DIS
 <5> TANK EXHRD IBAD

[23]
 <0> STACK MONITOR SAMPLE PUMP ON
 <1> {STACK MONITOR SAMPLE PUMP OFF
 <2> STACK MONITOR SAMPLE PUMP OFF
 <3> STACK MONITOR SAMPLE PMP INHA
 <4> {STACK MONITOR SAMPLE PMP DIS
 <5> STACK MONITOR SAMPLE PMP IBAD

[24]
 <0> RECORD SAMPLE PUMP ON
 <1> {RECORD SAMPLE PUMP OFF
 <2> RECORD SAMPLE PUMP OFF
 <3> RECORD SAMPLE PMP INHA
 <4> {RECORD SAMPLE PMP DIS
 <5> RECORD SAMPLE PMP IBAD

[25]
 <0>
 <1> {TANK EXH RAD MON FAIL
 <2> TANK EXH RAD MON FAIL
 <3> TANK EXH RAD MON INHA
 <4> {TANK EXH RAD MON DIS
 <5> TANK EXH RAD MON IBAD

[26]
 <0> TANK EXH FANS ON
 <1> {TANK EXH FANS OFF
 <2> TANK EXH FANS OFF
 <3> TANK EXH FAN INHA
 <4> {TANK EXH FAN DIS
 <5> TANK EXH FAN IBAD

[27]
 <0> ANN EXH RAD HI/FAIL FAN NORM
 <1> {ANN EXH RAD HI/FAIL FAN FAIL
 <2> ANN EXH RAD HI/FAIL FAN FAIL
 <3> ANN EXH RAD/FAN HI/FAIL INHA
 <4> {ANN EXH RAD/FAN HI/FAIL DIS
 <5> ANN EXH RAD/FAN HI/FAIL IBAD

WHC-SD-534-CSWL-001 REV 3

[28]
 <0> ANNULUS EXHAUST RADIATION MONITOR NORMAL
 <1> |ANNULUS EXHAUST RADIATION MONITOR FAILURE
 <2> ANNULUS EXHAUST RADIATION MONITOR FAILURE
 <3> ANN EXHAUST RAD MONITOR FAILURE INHA
 <4> |ANN EXHAUST RAD MONITOR FAILURE DIS
 <5> ANN EXHAUST RAD MONITOR FAILURE BAD INPUT

[29]
 <0> ANNULUS EXH RADIATION OK
 <1> |ANNULUS EXH RADIATION HI
 <2> ANNULUS EXH RADIATION HI
 <3> |ANN EXH RAD MON FAILURE
 <4> ANN EXH RAD MON FAILURE
 <5> ANN EXH RAD MON INHIBIT
 <6> ANN EXH RAD MON OFF SCAN
 <7> ANN EXH RAD MON BAD INPT

[30]
 <0>
 <1> |LEAK PIT RADIATION HI
 <2> LEAK PIT RADIATION HI
 <3> LEAK PIT RAD MON INHA
 <4> LEAK PIT RAD OFF SCAN
 <5> LEAK PIT RAD BAD INPT
 <6>
 <7>

[31]
 <0> DRAIN PIT LEAK
 <1> |DRAIN PIT LEAK
 <2> DRAIN PIT LEAK
 <3> DRN PIT LK INHA
 <4> |DRN PIT LK DIS
 <5> D PIT LK BD IPT

[32]
 <0> SERVICE PIT MONITOR NORMAL
 <1> |SERVICE PIT MONITOR FAILURE
 <2> SERVICE PIT MONITOR FAILURE
 <3> SERVICE PIT MONITOR INHA
 <4> |SERPIT MON FAILURE DIS
 <5> SER PIT MON FAIL BAD INPUT

[33]
 <0> RADIATION MONITOR NORMAL
 <1> |RADIATION MONITOR FAILURE
 <2> RADIATION MONITOR FAILURE
 <3> RAD MON FAIL ALM INHIBIT
 <4> RAD MON FAIL OFF SCAN
 <5> RAD MON FAIL BAD INPUT

[34]
 <0>
 <1> |RADIATION HIGH
 <2> RADIATION HIGH
 <3> RADIATION HIGH
 <4> RADIATION HIGH
 <5> RADIATION HIGH

[35]
 <0> SL-509/SL-510

[36]
 <0>
 <1> |SL-509/SL-510 LEAK DETECTED
 <2> SL-509/SL-510 LEAK DETECTED
 <3> SL-509/SL-510 LEAK DET INHA
 <4> SL-509/SL-510 LEAK DET OSCN
 <5> SL-509/SL-510 LEAK DET IBAD

[37]
 <0>
 <1> |SN-609/SN-610 LEAK DETECTED
 <2> SN-609/SN-610 LEAK DETECTED
 <3> SN-609/SN-610 LEAK DET INHA
 <4> SN-609/SN-610 LEAK DET OSCN
 <5> SN-609/SN-610 LEAK DET IBAD

[38]
 <0>
 <1> |SN-621 LEAK DETECTED
 <2> SN-621 LEAK DETECTED
 <3> SN-621 LEAK DET INHA
 <4> SN-621 LEAK DET OSCN
 <5> SN-621 LEAK DET IBAD

[39]
 <0>
 <1> |SN-650 ENCASEMENT LEAK DETECTED
 <2> SN-650 ENCASEMENT LEAK DETECTED
 <3> SN-650 ENCASEMENT LEAK DET INHA
 <4> SN-650 ENCASEMENT LEAK DET OSCN
 <5> SN-650 ENCASEMENT LEAK DET IBAD

[40]
 <0> NORMAL
 <1> |FLOW
 <2> FLOW
 <3> INHA
 <4> OSCN
 <5> IBAD

[41]
 <0>
 <1> |FLOW DETECTED
 <2> FLOW DETECTED
 <3> FLOW DET INHA
 <4> FLOW DET OSCN
 <5> FLOW DET IBAD

[42]
 <0> CLOSED
 <1> |OPEN
 <2> OPEN
 <3> INHA
 <4> OSCN
 <5> IBAD

WHC-SD-534-CSWL-001 REV 3

[43]
 <0>
 <1> HIGH PRESSURE
 <2> HIGH PRESSURE
 <3> PSH-APFL1 INHA
 <4> PSH-APFL1 OSCN
 <5> PSH-APFL1 IBAD

[44]
 <0>
 <1> SHUTDOWN
 <2> SHUTDOWN
 <3>
 <4>

[45]
 <0>
 <1> LEAK DETECTED
 <2> LEAK DETECTED
 <3> LEAK DET INHA
 <4> LEAK DET DIS
 <5> LEAK DET IBAD

[46]
 <0> ANN EXHAUST FANS ON
 <1> ANN EXHAUST FANS OFF
 <2> ANN EXHAUST FANS OFF
 <3> ANN EXH FANS INHIBIT
 <4> ANN EXH FANS OFF SCAN
 <5> ANN EXH FANS BAD INPUT

[47]
 <0> MONITOR SAMPLE FLO NORM
 <1> MONITOR SAMPLE LOW FLOW
 <2> MONITOR SAMPLE LOW FLOW
 <3> MON SNPL LOW FLOW INHA
 <4> MON SNPL LOW FLOW OSCN
 <5> MON SNPL LOW FLOW IBAD

[48]
 <0> EXHAUST RADIATION NORM
 <1> EXHAUST RADIATION HIGH
 <2> EXHAUST RADIATION HIGH
 <3> EXH RAD ALARM INHIBITED
 <4> EXH RAD ALARM OFF SCAN
 <5> EXH RAD ALARM BAD INPUT

[49]
 <0> RADIATION MONITOR NORMAL
 <1> RADIATION MONITOR FAILED
 <2> RADIATION MONITOR FAILED
 <3> RAD MON FAIL ALM INHIBIT
 <4> RAD MON FAIL ALARM OSCN
 <5> RAD MON FAIL ALARM IBAD

[50]
 <0> RECORD SAMPLE FLO NORM
 <1> IRECORD SAMPLE LOW NORM
 <2> RECORD SAMPLE LOW FLOW
 <3> REC SAMPLE ALM INHIBIT
 <4> REC SAMPLE ALARM OSCN
 <5> REC SAMPLE ALARM IBAD

[51]
 <0> STACK FLO NORM
 <1> STACK FLOW LOW
 <2> STACK FLOW LOW
 <3> STACK FLOW LOW
 <4> STACK FLOW LOW
 <5> STACK FLOW LOW

[52]
 <0> SEAL-POT PIT NORMAL
 <1> SEAL POT PIT LEAK
 <2> SEAL POT PIT LEAK
 <3> SEAL POT PIT LEAK ALARM INHA
 <4> SEAL POT PIT LEAK ALARM OSCN
 <5> SEAL POT PIT LEAK ALARM IBAD

[53]
 <0>
 <1> FLOW DETECT FAILED
 <2> FLOW DETECT FAILED
 <3> FLOW DET FAIL INHA
 <4> FLOW DET FAIL OSCN
 <5> FLOW DET FAIL IBAD
 <6>

[54]
 <0>
 <1> FAILURE
 <2> FAILURE
 <3> INHIBIT
 <4> OFF SCN
 <5> BAD INP

[55]
 <0> ENCASEMENT
 <1> ENCASEMENT
 <2> ENCASEMENT
 <3> ENCASEMENT
 <4> ENCASEMENT
 <5> ENCASEMENT

[56]
 <0>
 <1> LEAK
 <2> LEAK
 <3> INHA
 <4> OSCN
 <5> IBAD

[57]
 <0>
 <1> VALVE PIT LEAK
 <2> VALVE PIT LEAK
 <3> VALVE PIT INHA
 <4> VALVE PIT OSCN
 <5> VALVE PIT IBAD

WMC-SD-534-CSWL-001 REV 3

[58]

