MULTIPLEXER CODES FOR THE PDP-7

N. P. WILBURN

FEBRUARY 1968

**AEC RESEARCH &
DEVELOPMENT REPORT**

Project No.	Project Title	Project Status	Project Date
1-117-1	20823 JJS	ALL	10 1968

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

PACIFIC NORTHWEST LABORATORY

RICHLAND, WASHINGTON

operated by

BATTELLE MEMORIAL INSTITUTE

for the

UNITED STATES ATOMIC ENERGY COMMISSION UNDER CONTRACT AT(45-1)-1830

PRINTED BY/FOR THE U. S. ATOMIC ENERGY COMMISSION

3 3679 00060 7079

BNWL-597

UC-80, Reactor Technology

MULTIPLEXER CODES FOR THE PDP-7

By

N. P. Wilburn

Reactor Engineering Section
Engineering Development Department

February, 1968

(2)

100-100-000

PACIFIC NORTHWEST LABORATORY
RICHLAND, WASHINGTON

Printed in the United States of America
Available from
Clearinghouse for Federal Scientific and Technical Information
National Bureau of Standards, U. S. Department of Commerce
Springfield, Virginia 22151
Price: Printed Copy \$3.00; Microfiche \$0.65

MULTIPLEXER CODES FOR THE PDP-7

N. P. Wilburn

ABSTRACT

A small digital computer (PDP-7) has been interfaced to a low-speed relay multiplexer (2000 points/sec) and a high-speed solid-state multiplexer (240,000 points/sec). The three codes which have been written to operate these multiplexers and to record the data collected onto magnetic tape, are described in this report.

TABLE OF CONTENTS

ABSTRACT	iii
LIST OF TABLES	vii
INTRODUCTION	1
DESCRIPTION OF HARDWARE	1
General	1
Low Speed Multiplexer	3
High Speed Multiplexer	4
Time Zero Interrupt	6
GENERAL DESCRIPTION OF THE MULTIPLEXER CODE	6
Function Programs	6
Executive Control Program	7
Interrupt Service Programs	8
Queues	8
Initialization	9
Scanner Control	9
Control of the DECTAPE Units	16
Startup Program	18
MPGEN2	21
OPERATIONAL DETAILS.	22
Halting and Restarting	24
Error Messages	25
DETAILED DESCRIPTION OF CSEMPLEX	25
CSEMPLEX 01	26
CSEMPLEX 02	26
CSEMPLEX 03	26
CSEMPLEX 04	26
CSEMPLEX 05	27
CSEMPLEX 06	27
CSEMPLEX 07	27
CSEMPLEX 08	27
CSEMPLEX 09	28

TABLE OF CONTENTS (contd)

CSEMPLEX 10	28
CSEMPLEX 11	29
CSEMPLEX 12	30
CSEMPLEX 13	30
CSEMPLEX 14	30
CSEMPLEX 15	30
CSEMPLEX 16	31
CSEMPLEX 19	31
REFERENCES	32
APPENDIX A. ENTRY OF CLKM AND CLKL UPON RESTART .	A-1
APPENDIX B. PROGRAM LISTING	B-1
DISTRIBUTION	Distr-1

LIST OF TABLES

I	Computer Point Designation Versus External Connector	12
II	Schematic of CLPER Table	13
III	Schematic of Data Storage on DECTAPE	14
IV	Definition of Startup Table (SUT) Commands	19
V	Typical Startup Table (SUT)	20
VI	Relay Buffer Control Circuits	21
B-I	Listing of MPGEN2 Code	B-1
B-II	Typical Teletype Input-Output When Using MPGEN2	B-23
	A. Accumulator Switches Cleared	B-23
	B. Switch 1 Set	B-26
	C. Switches 1, 2 Set	B-28
	D. Switches 1, 2, 3 Set	B-30
B-III	Listing of Multiplexer Test Code	B-32
B-IV	Listing of CSEMPLEX Code	B-44
B-V	Octal Dump of DECTAPE Data from Example	B-85