<0> OFF
 <1> ON > HI
 <2> LO
 <3> HIHI
 <4> LOLO
 <5> ROC
 <6> LOOP
 <7> DEV
 <8> OBAD
 <9> RANGE
 <10> OVER
 <11> OC
 <12> IBAD
 <13> UNDR
 <14> TMR
 <15> CNTR
 <16> HI
 <17> LO
 <18> HIHI
 <19> LOLO
 <20> ROC
 <21> LOOP
 <22> DEV
 <23> OBAD
 <24> RANG
 <25> OVER

[59]

<0> HIGH TEMPERATURE
 <1> HIGH TEMPERATURE
 <2> HIGH TEMPERATURE
 <3> HI TEMP DET INHA
 <4> HI TEMP DET OSCN
 <5> HI TEMP DET IBAD

[60]

<0>
 <1> MOTOR HOUSING LEAK
 <2> MOTOR HOUSING LEAK
 <3> MOTOR LEAK INHA
 <4> MOTOR LEAK OSCN
 <5> MOTOR LEAK IBAD

[61]

<0>
 <1> HIGH VIBRATION
 <2> HIGH VIBRATION
 <3> HI VIB DET INHA
 <4> HI VIB DET OSCN
 <5> HI VIB DET IBAD

[62]

<0>
 <1> EXCESSIVE SHAFT STRAIN
 <2> EXCESSIVE SHAFT STRAIN
 <3> EXCESS SHAFT STRN INHA
 <4> EXCESS SHAFT STRN OSCN
 <5> EXCESS SHAFT STRN IBAD

[63]

<0>
 <1> DISCHARGE PRESS LOW
 <2> DISCHARGE PRESS LOW
 <3> DIS PRES LOW INHA
 <4> DIS PRES LOW OSCN
 <5> DIS PRES LOW IBAD

[64]

<0>
 <1> DISCHARGE PRESS HIGH
 <2> DISCHARGE PRESS HIGH
 <3> DIS PRESS HIGH INHA
 <4> DIS PRESS HIGH OSCN
 <5> DIS PRESS HIGH IBAD

[65]

<0>
 <1> HIGH TEMPERATURE
 <2> HIGH TEMPERATURE
 <3> HI TEMP ALM INHA
 <4> HI TEMP ALM OSCN
 <5> HI TEMP ALM IBAD

[66]

<0>
 <1> RADN HIGH/FAIL
 <2> RADN HIGH/FAIL
 <3> RAD HI/FA INHA
 <4> RAD HI/FA OSCN
 <5> RAD HI/FA IBAD

[67]

<0>
 <1> ANNULUS EXHAUST
 <2> ANNULUS EXHAUST
 <3> ANNULUS EXHAUST
 <4> ANNULUS EXHAUST
 <5> ANNULUS EXHAUST

[68]

<0>
 <1> LEAK PIT
 <2> LEAK PIT
 <3> LEAK PIT
 <4> LEAK PIT
 <5> LEAK PIT
 <6>

[69]

<0>
 <1> ALARM (UP)
 <2> ALARM
 <3> ALARM (UP)
 <4> ALARM
 <5> ALARM (UP)
 <6> ALARM
 <7>
 <8>
 <9>
 <10>
 <11>

WHC-SD-534-CSWL-001 REV 3

[70]
 <0>
 <1> |ALARM (DN)
 <2> |ALARM
 <3> |ALARM (DN)
 <4> |ALARM
 <5> |ALARM (DN)
 <6> |ALARM
 <7>
 <8>
 <9>
 <10>
 <11>

[71]
 <0>
 <1> |INTERLOCK
 <2> |INTERLOCK
 <3> |INTERLOCK
 <4> |INTERLOCK
 <5>
 <6>
 <7>

[72]
 <0> |LEAK DETECTION ALARM
 <1> |LEAK DETECTION ALARM
 <2> |LEAK DETECTION ALARM
 <3> |LEAK DETECT INHIBITED
 <4> |LEAK DETECT OFF SCAN
 <5> |LEAK DETECT BAD INPUT
 <6>
 <7>

[73]
 <0> |SEAL POT HI ALM NORM
 <1> |SEAL POT HI ALM FAIL
 <2> |SEAL POT HI ALM FAIL
 <3> |SEAL POT HI ALM INHA
 <4> |SEAL POT HI ALM OSCN
 <5> |SEAL POT HI ALM IBAD
 <6>
 <7>

[74]
 <0> |SEAL POT ALARM NORM
 <1> |SEAL POT HIGH ALARM
 <2> |SEAL POT HIGH ALARM
 <3> |SEAL POT HI ALM INHA
 <4> |SEAL POT HI ALM OSCN
 <5> |SEAL POT HI ALM IBAD
 <6>
 <7>

[75]
 <0> |SEAL POT LOW ALM NORM
 <1> |SEAL POT LOW LEVEL ALM
 <2> |SEAL POT LOW LEVEL ALM
 <3> |SEAL POT LOW ALM INHA
 <4> |SEAL POT LOW ALM OSCN
 <5> |SEAL POT LOW ALM IBAD
 <6>
 <7>

[76]
 <0> |NORMAL
 <1> |GENERAL ALARM
 <2> |GENERAL ALARM
 <3> |GEN ALARM INHA
 <4> |GEN ALARM OSCN
 <5> |GEN ALARM IBAD
 <6>
 <7>

[77]
 <0> |RADIATION NORM
 <1> |RADIATION HIGH
 <2> |RADIATION HIGH
 <3> |RAD DETECT INHA
 <4> |RAD DETECT OSCN
 <5> |RAD DETECT IBAD
 <6>
 <7>

[78]
 <0>
 <1> |LOLO
 <2> |LOLO
 <3> |LO
 <4> |LO
 <5> |HI
 <6> |HI
 <7> |HIHI
 <8> |HIHI
 <9> |INHA
 <10> |OVER
 <11> |OC
 <12> |IBAD
 <13> |UNDR
 <14> |TMR
 <15> |CNTR
 <16> |HI
 <17> |LO
 <18> |HIHI
 <19> |LOLO
 <20> |ROC
 <21> |LOOP
 <22> |DEV
 <23> |OBAD
 <24> |RANG
 <25> |OVER
 <26> |OC
 <27> |IBAD
 <28> |UNDR
 <29> |TMR
 <30> |CNTR
 <31>
 <32>

[79]
 <0> |LIFT STATION LEAK
 <1> |LIFT STATION LEAK
 <2> |LIFT STATION LEAK
 <3> |LEAK DET INHIBIT
 <4> |LEAK DET OFFSCAN
 <5> |LEAK DET BADINPUT
 <6>
 <7>

WHC-SD-534-CSWL-001 REV 3

[80]
<0> DIVERSION BOX LEAK
<1> DIVERSION BOX LEAK
<2> DIVERSION BOX LEAK
<3> DIV BOX LEAK INHA
<4> DIV BOX LEAK OSCN
<5> DIV BOX LEAK IBAD
<6>
<7>

[81]
<0> VALVE LEAK
<1> VALVE LEAK
<2> VALVE LEAK
<3> VLVLK INHA
<4> VLVLK OSCN
<5> VLVLK IBAD
<6>
<7>

[82]
<0>
<1> HI PRESSURE
<2> HI PRESSURE
<3> HI PRES INHA
<4> HI PRES OSCN
<5> HI PRES IBAD
<6>
<7>

[83]
<0> HEAT TRACE TEMPERATURE NORM
<1> HEAT TRACE OVERTEMPERATURE
<2> HEAT TRACE OVERTEMPERATURE
<3> HEAT TRACE TEMP MON INHIBIT
<4> HEAT TRACE TEMP MON OFFSCAN
<5> HEAT TRACE TEMP MON IBAD
<6>
<7>

[84]
<0> 241 AREA RADIATION NORM
<1> 241 AREA RADIATION HIGH
<2> 241 AREA RADIATION HIGH
<3> 241 AREA RAD MONIT INHA
<4> 241 AREA RAD MONIT OSCN
<5> 241 AREA RAD MONIT IBAD
<6>
<7>

[85]
<0> 271 BUILDING RADIATION NORM
<1> 271 BUILDING RADIATION HIGH
<2> 271 BUILDING RADIATION HIGH
<3> 271 BLDG RADIATION INHIBIT
<4> 271 BLDG RADIATION OFF SCAN
<5> 271 BLDG RADIATION BAD INPT
<6>
<7>

[86]
<0> CENTRAL PUMP PIT NORM
<1> CENTRAL PUMP PIT LEAK
<2> CENTRAL PUMP PIT LEAK
<3> CENTRAL PUMP PIT INHA
<4> CENTRAL PUMP PIT OSCN
<5> CENTRAL PUMP PIT IBAD
<6>
<7>

[87]
<0>
<1> PUMP PIT LEAK
<2> PUMP PIT LEAK
<3> LEAK DET INHA
<4> LEAK DET OSCN
<5> LEAK DET IBAD
<6>
<7>

[88]
<0> 271 CAM FLOW NORM
<1> 271 CAM FLOW LOW
<2> 271 CAM FLOW LOW
<3> 271 CAM FLOW INHA
<4> 271 CAM FLOW OSCN
<5> 271 CAM FLOW IBAD
<6>
<7>

[89]
<0> ANNULUS PUMP PIT NORM
<1> ANNULUS PUMP PIT LEAK
<2> ANNULUS PUMP PIT LEAK
<3> ANNULUS PUMP PIT INHA
<4> ANNULUS PUMP PIT OSCN
<5> ANNULUS PUMP PIT IBAD
<6>
<7>

[90]
<0> ANNUNCIATOR POWER NORM
<1> ANNUNCIATOR POWER LOSS
<2> ANNUNCIATOR POWER LOSS
<3> ANNUNCIATOR POWER INHA
<4> ANNUNCIATOR POWER OSCN
<5> ANNUNCIATOR POWER IBAD
<6>
<7>

[91]
<0> BUILDING RADIATION MONITOR NORM
<1> BUILDING RADIATION MONITOR FAIL
<2> BUILDING RADIATION MONITOR FAIL
<3> BUILDING RADIATION MONITOR INHA
<4> BUILDING RADIATION MONITOR OSCN
<5> BUILDING RADIATION MONITOR IBAD
<6>
<7>

WHC-SD-534-CSWL-001 REV 3

[92]
 <0> 241 AREA RADIATION MONITOR NORM
 <1> 241 AREA RADIATION MONITOR FAIL
 <2> 241 AREA RADIATION MONITOR FAIL
 <3> 241 AREA RADIATION MONITOR INHA
 <4> 241 AREA RADIATION MONITOR OSCN
 <5> 241 AREA RADIATION MONITOR IBAD
 <6>
 <7>

[93]
 <0>
 <1> WF HIGH
 <2> WF HIGH
 <3> WF HIGH
 <4> WF HIGH
 <5> WF HIGH
 <6>
 <7>

[94]
 <0>
 <1> DE-ENTRAINMENT
 <2> DE-ENTRAINMENT
 <3> LOWER PAD
 <4> LOWER PAD
 <5> HIGH DELTA P
 <6> HIGH DELTA P
 <7> HIGH DELTA P
 <8>

[95]
 <0> M
 <1> A
 <2>
 <3>
 <4>
 <5>
 <6>
 <7>
 <8>

[96]
 <0>
 <1> REBOILER 10 LB STEAM PRESSURE LOW
 <2> REBOILER 10 LB STEAM PRESSURE LOW
 <3>
 <4>
 <5>
 <6>
 <7>

[97]
 <0> D
 <1> C
 <2>
 <3>
 <4>
 <5>
 <6>
 <7>

[98]
 <0> DESUPERHEATER OUTLET TEMPERATURE HIGH
 <1> DESUPERHEATER OUTLET TEMPERATURE HIGH
 <2> DESUPERHEATER OUTLET TEMPERATURE HIGH
 <3>
 <4>
 <5>
 <6>
 <7>

[99]
 <0> OPEN
 <1> CLOSED
 <2> CLOSED
 <3> INHA
 <4> OSCN
 <5> IBAD
 <6>
 <7>

[100]
 <0>
 <1> SHUTDOWN
 <2>
 <3>
 <4>
 <5>
 <6>
 <7>

[101]
 <0>
 <1> FAILED
 <2> FAILED
 <3> INHA
 <4> OSCN
 <5> IBAD
 <6>
 <7>

[102]
 <0>
 <1> RADN HI
 <2> RADN HI
 <3> INHA
 <4> OSCN
 <5> IBAD
 <6>
 <7>

[103]
 <0>
 <1> ANN EXH RAD HIGH
 <2> ANN EXH RAD HIGH
 <3> ANN EXH RAD INHA
 <4> ANN EXH RAD OSCN
 <5> ANN EXH RAD IBAD
 <6>
 <7>

WHC-SD-534-CSWL-001 REV 3

[104]
 <0>
 <1> {ANN EXH RAD FAIL
 <2> ANN EXH RAD FAIL
 <3> ANN EXH RAD INHA
 <4> ANN EXH RAD OSCN
 <5> ANN EXH RAD IBAD
 <6>
 <7>

[105]
 <0>
 <1> {LEAK PIT RAD MON FAIL
 <2> LEAK PIT RAD MON FAIL
 <3> LEAK PIT RAD MON INHA
 <4> LEAK PIT RAD MON OSCN
 <5> LEAK PIT RAD MON IBAD
 <6>
 <7>

[106]
 <0> LEAK DETECT PUMP PIT LEAK
 <1> {LEAK DETECT PUMP PIT LEAK
 <2> LEAK DETECT PUMP PIT LEAK
 <3> LEAK DET PMP PIT LEAK INHA
 <4> LEAK DET PMP PIT LEAK OSCN
 <5> LEAK DET PMP PIT LEAK IBAD
 <6>
 <7>

[107]
 <0> BYPASS OFF
 <1> {BYPASS ON
 <2> BYPASS ON
 <3> BYPASS ON
 <4> BYPASS ON
 <5> BYPASS
 <6>
 <7>

[108]
 <0>
 <1> {ALARM
 <2> ACKWD
 <3>
 <4>
 <5>
 <6>
 <7>

[109]
 <0>
 <1> {ALARM (PG)
 <2> ALARM
 <3>
 <4>
 <5>
 <6>
 <7>

[110]
 <0>
 <1> {ALARM (LT)
 <2> ALARM
 <3>
 <4>
 <5>
 <6>
 <7>

[111]
 <0>
 <1> {ALARM (RT)
 <2> ALARM
 <3>
 <4>
 <5>
 <6>
 <7>

[112]
 <0> EXCESS VACUUM ALM NORM
 <1> {TANKS EXCESS VACUUM ALM
 <2> TANKS EXCESS VACUUM ALM
 <3> TANKS EXCS VAC ALM INHA
 <4> TANKS EXCS VAC ALM OSCN
 <5> TANKS EXCS VAC ALM IBAD
 <6>
 <7>

[113]
 <0> VACUUM LOSS ALM NORMAL
 <1> {TANKS VACUUM LOSS ALARM
 <2> TANKS VACUUM LOSS ALARM
 <3> TANKS VAC LOSS ALM INHA
 <4> TANKS VACU LOSS ALM OSCN
 <5> TANKS VACU LOSS ALM IBAD
 <6>
 <7>

[114]
 <0>
 <1> {TL-021 LINE LEAK
 <2> TL-021 LINE LEAK
 <3> TL-021 LEAK INHA
 <4> TL-021 LEAK OSCN
 <5> TL-021 LEAK IBAD
 <6>
 <7>

[115]
 <0>
 <1> {AW CIRCUIT TROUBLE/LEAK
 <2> AW CIRCUIT TROUBLE/LEAK
 <3> AW CIRCUIT TROUBL INHA
 <4> AW CIRCUIT TROUBL OSCN
 <5> AW CIRCUIT TROUBL IBAD
 <6>
 <7>

WHC-SD-534-CSWL-001 REV 3

[116]

<0>
 <1> DRAIN ENCASEMENT LEAK
 <2> DRAIN ENCASEMENT LEAK
 <3> DRAIN ENCASE LK INHA
 <4> DRAIN ENCASE LK OSCN
 <5> DRAIN ENCASE LK IBAD
 <6>
 <7>

[117]

<0>
 <1> SLURRY ENCASEMENT LEAK
 <2> SLURRY ENCASEMENT LEAK
 <3> SLURRY ENCASE LK INHA
 <4> SLURRY ENCASE LK OSCN
 <5> SLURRY ENCASE LK IBAD
 <6>
 <7>

[118]

<0>
 <1> SUPERNATE ENCASEMENT LEAK
 <2> SUPERNATE ENCASEMENT LEAK
 <3> SUPERNATE ENCASE LK INHA
 <4> SUPERNATE ENCASE LK OSCN
 <5> SUPERNATE ENCASE LK IBAD
 <6>
 <7>

[119]

<0>
 <1> SN-269 LINE LEAK
 <2> SN-269 LINE LEAK
 <3> SN-269 LINE INHA
 <4> SN-269 LINE OSCN
 <5> SN-269 LINE IBAD
 <6>
 <7>

[120]

<0>
 <1> SN-270 LINE LEAK
 <2> SN-270 LINE LEAK
 <3> SN-270 LINE INHA
 <4> SN-270 LINE OSCN
 <5> SN-270 LINE IBAD
 <6>
 <7>

[121]

<0>
 <1> SN-FLUSH LINE PRESSURE HI
 <2> SN-FLUSH LINE PRESSURE HI
 <3> SN-FLUSH LINE PRESS INHA
 <4> SN-FLUSH LINE PRESS OSCN
 <5> SN-FLUSH LINE PRESS IBAD
 <6>
 <7>

[122]

<0>
 <1> FEEDLINE PRESSURE HI
 <2> FEEDLINE PRESSURE HI
 <3> FEEDLINE PRESS INHA
 <4> FEEDLINE PRESS OSCN
 <5> FEEDLINE PRESS IBAD
 <6>
 <7>

[123]

<0> CLEANOUT BOX NORM
 <1> CLEANOUT BOX LEAK
 <2> CLEANOUT BOX LEAK
 <3> CLEANOUT BOX INHA
 <4> CLEANOUT BOX OSCN
 <5> CLEANOUT BOX IBAD
 <6>
 <7>

[124]

<0> RADIATION NORMAL
 <1> RADIATION HI/FAIL
 <2> RADIATION HI/FAIL
 <3> RAD HI/FAIL INHA
 <4> RAD HI/FAIL OSCN
 <5> RAD HI/FAIL IBAD
 <6>
 <7>

[125]

<0> CLEANOUT BOX NORM
 <1> CLEANOUT BOX LEAK
 <2> CLEANOUT BOX LEAK
 <3> CLEANOUT BOX INHA
 <4> CLEANOUT BOX OSCN
 <5> CLEANOUT BOX IBAD
 <6>
 <7>

[126]

<0>
 <1> MCC OFF
 <2> MCC OFF
 <3> MCC INHA
 <4> MCC OSCN
 <5> MCC IBAD
 <6>
 <7>

[127]

<0> ANN EXH RAD HI/FAIL NORM
 <1> ANN EXH RAD HI/FAIL FAIL
 <2> ANN EXH RAD HI/FAIL FAIL
 <3> ANN EXH RAD HI/FAIL INHA
 <4> ANN EXH RAD HI/FAIL OSCN
 <5> ANN EXH RAD HI/FAIL IBAD
 <6>
 <7>

WHC-SD-534-CSWL-001 REV 3

[128]
 <0>
 <1> TANK EXH FANS OFF
 <2> TANK EXH FANS OFF
 <3> TANK EXH FAN INHA
 <4> TANK EXH FAN OSCN
 <5> TANK EXH FAN IBAD
 <6>
 <7>