MULTIPLEXER CODES FOR THE PDP-7

N. P. Wilburn

INTRODUCTION

Late in 1963, a program entitled, "The Containment System Experiment," was undertaken by the Battelle Memorial Institute, Pacific Northwest Laboratory. The primary purpose of the CSE program is to evaluate on a significant engineering scale the effectiveness of natural processes and engineering safeguards in limiting the release of fission products from nuclear reactor accidents to the environment.⁽¹⁾ The secondary purpose is to investigate engineering aspects of a contained coolant loss accident, including the dynamic effects of a rupture in high temperature coolant piping and the leakage characteristics of the representative containment system. In order to measure these dynamic effects, a large number of instruments have been installed on the containment system vessels. In order to ease the processing of the large amount of data to be obtained from this experiment, it was decided to collect the data by means of a high speed multiplexer-digital computer combination and store the data on magnetic tape, rather than to use a series of analogue recording devices which would later have to be read by skilled technicians. This procedure minimizes the amount of human error inherent in conversion of analogue recorded data. In order to carry out the data logging, a PDP-7 computer, manufactured by the Digital Equipment Corporation was purchased, and additional equipment was added to the basic system. It is the purpose of this document to describe the computer codes which have been written to operate the computer in conjunction with the locally added equipment.⁽³⁾

DESCRIPTION OF HARDWARE

GENERAL

The PDP-7 computer is basically the standard system supplied by Digital Equipment Corporation.⁽²⁾ The detailed description follows and is shown in a schematic diagram, Figure 1. The machine uses a high speed

FIGURE 1. Schematic of PDP-7 Hardware

optical paper tape reader capable of reading 8-channel, 1-in. paper tape of the fanfold variety, and it is also capable of punching this tape. The central processing unit has the "normal" arithmetic option and is supplied with an 8,192-word core of 18 bits per word. Two teletypes, model 33 ASR, were supplied with the machine, and a switch permits the operation of one teletype unit placed near the computer and the other, located approximately 150 ft from the computer in a counting room. Magnetic tape units, a DECTAPE Type 555 dual transport, and two DECTAPE Type TU55 single transports were also included.

A real time clock which produces either a pulse every 1/60 sec or every 0.2 millisecc, depending upon a switch setting, is connected to the data break and causes memory location 7 to be incremented by 1 upon every pulse. If the contents of location 7 overflow, the clock flag is set to 1. The clock flag is connected to the program interrupt system.

The machine is also equipped with a relay buffer which consists of an 8-bit relay register and contains filters to reduce noise due to contact bounce. Under program control, the flip-flop register can be set to correspond to the contents of the accumulator and can be cleared. Each bit of the flip-flop register in the binary 1 condition energizes an associated relay in the relay register. External connection to the relay-contacts is accomplished by terminal strips at the back of the relay mounting panel.

LOW SPEED MULTIPLEXER

The computer has been interfaced to a DEC Type 138 general purpose analogue-to-digital converter. The Type 138 is a successive approximation converter that measures a zero to minus 10-volt analogue input signal and produces a binary output indication of the amplitude of the input signal. Upon the front panel of the analogue-to-digital converter is a control used to set the word length in bits, which is variable from 6 to 12 bits, and a control to set the maximum switching point error, which determines the pulse rate of conversion. The accuracy of conversion versus the settings of these controls is given in the PDP-7 USERS HANDBOOK. ⁽²⁾

The analogue-to-digital converter was combined with a multiplexer system consisting of relay and solid state switches, which was designed locally.⁽³⁾ The multiplexer consists of 480 reed type-single-pole-single-throw relays which are actuated 12 at a time by means of a computer command. The relays serve to switch incoming analogue signals connected to three wires per point: the high, the low, and the guard. The relays are physically mounted three per board on a fiber plug-in board which can be used to mount additional electronic components for signal conditioning.

The four signals from the 12 relays are then connected to four precision analogue amplifiers having gains of 300. These amplifiers are low drift and have accuracies of approximately 0.01 of 1%. The signals are amplified from approximately 20 mV to approximately 2 to 10 V by means of the amplifiers. The resulting high level signals are presented to a four input solid state switch. The operation of the switch is controlled by a computer IOT commands.^(2, 3) Subpoint 0 can be selected and then the switch can be advanced to subpoints 1, 2, or 3 in a sequential fashion.

The operation time of the relays and the setting time of the amplifiers totals about 1.5 msec. Selecting or advancing the solid state switch takes approximately 2 msec, and the analogue-to-digital conversion takes approximately 100 msec. Thus, the four points selected by means of a computer command through the relay matrix can be scanned and converted to digital equivalents in approximately 2 msec. This gives an overall maximum scanning rate for this multiplexer system of approximately 2000 points/sec.

Long term stability tests on this multiplexer system have indicated that a signal can be monitored and converted with about 0.1% standard deviation. Thus, when monitored for 24 hr, a 10 V signal had a range of readings approximately equal to 0.25%.

HIGH SPEED MULTIPLEXER

The low speed multiplexer is capable of monitoring most of the signals which will be obtained from the blowdown during a typical CSE

experiment. However, in order to monitor pressures resulting from the accoustical waves occurring in the primary simulator vessel, a much higher speed data logging device had to be incorporated. It is desired to monitor up to 24 signals at approximately 10,000 points/sec each. Hence, this gives a band width for each of the signals of approximately 5000 cycles/sec.

To accomplish this data logging, requiring an overall gross scanning rate of 240,000 points/sec, it was necessary to design a multiplexer analogue-to-digital converter combination with a different concept. First, three Type 138E analogue-to-digital converters manufactured by Digital Equipment Corporation were required which would operate asynchronously with the computer, and, upon being enabled, would enter the data directly into the computer memory by means of the data break. In addition, three multiplexers, DEC Type 139E, with 8 solid state switches each, DEC Type A103, were also required. With this configuration and using a six or seven-bit maximum conversion accuracy, the required scanning rate can be obtained.

Each one of the 24 analogue points requires its own amplifier to obtain a signal in the range of 0 to 10 V, which is large enough to be effectively switched by the solid state switches. Hardware has been provided to interface each of the multiplexers through the data break. The computer merely enables and disables the multiplexers by means of IOT's.⁽²⁾ When enabled, the multiplexer is reset to the 0 channel and proceeds to cycle through each of its 8 points in sequence, converting each point, and storing the result directly into an assigned memory location by means of the data break. When the memory assigned for the particular multiplexer is exhausted, the multiplexer automatically terminates operation, and sets a flag which the computer can interrogate to ascertain that the multiplexer had ceased operation.

The 24 inputs are divided into 3 subsets of 8 each, a subset per multiplexer. Each input in a subset can be logged 128 times. Thus, it can be monitored for a period of about 12.8 msec. The logging of the

inputs within a subset occurs sequentially and continuously until each voltage is logged the 128 times. However, the logging of each subset can be initiated at different times. An earlier version of this code started monitoring all three subsets simultaneously. Review of the data to be collected indicated this was not desirable. Roughly, an acoustic pressure wave is to be monitored at different stations along a pressurized vessel. After the sudden opening of one end of the vessel (indicated to the computer by an interrupt described below) the pressure wave takes a finite time to propagate along the vessel. Thus, it seems desirable to place within a subset voltages common to a station and then delay the monitoring of the subset as determined by propagation times.

Time Zero Interrupt

The computer has been provided with an interrupt which is actuated when a ground connection is broken. This is used to indicate when the containing rupture disk on the reactor simulator vessel is broken. This event indicates "time zero" and causes operation of the 3 high speed multiplexers to commence after programmed delays.