[129]
 <0>
 <1> AX-B SL-100 LINE LEAK
 <2> AX-B SL-100 LINE LEAK
 <3> AX-B SL-100 LINE INHA
 <4> AX-B SL-100 LINE OSCN
 <5> AX-B SL-100 LINE IBAD
 <6>
 <7>

[130]
 <0>
 <1> AX-B SN-200 SN-213 LINE LEAK
 <2> AX-B SN-200 SN-213 LINE LEAK
 <3> AX-B SN-200 SN-213 LEAK INHA
 <4> AX-B SN-200 SN-213 LEAK OSCN
 <5> AX-B SN-200 SN-213 LEAK IBAD
 <6>
 <7>

[131]
 <0>
 <1> AX-A SN-201 SN-214 LINE LEAK
 <2> AX-A SN-201 SN-214 LINE LEAK
 <3> AX-A SN-201 SN-214 LEAK INHA
 <4> AX-A SN-201 SN-214 LEAK OSCN
 <5> AX-A SN-201 SN-214 LEAK IBAD
 <6>
 <7>

[132]
 <0>
 <1> AX-A SL-101 LEAK
 <2> AX-A SL-101 LEAK
 <3> AX-A SL-101 INHA
 <4> AX-A SL-101 OSCN
 <5> AX-A SL-101 IBAD
 <6>
 <7>

[133]
 <0>
 <1> A-B SL-113 LEAK
 <2> A-B SL-113 LEAK
 <3> A-B SL-113 INHA
 <4> A-B SL-113 OSCN
 <5> A-B SL-113 IBAD
 <6>
 <7>

[134]
 <0>
 <1> A-A SL-114 LEAK
 <2> A-A SL-114 LEAK
 <3> A-A SL-114 INHA
 <4> A-A SL-114 OSCN
 <5> A-A SL-114 IBAD
 <6>
 <7>

[135]
 <0>
 <1> HEPA DP FAN FAIL EXHAUST RADIATION
 <2> HEPA DP FAN FAIL EXHAUST RADIATION
 <3> HEPA DP FAN FAIL EXHAUST RAD INHA
 <4> HEPA DP FAN FAIL EXHAUST RAD OSCN
 <5> HEPA DP FAN FAIL EXHAUST RAD IBAD
 <6>
 <7>

[136]
 <0>
 <1> HEPA DELTA PRESS HIGH
 <2> HEPA DELTA PRESS HIGH
 <3> HEPA DELTA PRESS INHA
 <4> HEPA DELTA PRESS OSCN
 <5> HEPA DELTA PRESS IBAD
 <6>
 <7>

[137]
 <0>
 <1> STACK MONITOR SYSTEM FAILURE
 <2> STACK MONITOR SYSTEM FAILURE
 <3> STACK MONITOR SYS FAIL INHA
 <4> STACK MONITOR SYS FAIL OSCN
 <5> STACK MONITOR SYS FAIL IBAD
 <6>
 <7>

[138]
 <0>
 <1> EXHAUST STACK RADIATION HIGH
 <2> EXHAUST STACK RADIATION HIGH
 <3> EXHAUST STACK RADIATION INHA
 <4> EXHAUST STACK RADIATION OSCN
 <5> EXHAUST STACK RADIATION IBAD
 <6>
 <7>

[139]
 <0>
 <1> SERVICE PIT
 <2> SERVICE PIT
 <3> SERVICE PIT
 <4> SERVICE PIT
 <5> SERVICE PIT
 <6>
 <7>

WHC-SD-534-CSWL-001 REV 3

[140]
 <0>
 <1> | FLUSH PIT
 <2> FLUSH PIT
 <3> FLUSH PIT
 <4> FLUSH PIT
 <5> FLUSH PIT
 <6>
 <7>

[141]
 <0>
 <1> | LOLO
 <2> LOLO
 <3> INHA
 <4> OSCN
 <5> IBAD
 <6>
 <7>

[142]
 <0>
 <1> | LO
 <2> LO
 <3> INHA
 <4> OSCN
 <5> IBAD
 <6>
 <7>

[143]
 <0>
 <1> | HI
 <2> HI
 <3> INHA
 <4> OSCN
 <5> IBAD
 <6>
 <7>

[144]
 <0>
 <1> | HIHI
 <2> HIHI
 <3> INHA
 <4> OSCN
 <5> IBAD
 <6>
 <7>

[145]
 <0>
 <1> | STACK FLOW HIGH
 <2> STACK FLOW HIGH
 <3> STACK FLOW INHA
 <4> STACK FLOW OSCN
 <5> STACK FLOW IBAD
 <6>
 <7>

[146]
 <0>
 <1> | PW-481 WASTE LINE LEAK
 <2> PW-481 WASTE LINE LEAK
 <3> PW-481 WASTE LINE INHA
 <4> PW-481 WASTE LINE OSCN
 <5> PW-481 WASTE LINE IBAD
 <6>
 <7>

[147]
 <0>
 <1> | LINE LEAK
 <2> LINE LEAK
 <3> LEAK INHA
 <4> LEAK OSCN
 <5> LEAK IBAD
 <6>
 <7>

[148]
 <0>
 <1> | BASIN LEVEL LOW
 <2> BASIN LEVEL LOW
 <3> BASIN LEVEL INHA
 <4> BASIN LEVEL OSCN
 <5> BASIN LEVEL IBAD
 <6>
 <7>

[149]
 <0>
 <1> | BASIN LEVEL HIGH
 <2> BASIN LEVEL HIGH
 <3> BASIN LEVEL INHA
 <4> BASIN LEVEL OSCN
 <5> BASIN LEVEL IBAD
 <6>
 <7>

[150]
 <0>
 <1> | PRESS LOW
 <2> PRESS LOW
 <3> PRSS INHA
 <4> PRSS OSCN
 <5> PRSS IBAD
 <6>
 <7>

[151]
 <0>
 <1> 24 VDC POWER SUPPLY NORM
 <2> 24 VDC POWER SUPPLY FAIL
 <3> 24 VDC POWER SUPPLY FAIL
 <4> 24 VDC POWER SUPPLY INHA
 <5> 24 VDC POWER SUPPLY DIS
 <6> 24 VDC POWER SUPPLY IBAD
 <7>
 <8>

WHC-SD-534-CSWL-001 REV 3

[152]
 <0> DIESEL GENERATOR NORMAL
 <1> DIESEL GENERATOR TROUBLE
 <2> DIESEL GENERATOR TROUBLE
 <3> DIESEL GENERATOR INHA
 <4> DIESEL GENERATOR DIS
 <5> DIESEL GENERATOR IBAD
 <6>
 <7>

[153]
 <0> DIESEL GENERATOR OFF
 <1> DIESEL GENERATOR RUNNING
 <2> DIESEL GENERATOR RUNNING
 <3> DIESEL GEN OFF/ON INHA
 <4> DIESEL GEN OFF/ON DIS
 <5> DIESEL GEN OFF/ON IBAD
 <6>
 <7>

[154]
 <0> UPS SUMMARY NORML
 <1> UPS SUMMARY ALARM
 <2> UPS SUMMARY ALARM
 <3> UPS SUMMARY INHA
 <4> UPS SUMMARY DIS
 <5> UPS SUMMARY IBAD
 <6> UPS SUMMARY ALARM
 <7>

[155]
 <0>
 <1> CF-ON
 <2> CF-ON
 <3>
 <4>
 <5>
 <6>
 <7>

[156]
 <0> RAD MON NORM
 <1> RAD MON FAIL
 <2> RAD MON FAIL
 <3> RAD MON INHA
 <4> RAD MON OSCN
 <5> RAD MON IBAD

[157]
 <0>
 <1> RAD HI/FAIL
 <2> RAD HI/FAIL
 <3> RAD INHA
 <4> RAD OSCN
 <5> RAD IBAD

[158]
 <0>
 <1> ALPHA RADN HI
 <2> ALPHA RADN HI
 <3> ALPHA RAD INHA
 <4> ALPHA RAD OSCN
 <5> ALPHA RAD IBAD

[159]
 <0>
 <1> BETA/GAMMA HI
 <2> BETA/GAMMA HI
 <3> BETA/GAM INHA
 <4> BETA/GAM OSCN
 <5> BETA/GAM IBAD
 <6>

[160]
 <0>
 <1> SYSTEM TROUBLE
 <2> SYSTEM TROUBLE
 <3> SYS TROUB INHA
 <4> SYSTEM OSCN
 <5> SYSTEM IBAD
 <6>

[161]
 <0> RADIATION HI
 <1> RADIATION HI
 <2> RAD HI INHA
 <3> RAD HI OSCN
 <4> RAD HI IBAD

[162]
 <0>
 <1> RAD HI/FAIL
 <2> RAD HI/FAIL
 <3> RAD H/F INHA
 <4> RAD H/F OSCN
 <5> RAD H/F IBAD

[163]
 <0>
 <1> DP LOW
 <2> DP LOW
 <3>
 <4>
 <5>

[164]
 <0>
 <1> DP HI
 <2> DP HI
 <3>
 <4>
 <5>

WHC-SD-534-CSWL-001 REV 3

[165]

<0> REMOTE
<1> LOCAL
<2> LOCAL
<3>
<4>
<5>

[166]

<0> PUMP ON
<1> PUMP OFF
<2> PUMP OFF
<3>
<4>
<5>

[167]

<0>
<1> OVERFLOW TRAP LEVEL LO
<2> OVERFLOW TRAP LEVEL LO
<3> OVERFLOW TRAP LEVEL LO
<4> OVERFLOW TRAP LEVEL LO
<5> OVERFLOW TRAP LEVEL LO

[168]

<0>
<1> LOW
<2> LOW

[169]

<0>
<1> LEVEL
<2> LEVEL

[170]

<0>
<1> WATER SERVICE BLDG TEMP LO
<2> WATER SERVICE BLDG TEMP LO
<3> WATER SERVICE BLDG TEMP INHA
<4> WATER SERVICE BLDG TEMP OSCN
<5> WATER SERVICE BLDG TEMP IBAD
<6>
<7>

[171]

<0>
<1> EVAP VESSEL PRESSURE HIGH
<2> EVAP VESSEL PRESSURE HIGH
<3> EVAP VESSEL PRESSURE INHA
<4> EVAP VESSEL PRESSURE OSCN
<5> EVAP VESSEL PRESSURE IBAD
<6>
<7>

[172]

<0>
<1> LEAK
<2> LEAK
<3> LEAK
<4> LEAK
<5> LEAK

[173]

<0>
<1> ON
<2>
<3>
<4>
<5>

[174]

<0>
<1> OVERFLOW TRAP LEVEL LO
<2> OVERFLOW TRAP LEVEL LO
<3> OVERFLOW TRAP LVL INHA
<4> OVERFLOW TRAP LVL OSCN
<5> OVERFLOW TRAP LVL IBAD

[175]

<0>
<1> LINE TRAP LEVEL LO
<2> LINE TRAP LEVEL LO
<3> LINE TRAP LVL INHA
<4> LINE TRAP LVL OSCN
<5> LINE TRAP LVL IBAD

[176]

<0>
<1> TEMPERATURE HI
<2> TEMPERATURE HI
<3> TEMP HI INHA
<4> TEMP HI OSCN
<5> TEMP HI IBAD

[177]

<0>
<1> BROKEN
<2> BROKEN
<3> INHA
<4> OSCN
<5> IBAD

[178]

<0>
<1> DESUP OUTLET TEMP HIGH
<2> DESUP OUTLET TEMP HIGH
<3> DESUP OUTLET TEMP INHA
<4> DESUP OUTLET TEMP OSCN
<5> DESUP OUTLET TEMP IBAD

[179]

<0>
<1> A SN-215 LEAK
<2> A SN-215 LEAK
<3> A SN-215 INHA
<4> A SN-215 OSCN
<5> A SN-215 IBAD

[180]

<0>
<1> A SN-216 LEAK
<2> A SN-216 LEAK
<3> A SN-216 INHA
<4> A SN-216 OSCN
<5> A SN-216 IBAD

WHC-SD-534-CSWL-001 REV 3

[181]
 <0>
 <1> VALVE FAILED TO CLOSE
 <2> VALVE FAILED TO CLOSE
 <3>
 <4>
 <5>

[182]
 <0>
 <1> RUNNING
 <2> TRAPPED
 <3> CRITICAL
 <4> WAIT
 <5> WAITUNTIL
 <6> NOT RUNNING
 <7> NO STATUS
 <8>
 <9>
 <10>

[183]
 <0>
 <1> LOCAL

[184]
 <0>
 <1> TURBIN FAN ON
 <2> TURBIN FAN ON
 <3> TURBIN FN INHA
 <4> TURBIN FN OSCN
 <5> TURBIN FN IBAD

[185]
 <0>
 <1> ERROR

[186]
 <0> PERMISSIVE OFF
 <1> PERMISSIVE ON

[187]
 <0> PUMP OFF
 <1> PUMP ON

[188]
 <0> INTERLOCK
 <1> NORMAL

[189]
 <0> BYPASS OFF
 <1> BYPASS ON

[190]
 <0> NOT OPEN
 <1> OPEN

[191]
 <0> NOT CLOSED
 <1> CLOSED

[192]
 <0> LOCAL
 <1> REMOTE

[193]
 <0> OFF
 <1> ON

[194]
 <0> PRMSSV OFF
 <1> PERMISSIVE

[195]
 <0> STOP/RESET
 <1> START

[196]
 <0>
 <1> TIMEOUT
 <2>

[197]
 <0> TIMER OFF
 <1> TIMER ON

[198]
 <0> POSN-1
 <1> NOT POS1

[199]
 <0> POSN-2
 <1> NOT POS2

[200]
 <0> NOT DVRT
 <1> DIVERTED

[201]
 <0> NOT NORM
 <1> TO WEIR

[202]
 <0> NOT NORM
 <1> TO BASIN

[203]
 <0> DIVERTED
 <1> NORMAL

[204]
 <0>
 <1> SAMPLER LOW FLOW
 <2> SAMPLER LOW FLOW
 <3> SAMPLER LOW FLOW
 <4> SAMPLER LOW FLOW
 <5> SAMPLER LOW FLOW

WHC-SD-534-CSWL-001 REV 3

[205]
 <0>
 <1> |RAD SYSTEM FAIL
 <2> RAD SYSTEM FAIL
 <3> RAD SYSTEM FAIL
 <4> RAD SYSTEM FAIL
 <5> RAD SYSTEM FAIL

[206]
 <0>
 <1> |PUMP ENCODER LOSS
 <2> PUMP ENCODER LOSS
 <3> PUMP ENCODER INHA
 <4> PUMP ENCODER OSCN
 <5> PUMP ENCODER IBAD

[207]
 <0>
 <1> |VAR FREQ DRIVE FAULT
 <2> VAR FREQ DRIVE FAULT
 <3> VAR FREQ DRIVE INHA
 <4> VAR FREQ DRIVE OSCN
 <5> VAR FREQ DRIVE IBAD

[208]
 <0>
 <1> |PUMP ZERO SPEED
 <2> PUMP ZERO SPEED
 <3> PUMP SPEED INHA
 <4> PUMP SPEED OSCN
 <5> PUMP SPEED IBAD

[209]
 <0> EMERGENCY POWER TRANSFR SWITCH OFF
 <1> |EMERGENCY POWER TRANSFER SWITCH ON
 <2> EMERGENCY POWER TRANSFER SWITCH ON
 <3> EMERGENCY POWER XFR SWITCH INHA
 <4> EMERGENCY POWER XFR SWITCH DIS
 <5> EMERGENCY POWER XFR SWITCH IBAD

[210]
 <0>
 <1> |LOW
 <2> LOW
 <3> INHA
 <4> OSCN
 <5> IBAD

[211]
 <0>
 <1> |LEVEL
 <2> LEVEL
 <3>
 <4>
 <5>

[212]
 <0> SD-RESET

[213]
 <0> SHUTDOWN

[214]
 <0> INTERLOK

[215]
 <0>
 <1> |FAULT
 <2> FAULT

[216]
 <0> CF-OFF
 <1> CF-ON
 <2> SD-RESET
 <3>
 <4> SHUTDOWN
 <5> INTERLOCK
 <6> STOPPING
 <7> STARTING

[217]
 <0>
 <1>
 <2>
 <3> INHA
 <4> |DIS
 <5> IBAD

[218]
 <0>
 <1> |LO LEVEL
 <2> LO LEVEL
 <3> LO INHA
 <4> |LO DIS
 <5> LO IBAD

[219]
 <0>
 <1> |SHUTDOWN
 <2> SHUTDOWN
 <3> SHUTDOWN
 <4> SHUTDOWN
 <5> SHUTDOWN

[220]
 <0>
 <1> |INLET PRESS HI
 <2> INLET PRESS HI
 <3> INLET PRESS HI
 <4> INLET PRESS HI
 <5> INLET PRESS HI

[221]
 <0> TIMER OFF
 <1> |TIMER ON

[222]
 <0> MANUAL
 <1> AUTO

WHC-SD-534-CSWL-001 REV 3

[223]
 <0> NOT SELECTED
 <1> SELECTED
 <2> OFFLINE
 <3>
 <4>
 <5>

[224]
 <0> VESSEL VENT RADIATION SYSTEM NORMAL
 <1> VESSEL VENT RADIATION SYSTEM FAILURE
 <2> VESSEL VENT RADIATION SYSTEM FAILURE
 <3> VESSEL VENT RADIATION SYSTEM INHIBIT
 <4> VESSEL VENT RADIATION SYSTEM OFF SCN
 <5> VESSEL VENT RADIATION SYSTEM BAD INP

[225]
 <0> CLOSED
 <1> OPEN

[226]
 <0>
 <1> 242-A TO L.E.R.F. LEAK
 <2> 242-A TO L.E.R.F. LEAK
 <3> 242-A TO L.E.R.F. LEAK
 <4> 242-A TO L.E.R.F. LEAK
 <5> 242-A TO L.E.R.F. LEAK

[227]
 <0> OFF
 <1> ON
 <2> ON
 <3>
 <4>
 <5>

[228]
 <0>
 <1> *
 <2> *
 <3>
 <4>
 <5>

[229]
 <0> OFF
 <1> ON
 <2> ON
 <3> INHA
 <4> DIS
 <5> IBAD

[230]
 <0>
 <1> CF-OFF
 <2> CF-OFF

[231]
 <0> UPS AC INPUT NORM
 <1> UPS AC INPUT FAIL
 <2> UPS AC INPUT FAIL
 <3> UPS AC INPUT INHA
 <4> UPS AC INPUT DIS
 <5> UPS AC INPUT IBAD
 <6>

[232]
 <0> UPS BATTERY NORM
 <1> UPS BATTERY LOW
 <2> UPS BATTERY LOW
 <3> UPS BATTERY INHA
 <4> UPS BATTERY DIS
 <5> UPS BATTERY IBAD
 <6>

[233]
 <0> UPS BYP SWITCH NORMAL
 <1> UPS BYP SWITCH BYPASS
 <2> UPS BYP SWITCH BYPASS
 <3> UPS BYP SWITCH INHA
 <4> UPS BYP SWITCH DIS
 <5> UPS BYP SWITCH IBAD
 <6>

[234]
 <0> ON
 <1> OFF
 <2> OFF
 <3> INHA
 <4> DIS
 <5> IBAD

[235]
 <0>
 <1> AREA RAD MON FAIL
 <2> AREA RAD MON FAIL
 <3> AREA RAD MON INHA
 <4> AREA RAD MON DIS
 <5> AREA RAD MON IBAD

[236]
 <0>
 <1> AIR SAMP MON FAIL
 <2> AIR SAMP MON FAIL
 <3> AIR SAMP MON INHA
 <4> AIR SAMP MON DIS
 <5> AIR SAMP MON IBAD