GENERAL DESCRIPTION OF THE MULTIPLEXER CODE

The code, CSEMPLEX, which has been written to operate the low speed, 2000 point/sec multiplexer, and the high speed, 240,000 point/sec multiplexer, can be described as a real time multiprogrammed-multithread code.⁽³⁾ Similar to most codes of this type, this code consists of three major sections: (1) the executive control program which decides which of the several function programs are to be operated, (2) the function programs, and (3) the interrupt service programs. In addition, there are several subroutines which are used in common by the interrupt service routines and the function programs.

FUNCTION PROGRAMS

This code has provision for seven function programs, numbered 0 through 6. Program 0 is an executive control program; 1 is the scan program which controls the low speed multiplexer; 2 is the program to

initiate transfers from core memory to the DECTAPE units; 3 is the input-output program which controls the typeout of messages on the teletyper; 4 is a routine to scan a table of commands to be executed during startup; 5 is a spare position not currently used; and 6 is a program which initiates operation of the high speed multiplexers and upon their completion disables them and causes the stored data to be written out to magnetic tape. For each of these programs, there exists eight words in a table called PROG. These are numbered 0 through 7. Word 0 is either ZERO or NOT ZERO to indicate that the program is to be not operated or operated respectively. Word 1 is the final value of the accumulator when the program was last operated. Word 2 is the entry address upon first-time operation of the program. Word 3 contains return address to which the program control is to flow when this function program is to next be operated. Bit 0 of this word contains the final value of the link. Word 4 contains the value of the auto-index register, memory location 10, which was present when control left this function program. Words 5 through 7 are not used.

EXECUTIVE CONTROL PROGRAM

Entry to the function program is controlled by the executive control program which scans the word 0 entry for each of the function programs in the table PROG. When the 0 word is not equal to 0, the executive program resets the auto-index register 10, the accumulator, and the link to their former values and turns the control over to the program. The function program has control until at some point in the program-either a WAIT instruction or an EXIT instruction is encountered. The WAIT instruction is used when the function program is waiting for some external event to occur such as a magnetic tape data flag, a typewriter entry, or some other phenomenon. The contents of auto-index register and the contents of the accumulator, and the return address, are saved by means of the WAIT subroutine. When an EXIT is encountered, the auto-index register storage in the PROG array is set to zero as is the accumulator storage, and the return address of word 3 is set to the initial entry address located in word 2. Then, control is turned over to the executive control program.

INTERRUPT SERVICE PROGRAMS

During operation of any of the function programs or the executive control program, control can be momentarily taken away from these programs by means of a hardware interrupt. Upon interrupt, the contents of the program address register are stored in memory location 0, and the link is stored in Bit 0 memory location 0. Control is then turned over to location 1. At this point, a jump to the interrupt service routine is made. The interrupt service determines what hardware device caused the interrupt and then jumps to the service routine which then takes care of the interrupt. Hardware interrupts can be caused by a number of external devices, such as: the teleprinter, the keyboard, the low speed scanner, the magnetic tape data flag, block flag, or error flag, the "time zero" interrupt caused by the break of the rupture disk wires, the real time clock, the paper tape reader, and the paper tape punch. All of these hardware interrupts are serviced except for the last two in this multiplexer code. When an interrupt service routine is completed the normal exit is a jump to CONTIN which restores the accumulator and returns control to the executive control program or the function program.

QUEUES

In order to simplify the handling of information in a real time multiprogrammed system, extensive use is made of queues. In this particular program, four queues are utilized. The first queue is a storage queue. The memory which is not used for programmed instructions is divided into blocks of 400_g which match the blocks on the magnetic DECTAPE units. This consists of the memory area from 4000_g to 17777_g . The storage queue handler assigns storage to a requesting program through the subroutine STOREQ and returns storage to the queue through the subroutine STORAD. At the present time, storage is requested only by the scanning program (No. 1) wherein the incoming analogue data is stored, and the storage block is returned to the queue by the magnetic tape handling routines after the data is transferred to the DECTAPE.