[237]
 <0>
 <1> AS RADIATION HI
 <2> AS RADIATION HI
 <3> AS RADN HI INHA
 <4> AS RADN HI DIS
 <5> AS RADN HI IBAD

WHC-SD-534-CSWL-001 REV 3

[238]
 <0>
 <1> AREA RADIATION HI
 <2> AREA RADIATION HI
 <3> AREA RADN HI INHA
 <4> AREA RADN HI DIS
 <5> AREA RADN HI IBAD

[239]
 <0> PRESSURE NORM
 <1> PRESSURE HIGH
 <2> PRESSURE HIGH
 <3> PRESS HI INHA
 <4> PRESS HI DIS
 <5> PRESS HI IBAD

[240]
 <0>
 <1> LERF MODEM COMM FAIL
 <2> LERF MODEM COMM FAIL
 <3> LERF MODEM FAIL INHA
 <4> LERF MODEM FAIL DIS
 <5> LERF MODEM FAIL IBAD

[241]
 <0> ON
 <1> OFF
 <2> OFF
 <3>
 <4>
 <5>

[242]
 <0>
 <1> RW ON
 <2> RW ON
 <3> INHA
 <4> DIS
 <5> IBAD

[243]
 <0>
 <1> SHUTDOWN
 <2>
 <3> SHUTDOWN
 <4> SHUTDOWN
 <5> SHUTDOWN

[244]
 <0>
 <1> ANNULUS EXH RADIATION HI
 <2> ANNULUS EXH RADIATION HI
 <3> ANNULUS EXH RADN INHA
 <4> ANNULUS EXH RADN DIS
 <5> ANNULUS EXH RADN IBAD

[245]
 <0>
 <1> LEAK DETECT PUMP PIT LEAK
 <2> LEAK DETECT PUMP PIT LEAK
 <3> LEAK DETECT PUMP PIT INHA
 <4> LEAK DETECT PUMP PIT DIS
 <5> LEAK DETECT PUMP PIT IBAD

[246]
 <0>
 <1> WATER SERVICE BLDG TEMP LO
 <2> WATER SERVICE BLDG TEMP LO
 <3> WATER SERVICE BLDG TEMP INHA
 <4> WATER SERVICE BLDG TEMP OSCN
 <5> WATER SERVICE BLDG TEMP IBAD
 <6>
 <7>

WHC-SD-534-CSWL-001 REV 3

THIS PAGE LEFT BLANK INTENTIONALLY

WHC-SD-534-CSWL-001 REV 3

APPENDIX H - REBUILD PROCEDURE

INTRODUCTION

The 242-A control room is run by a Distributive Controls System (DCS) designed by Texas Instruments (TI) now known as GSE Process Solutions Inc. (GSE). The system runs on two redundant VAX 4000-105A computers located in the control room. The operating system on the computers is VMS version 5.5-2AH. The TI software is called D3 and is version 6.3-2. The two computers are known as CDCM and DCMO. The disk on both machines is DKA300: and is organized as follows. The D3 source code is located in a directory called D3SOURCE: and the site specific files are located in the directory called D3DATA:. To properly rebuild the system a few VMS commands are good to know. First to change you place iOn the directory structure the command SET DEFAULT is used or SET DEF for short. To change to the directory D3DATA: you would type SET DEF D3DATA:[000000]. Since D3DATA is a logical pointing to the a specific place on the directory structure it will appear that now the drive is D3DATA:. If you want to see where you are located in the directory structure you would type SHOW DEF. You will see D3DATA:[000000]. The six zero imply that you are at the top of that directory. Within the D3DATA directory you will see many other directory by typing DIRECTORY or DIR for short. The directories will be 6 digit numbers. The information for building the Continuous Control Database is located in directory [100001] or [100,1]. To change to that directory you would type SET DEF [100001]. A SHOW DEF command you display D3DATA:[100001] for your location on the disk. Another useful way to use the DIR command is to search for a file on the disk when you have forgotten where it is located. If you go to the top of the directory structure by typing SET DEF DKA300:[000000] you can then look for a file anywhere on the disk by typing DIR [*...]*file-name*. If you would like to search all .COM files to see which file contains a phrase you are looking for you can use the SEARCH or SEA command. From the top of the directory structure you could type SEA [*...]*.COM PCMOA. This would search in every .COM file on the disk and report back every file that contains the word PCMOA. Other commands that may be useful are COPY, MOVE, RENAME, and DELETE. All these file work as they do for IBM DOS.

Command files are used to rebuild the system. They are files that end in .COM and can be found through out the directory structure. These file are described through out this document. For the most part the command files for rebuilding the system are in D3DATA:[001001].

SYSTEM REBUILD PROCEDURE

1. REBUILD ALL OR PORTIONS OF CONTINUOUS DATABASE:

a. REBUILD A PORTION - for example redefine an EPN:

1. Write command file to report out existing EPN you wish to redefine. The command file content should be as follows:

```
KILL          (prevents new version of UDB)
NOCHECK       (avoids rebuild of SDB, SEQ, CGR)
REP <EPN>     (do this for each EPN)
EXIT          (exits MDL utility)
```

2. Run the command file with MODEL to report out the EPNs to be modified. The command format to run MODEL should be as follows:

```
MDL PCM__ -I <infile name.CMD> -O <outfile name.OUT>
```

where PCM__ is the name of the PCM database.

3. Modify the output (.OUT) file. I suggest that the modified file be named with ".INP" as the file name extension. This will differentiate the old from the new and will result in concise records of both configurations. After modifying, make sure of the following:

- a. The .INP file uses the MODEL commands KILL, NOCHECK, and EXIT.
- b. All REPs replaced with DELs.
- c. Analogs are redefined before MDVs and DEVs, in case the analogs "forward reference" the DEVs.
- d. DEVs are deleted first followed by MDVs, and that the MDVs are redefined before the DEVs.
- e. Some MDVs are associated with more then one DEV. So make sure all DEVs associated with a common MDV are redefined.

4. Run local MODEL with KILL on for error checking and debugging. With KILL on, a new database is not created. If MDL is run without KILL on and errors are generated, then the new database will be corrupted. I suggest writing the output file of this session to a file name having the .ERR extension, denoting error output file. For example:

```
MDL PCM__ -I <infile name.INP> -O <outfile name.ERR>
```

Pay attention to the CRT as MODEL runs as not all errors get written to the error output file, although most do. View the error file for errors. If no errors are indicated at the 'end of file', then proceed as long as the CRT did not display any errors.

5. EDT the .INP file to remove the KILL command.
6. Run Remote MODEL (CAUTION !!!) with KILL off again using the .ERR output file specification:

```
MDL -R PCM__ -I <infile name.INP> -O <outfile name.ERR>
```

7. View the error file. If all looks well and this is a permanent change, modify the source input file for the modified database. The source input file for PCMOA PCMOB are in D3DATA:[100001]. They have .INP file extensions. All pertinent PCMOA text files are also in D3DATA:[100001] Otherwise just copy an EPN to create a new one.
8. With NOCHECK on, the sequence database, sequence programs, graphics, other displays, and c-code do not need to be rebuilt. But the display database may need to be reloaded to the DCMs (use D3DATA:[001001]DBL.COM, or a select part of it).

b. REBUILD ALL - that is, rebuild the system.

1. Use the latest PCM .INP input files in D3DATA:[100001]. These use the CREATE model command. Do not use remote model. Do not error check with KILL on - MODEL ignores KILL when CREATE is used. If at first you don't succeed, edit the input file and try again. Run local model, which only creates a new disk version:

```
MDL PCM__ -I <infile name.INP> -O <outfile name.ERR>
```

You can go to D3DATA:[001001] and use command files if creating a new database. The command @BLDALLPCMS will build both PCMOA and PCMI1A databases using PCMOA.INP and PCMI1A.INP files. It will also create the .ERR files and locate them in D3DATA:[100001]. Command file @BLDPCMOA and @BLDPCMI1A will build just the respective databases. These command files remove all display databases from the CDCM only. Then runs MDL with PCMOA and PCMI1A as input and creates the .ERR files.

2. After model is done, load the display database to the Host only! This is necessary to rebuild SABL, graphics, and other displays on the Host. Do not load to DCMO yet to minimize impact to the control room. Use the following DBL commands. From now on CDCM OCM displays may be out to lunch, but the DCMO will be unaffected. Or use command file @LOADISDBCDCM located in D3DATA:[001001].

```
DBL -H -RD PCMOA
DBL -H -D PCMOA
DBL -H -RD PCMI1A
DBL -H -D PCMI1A
```

3. Rebuild sequence database:

- a. In D3DATA:[300001], if just rebuilding PCMOA OR PCMI1A copy (NOT rename) SDBMASTER.XRF and SDBSHORT.XRF to the same file name but with .OLD extension. When SDB utility runs it overwrites the .XRF files (the old XRF files will be lost).

- b. Rebuild the Sequence databases in D3DATA:[001001]:

```
For PCMOA: SDB POASDB (runs POASDB.COM)
For PCMI1A: SDB PIASDB (runs PIASDB.COM)
```

[Note - to build all sdb files use D3DATA:[001001]BLDRMS. To build just the sdb you may use D3DATA:[001001]BLDSDB. For all PCMO, you may use D3DATA:[001001]POASDB. For all PCMI1, you may use D3DATA:[001001]PIASDB. There also is a D3DATA:[001001]BLDSYS file. Look at these and use what meets your needs at the time.]

These command files will copy the SDBMASTER.XRF and SDBSHORT.XRF to *.OLD. These command files use input files located in D3DATA:[300001] called 242A.CMD, ALLO.CMD, ALL1.CMD, BANKK.CMD, EMTY.CMD, FARM.CMD, OVRVIEW.CMD, PCMOA.CMD, PCMI1A.CMD, PROESS.CMD, SHORTO.CMD, TNKFRM.CMD, TOTALO.CMD, and TATLA1.CMD as necessary.