The second queue is used to designate what areas of memory are to be transferred to magnetic tape by the magnetic tape handler. The scanning program places the addresses of storage blocks that have been filled by the scanning routines into the magnetic tape queue for transfer to magnetic tape. When the transfer is completed, the magnetic tape handlers return the storage block to the storage queue by means of the STORAD subroutine.

The third queue is used for handling messages to be typed on the teletyper. The requesting function program enters the TYP OUT subroutine with the first address of the message to be typed, and the subroutine enters this address into the teletype queue, TQUEUE. At the same time, the input-output program is turned on and this program breaks the message up into its individual ASCII characters which it stores into the queue TELBUF, the fourth queue used by this code. The individual characters are removed by the teletype interrupt service routine and typed.

INITIALIZATION

The initialization routine, which is entered by setting the address switches to 20 and pressing the START switch, returns all programs and subprograms to their initial undisturbed condition which was present upon "read-in" of the program from the "funny format" paper tape. This allows debugging and restarting to proceed without having to reread the "funny format" paper tape into the core every time.

SCANNER CONTROL

The blowdown experiment to be monitored by this multiplexer code consists of a long steady state operational period previous to the blowdown, a very short period right after blowdown when events are happening very rapidly, then several periods of time in which the events occur progressively more slowly (See Figure 2). In order to adequately monitor the blowdown experiment during all phases, the design of the scanner control program incorporates seven distinct periods of operation. During the Period 0 previous to blowdown, all incoming analogue voltages to the low

FIGURE 2. Division of Data Collection into Periods

speed multiplexer are monitored at a single frequency determined by the SCTM table. During the successive periods, it is assumed that the different incoming voltages will probably be monitored at different scanning rates, depending upon the frequency components present in the incoming analogue voltage.

During each period of time, up to seven possible different scanning rates or classes, can be used. These scanning rates can be changed from period to period, thus, during the first period right after blowdown, the scanning rates might be in multiples of tens of milliseconds, whereas in the second period, it might be in multiples of seconds, and the later periods in multiples of minutes. The scanning rates during each period are specified by entries into the SCTM table. The points to be scanned, as well as the Class or rate, is specified by the CLPER table. In all periods, the scanning rates are limited by the maximum scanning capacity of the low speed multiplexer, 2000 points/sec. Thus, the sum of the number of points in a given scan class divided by the scanning time of that class must be less than 2, when the scanning times are expressed in milliseconds.

The CLPER table consists of one word for each of the 60_8 major points which are scanned by the low speed multiplexer and listed in Table I. The individual words are divided up into six sets of three bits each corresponding to the six periods subsequent to the blowdown (Table II). The special Period 0 is unique in that all points are scanned in a single class at a single rate. The 3 bits assigned to each of the periods are adequate to specify which of the seven scan classes is desired for the data point. The 3 bits being equal to 0 means that the point is not to be scanned during this period.

The times when the periods are ended are specified in the table PERT which consists of seven double precision integers corresponding to the seven periods. These are compared with the internal clock (CLKM and CLKL) which keeps track of time in milliseconds. When the internal clock equals the time for a particular period, changes are carried out in the scanning program to initiate operation of the events for the subsequent period.