4. Rebuild sequence programs in D3DATA:[001001]:
  - a. For PCMOA: @SICPCMOAPROGS
  - b. For PCM1A: @SICPCM1APROGS
  - c. For BOTH: @SICALLPROGS
5. Rebuild graphics in D3DATA:[001001]:
  - a. For PCMOA: @COMPILE_PCMOA
  - b. For PCM1A: @COMPILE_PCM1A
  - c. For BOTH: @COMPILE_GRAPHICS
6. Rebuild Group Displays (POV, EPN, TRN, HIS) in D3DATA:[001001]:
  - a. To build: @BLD_GROUPS

This command file first stops the current trends and stops the historical trends.
7. Now you are ready to load the PCMs. Crash the PCMs by pressing the RESET switch on the PCMs. Reset and load only one redundant PCM pair at a time (PCMO OR PCM1).

D3DATA:[001001]LOADPCMOA, or
D3DATA:[001001]LOADPCM1A

Answer the prompts and then everything is automatically loaded to the PCMs, including sequence programs, and runs them. Don't try to do anything else. Just watch the screen to make sure everything loads and runs properly. Repeat for the other redundant pcm pair, if necessary. The two command files above start command procedure PCMLD.COM located in D3SOURCE:[001003]. This command procedure runs four different command procedures located in D3SOURCE:[001001] which load and run all the sequence programs. These command procedures are SEQLDOA.COM, SEQLDOB.COM, SEQLD1A.COM, and SEQLD1B.COM.

8. At this point, the PCMs are loaded and the Host has the new display database loaded to it. Load the files to the DCMO by using command file D3DATA:[001001]LOADDCM. But nothing else is loaded to the Host. So finish loading the Host by re-booting it as follows:
  - a. Enter D3DOWN and answer the prompts.
  - b. Enter SHUTDOWN to stop the system.
  - c. Press the halt button on the back of the CDCM. Turn the power switch off on the Q buss for the CDCM then turn the power off on the CDCM.
  - d. Turn the power switch on the Q buss first then the power switch on the back of the CDCM.
  - e. When the power comes on the CDCM will reboot. When the prompt >>> comes on the CDCM alarm typer enter a "b" and hit enter.
  - f. When the system is up you can log back on a CDCM terminal.
  - g. After logging on type D3UP to start the D3 software.
9. Now the CDCM displays, for the most part, should be functional. There may be a few straggler displays that need to be re-compiled, but don't worry about these at the moment. Also, the DCMO displays are now out to lunch. Notice how the operator for the most part has had at least one console with completely functional displays.
10. Reboot the DCMO as follows:
  - a. Enter D3DOWN and answer the prompts.
  - b. Enter SHUTDOWN to stop the system.
  - c. Press the halt button on the back of the DCMO. Turn the power switch off on the Q buss for the DCMO then turn the power off on the DCMO.
  - d. Turn the power switch on the Q buss first then the power switch on the back of the DCMO.
  - e. When the power comes on the CDCM will reboot. When the prompt >>> comes on the DCMO alarm typer enter a "b" and hit enter.
  - f. When the system is up you can log back on a DCMO terminal.
  - g. After logging on type D3UP to start the D3 software.

Now DCMO is back to normal. All that is left is some miscellaneous rebuilds and cleanup.

11. Rebuild C-code, even if IPNs have not changed. The source code for the C-code is located in D3SOURCE:[261010] or [MICRO.200010]. The *.SCR files will place the executable files in the appropriate directories such as D3SOURCE:[003055], [004055], and [005055]. The LTK files loaded to the micros are in D3SOURCE:[001055]. To build an individual C-program, set default to D3SOURCE:[261010] or [MICRO.200010] and type the following:

BLD FILE.SCR -L -ALL -MOV -LIS

D3SOURCE:[261010]:

WATCH, FQIRESET, INHA, BALANCE, TFBALANCE, HISTDAT, UKF14, UKF34, UKF43, UKF44, etc)

[MICRO.200010]:

DOGOA, DOGOB, DOG1A, and DOG1B.

To build all the C-code type the use the following command file located in D3DATA:[001001]:BLDALLC

12. If IPNs for DEVs or MDVs have changed, then make sure the key task (D3SOURCE:[160010]KEY.C, KEY.TSK) are correct for:

AUDALARMO (pcm0a dev ipn 171)

HS-AB1ACK (pcm0a dev ipn 208)

HS-AB2ACK (pcmla dev ipn 15)

13. Reload C-code to PCMs (MICRO:[377001]DOGO, DOG1) if necessary. Use command file located in D3DATA:[001001]DOGSWAP.
14. Reinstall (reboot if you like) C-code to DCMs if necessary.
15. Rebuild the Report Writer reports on each DCM: (NOTE: report writer is no longer used)

DCM		DCM0
-----		-----
ROUTINEG1	ROUTINED1	ROUTINES1
ROUTINEG2	ROUTINED2	ROUTINES2
ROUTINEG3	ROUTINED3	ROUTINES3
ROUTINEG4	ROUTINED4	ROUTINES4
TOTALIZRS		

This is done by recompiling these using RGE in [244,2].

16. Purge the old files away as required:

```
D3DATA:[100001]*.UDB, *.REF
D3DATA:[300001]PCM*SDB.LST
D3DATA:[300001]*.SEQ, *.LST, *.LTQ
D3DATA:[100010]GRA*.BIN, *.POV, *.EPN, *.TRN, *.HIS, *.COL
D3DATA:[100010]GRA*.CGR
D3SOURCE:[261010]name.C
[MICRO.200010]*.C
```

To purge all these directories above use command procedure in
D3DATA:[001001]PGALL.

17. The String Editor can now only handle an input file of 50 string packets. Since there are about 274 string packets on this system, the input file had to be split into five separate files located in D3DATA:[100010]. They are STRING1.TXT, STRING2.TXT, STRING3.TXT, STRING4.TXT, and STRING5.TXT. Command procedure STRING.BUILD.COM is located in D3DATA:[100010] and will build one string file called STRING.BIN. After making any changes to the string any graphics that use the string changed will have to be recompiled and copied to the DCM0.
18. If all input files have been properly changed, the way to build all inputs and reload everything is as follows:

```
SET DEF D3DATA:[001001]
@BLDALLPCMS 3:30
@LOADDISDBCDCM :10
@BLDRMS 6:15
@SICALLPROGS 7:30
@COMPILE GRAPHICS 8:18
@BLD GROUPS 2:30
CRASH BOTH PCMS AND TURN THEM BACK ON. 0:20
(Wait until the pcms are in load request) 1:30
@LOADPCMOA 2:30
@LOADPCMIA 2:30
@DCMLoad 3:30
DO A D3DOWN AND D3UP ON BOTH COMPUTERS 5:00
39:273
```

APPENDIX I - SKID KEY CONFIGURATION

Listing of D3DATA:[100010]CDCMSK0.CMD on the CDCM
(Other SKID command files are identical)

WHC-SD-534-CSWL-001 REV 3

KEY: 0
 LABEL: AN:271
 VERIFICATION: N
 ECHO: AN-271
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 274
 LED MODE: NONE
 FUNCTION:
 KEY: 1
 LABEL: AN:VALVE:PIT
 VERIFICATION: N
 ECHO: AN VALVE PIT
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 272
 LED MODE: NONE
 FUNCTION:
 KEY: 2
 LABEL: AN:HVAC
 VERIFICATION: N
 ECHO: AN HVAC
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 273
 LED MODE: NONE
 FUNCTION:
 KEY: 3
 LABEL: AN:FARM
 VERIFICATION: N
 ECHO: AN FARM
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 270
 LED MODE: NONE
 FUNCTION:
 KEY: 4
 LABEL: AP:271
 VERIFICATION: N
 ECHO: AP-271
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 295
 LED MODE: NONE
 FUNCTION:
 KEY: 5
 LABEL: AP:VALVE:PIT
 VERIFICATION: N
 ECHO: AP VALVE PIT
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 292
 LED MODE: NONE
 FUNCTION:
 KEY: 6
 LABEL: AP:HVAC

VERIFICATION: N
 ECHO: AP HVAC
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 294
 LED MODE: NONE
 FUNCTION:
 KEY: 7
 LABEL: AP:FARM
 VERIFICATION: N
 ECHO: AP FARM
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 290
 LED MODE: NONE
 FUNCTION:
 KEY: 8
 LABEL: AW:271
 VERIFICATION: N
 ECHO: AW-271
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 302
 LED MODE: NONE
 FUNCTION:
 KEY: 9
 LABEL: AW:VALVE:PIT
 VERIFICATION: N
 ECHO: AW VALVE PIT
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 303
 LED MODE: NONE
 FUNCTION:
 KEY: 10
 LABEL: AW:HVAC
 VERIFICATION: N
 ECHO: AW HVAC
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 304
 LED MODE: NONE
 FUNCTION:
 KEY: 11
 LABEL: AW:FARM
 VERIFICATION: N
 ECHO: AW FARM
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 300
 LED MODE: NONE
 FUNCTION:
 KEY: 12
 LABEL: AZ:FARM
 VERIFICATION: N
 ECHO: AZ FARM