TABLE I. Computer Point Designation Versus External Connector

Major Point Number	Subpoint 0	Subpoint 1	Subpoint 2	Subpoint 3
00	A0	B0	C0	D0
01	A1	B1	C1	D1
02	A2	B2	C2	D2
03	A3	B3	C3	D3
04	A4	B4	C4	D4
05	A5	B5	C5	D5
06	A6	B6	C6	D6
07	A7	B7	C7	D7
10	A10	B10	C10	D10
11	A11	B11	C11	D11
12	A12	B12	C12	D12
13	A13	B13	C13	D13
14	A14	B14	C14	D14
15	A15	B15	C15	D15
16	A16	B16	C16	D16
17	A17	B17	C17	D17
20	E0	F0	G0	H0
21	E1	F1	G1	H1
22	E2	F2	G2	H2
23	E3	F3	G3	H3
24	E4	F4	G4	H4
25	E5	F5	G5	H5
26	E6	F6	G6	H6
27	E7	F7	G7	H7
30	E10	F10	G10	H10
31	E11	F11	G11	H11
32	E12	F12	G12	H12
33	E13	F13	G13	H13
34	E14	F14	G14	H14
35	E15	F15	G15	H15
36	E16	F16	G16	H16
37	E17	F17	G17	H17
40	J0	K0	n. c.	n. c.
41	J1	K1	n. c.	n. c.
42	J2	K2	n. c.	n. c.
43	J3	K3	n. c.	n. c.
44	J4	K4	n. c.	n. c.
45	J5	K5	n. c.	n. c.
46	J6	K6	n. c.	n. c.
47	J7	K7	n. c.	n. c.
50	J10	K10	n. c.	n. c.
51	J11	K11	n. c.	n. c.
52	J12	K12	n. c.	n. c.
53	J13	K13	n. c.	n. c.
54	J14	K14	n. c.	n. c.
55	J15	K15	n. c.	n. c.
56	J16	K16	n. c.	n. c.
57	J17	K17	n. c.	n. c.

TABLE II. Schematic of CLPER Table

Major Point I.D. or Memory Cell Mod 100_8	<u>Scan class in which point is logged (1-7)*</u>																	
	Bit assignment																	
	<u>0</u>	<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>	<u>5</u>	<u>6</u>	<u>7</u>	<u>8</u>	<u>9</u>	<u>10</u>	<u>11</u>	<u>12</u>	<u>13</u>	<u>14</u>	<u>15</u>	<u>16</u>	<u>17</u>
00			4					2							1			0
01			5					1							0			0
02			4					4							0			0
.			.					.							.			.
.			.					.							.			.
.			.					.							.			.
.			.					.							.			.
.			.					.							.			.
.			.					.							.			.
.			.					.							.			.
57			<u>0</u>					<u>0</u>							<u>0</u>			<u>0</u>
Period index			6					5							2			1
																		0 = (All points logged)

* 0 = Do not log during this period

The SCTM table controls the scanning rate for each of the seven classes during each of the seven periods. The interval between scans is stored in milliseconds in SCTM, and a negative number in this table means that the Class is not scanned during this period. A minimum scanning rate of 131,071 msec between scans is required because the table consists of single precision integers.

When the data is collected by the scan program and stored into memory in blocks of 400_8 for later writing out to magnetic tape, the format followed is presented in Table III. As was noted in the description of the hardware, the selection of the relay matrix actually picks up four data points simultaneously. These are identified by a major point number which varies between 0 and 57 octal in the present system. This major point number is stored in this data array in word zero (mod 4) in bits

TABLE III. Schematic of Data Storage on DECTAPE
($256_{10} = 400_8$ Words/Block)

Low Speed Multiplexer Data

<u>Address Mod 4</u>	<u>Bit Assignment</u>					
	0	11	12		17	
0	Subpoint 0 Data		Major Point #			
	0	11	12	14	15	17
1	Subpoint 1 Data		Scan Class		Period	
	0	11	12			17
2	Subpoint 2 Data		Scan Class Index Mod 100_8			
	0	11	12	13		16 17
3	Subpoint 3 Data		0	Bits 14-17 * of CLKL		

*0 = Negative Time (Prerupture)

1 = Positive Time (Postrupture)

High Speed Multiplexer Data

<u>Word Mod 400_8</u>	<u>Bit Assignment</u>					
	0	11	12	13-14	15	17