WHC-SD-534-CSWL-001 REV 3

DISPLAY TYPE:	GR	LED MODE:	NONE
DISPLAY NUMBER:	323	FUNCTION:	
LED MODE:	NONE	KEY:	19
FUNCTION:		LABEL:	244-A
KEY:	13	VERIFICATION:	N
LABEL:	AY:FARM	ECHO:	244-A
VERIFICATION:	N	DISPLAY TYPE:	GR
ECHO:	AY FARM	DISPLAY NUMBER:	340
DISPLAY TYPE:	GR	LED MODE:	NONE
DISPLAY NUMBER:	322	FUNCTION:	
LED MODE:	NONE	KEY:	20
FUNCTION:		LABEL:	
KEY:	14	VERIFICATION:	N
LABEL:	A:FARM	ECHO:	
VERIFICATION:	N	DISPLAY TYPE:	NONE
ECHO:	A FARM	LED MODE:	NONE
DISPLAY TYPE:	GR	FUNCTION:	
DISPLAY NUMBER:	321	KEY:	21
LED MODE:	NONE	LABEL:	
FUNCTION:		VERIFICATION:	N
KEY:	15	ECHO:	
LABEL:	A AX:AY AZ:FARMS	DISPLAY TYPE:	NONE
VERIFICATION:	N	LED MODE:	NONE
ECHO:	A AX AY AZ FARMS	FUNCTION:	
DISPLAY TYPE:	GR	KEY:	22
DISPLAY NUMBER:	320	LABEL:	AP-102:GROUT
LED MODE:	NONE	VERIFICATION:	N
FUNCTION:		ECHO:	AP-102 GROUT TANK
KEY:	16	DISPLAY TYPE:	GR
LABEL:	WASTE:XFR	DISPLAY NUMBER:	291
VERIFICATION:	N	LED MODE:	NONE
ECHO:	WASTE TRANSFER	FUNCTION:	
DISPLAY TYPE:	GR	KEY:	23
DISPLAY NUMBER:	344	LABEL:	204AR:& MISC
LED MODE:	NONE	VERIFICATION:	N
FUNCTION:		ECHO:	204-AR ALARMS
KEY:	17	DISPLAY TYPE:	FP
LABEL:	A-350	DISPLAY NUMBER:	81
VERIFICATION:	N	LED MODE:	NONE
ECHO:	A-350	FUNCTION:	
DISPLAY TYPE:	GR	KEY:	24
DISPLAY NUMBER:	343	LABEL:	
LED MODE:	NONE	VERIFICATION:	N
FUNCTION:		ECHO:	
KEY:	18	DISPLAY TYPE:	NONE
LABEL:	244-A:HVAC	LED MODE:	NONE
VERIFICATION:	N	FUNCTION:	
ECHO:	244-A HVAC	KEY:	25
DISPLAY TYPE:	GR	LABEL:	
DISPLAY NUMBER:	341	VERIFICATION:	N

WHC-SD-534-CSWL-001 REV 3

ECHO:
 DISPLAY TYPE: NONE
 LED MODE: NONE
 FUNCTION:
 KEY: 26
 LABEL:
 VERIFICATION: N
 ECHO:
 DISPLAY TYPE: NONE
 LED MODE: NONE
 FUNCTION:
 KEY: 27
 LABEL:
 VERIFICATION: N
 ECHO:
 DISPLAY TYPE: NONE
 LED MODE: NONE
 FUNCTION:
 KEY: 28
 LABEL: SC: BASIN
 VERIFICATION: N
 ECHO: STEAM CONDENSATE BASINS
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 22
 LED MODE: NONE
 FUNCTION:
 KEY: 29
 LABEL: STEAM: CND8
 VERIFICATION: N
 ECHO: STEAM CONDENSATE
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 14
 LED MODE: NONE
 FUNCTION:
 KEY: 30
 LABEL: PB-1
 VERIFICATION: N
 ECHO: PB-1 PUMP
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 12
 LED MODE: NONE
 FUNCTION:
 KEY: 31
 LABEL: PUMP: AW-102
 VERIFICATION: N
 ECHO: AW-102 FEED PUMP
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 301
 LED MODE: NONE
 FUNCTION:

KEY: 32
 LABEL:
 VERIFICATION: N
 ECHO:
 DISPLAY TYPE: NONE
 LED MODE: NONE
 FUNCTION:
 KEY: 33
 LABEL: E-A-1
 VERIFICATION: N
 ECHO: E-A-1 REBOILER
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 13
 LED MODE: NONE
 FUNCTION:
 KEY: 34
 LABEL: C-A-1
 VERIFICATION: N
 ECHO: C-A-1 EVAPORATOR
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 10
 LED MODE: NONE
 FUNCTION:
 KEY: 35
 LABEL: PB-2
 VERIFICATION: N
 ECHO: PB-2 PUMP
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 15
 LED MODE: NONE
 FUNCTION:
 KEY: 36
 LABEL: IX-D-1
 VERIFICATION: N
 ECHO: IX-D-1 / RC3
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 44
 LED MODE: NONE
 FUNCTION:
 KEY: 37
 LABEL: TK: C-100
 VERIFICATION: N
 ECHO: TK-C-100
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 18
 LED MODE: NONE
 FUNCTION:
 KEY: 38
 LABEL: E-C-1: COND
 VERIFICATION: N

WHC-SD-534-CSWL-001 REV 3

ECHO: E-C-1 VACUUM CONDENSER
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 16
 LED MODE: NONE
 FUNCTION:
 KEY: 39
 LABEL: VESSL:VENT
 VERIFICATION: N
 ECHO: VESSEL VENT
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 20
 LED MODE: NONE
 FUNCTION:
 KEY: 40
 LABEL: LERF
 VERIFICATION: N
 ECHO: L.E.R.F.
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 24
 LED MODE: NONE
 FUNCTION:
 KEY: 41
 LABEL:
 VERIFICATION: N
 ECHO:
 DISPLAY TYPE: NONE
 LED MODE: NONE
 FUNCTION:
 KEY: 42
 LABEL: USED:RAW:WATER
 VERIFICATION: N
 ECHO: USED RAW WATER
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 17
 LED MODE: NONE
 FUNCTION:
 KEY: 43
 LABEL: RAW:WATER
 VERIFICATION: N
 ECHO: RAW WATER SUPPLY
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 11
 LED MODE: NONE
 FUNCTION:
 KEY: 44
 LABEL: CASS
 VERIFICATION: N
 ECHO: CASS ALARMS
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 90

LED MODE: NONE
 FUNCTION:
 KEY: 45
 LABEL: AMU
 VERIFICATION: N
 ECHO: AQUEOUS MAKE UP
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 19
 LED MODE: NONE
 FUNCTION:
 KEY: 46
 LABEL: UTILS
 VERIFICATION: N
 ECHO: UTILITIES
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 21
 LED MODE: NONE
 FUNCTION:
 KEY: 47
 LABEL: 242A:RADS
 VERIFICATION: N
 ECHO: 242-A RADIATION / VENT
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 23
 LED MODE: NONE
 FUNCTION:
 KEY: 48
 LABEL: PROGRAM:STATUS
 VERIFICATION: N
 ECHO: SEQUENCE PROGRAM STATUS
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 43
 LED MODE: NONE
 FUNCTION:
 KEY: 49
 LABEL: CND8:OVER-:VIEW
 VERIFICATION: N
 ECHO: CONDENSATE OVERVIEW
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 42
 LED MODE: NONE
 FUNCTION:
 KEY: 50
 LABEL: VACUM:OVER-:VIEW
 VERIFICATION: N
 ECHO: VACUUM CONDENSER OVERVIEW
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 41
 LED MODE: NONE
 FUNCTION:

KEY: 51
 LABEL: EVAP:OVER-VIEW
 VERIFICATION: N
 ECHO: EVAPORATOR OVERVIEW
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 40
 LED MODE: NONE
 FUNCTION:
 KEY: 52
 LABEL:
 VERIFICATION: N
 ECHO:
 DISPLAY TYPE: NONE
 LED MODE: NONE
 FUNCTION:
 KEY: 53
 LABEL: INHA
 VERIFICATION: N
 ECHO: INHIBITED ALARMS
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 80
 LED MODE: NONE
 FUNCTION: ES
 UNIT NAME: WHOLEO
 PROGRAM NAME: TIMEINHA
 FUNCTION: EL
 NAME: INHALM
 FUNCTION:
 KEY: 54
 LABEL: HISTL:TREND:MENU
 VERIFICATION: N
 ECHO: HISTORICAL TREND MENU
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 9
 LED MODE: NONE
 FUNCTION:
 KEY: 55
 LABEL: CURR:TREND:MENU
 VERIFICATION: N
 ECHO: CURRENT TREND MENU
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 8
 LED MODE: NONE
 FUNCTION:
 KEY: 56
 LABEL:
 VERIFICATION: N
 ECHO:
 DISPLAY TYPE: NONE
 LED MODE: NONE

FUNCTION:
 KEY: 57
 LABEL:
 VERIFICATION: N
 ECHO:
 DISPLAY TYPE: NONE
 LED MODE: NONE
 FUNCTION:
 KEY: 58
 LABEL:
 VERIFICATION: N
 ECHO:
 DISPLAY TYPE: NONE
 LED MODE: NONE
 FUNCTION:
 KEY: 59
 LABEL: ROUNDS:MENU
 VERIFICATION: N
 ECHO: ROUNDS MENU
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 501
 LED MODE: NONE
 FUNCTION:
 KEY: 60
 LABEL: WHITE:ALARM
 VERIFICATION: N
 ECHO: WHITE ALARMS
 DISPLAY TYPE: GR
 DISPLAY NUMBER: 100
 LED MODE: NONE
 FUNCTION: MS
 UNIT NAME: WHOLE1
 TYPE: S
 VARIABLE NAME: FTRIPO
 STATUS: ON
 FUNCTION:
 KEY: 61
 LABEL: NOTE:BOOK
 VERIFICATION: N
 ECHO: NOTE BOOK
 DISPLAY TYPE: GF
 DISPLAY NUMBER: 4
 LED MODE: NONE
 FUNCTION:
 KEY: 62
 LABEL: EPN:ON/OFF
 VERIFICATION: N
 ECHO: EPN ON/OFF
 DISPLAY TYPE: NONE
 LED MODE: NONE

WHC-SD-534-CSWL-001 REV 3

FUNCTION: ES
UNIT NAME: WHOLEO
PROGRAM NAME: TAG
FUNCTION:
KEY: 63
LABEL:
VERIFICATION: N
ECHO:
DISPLAY TYPE: NONE
LED MODE: NONE
FUNCTION:
KEY: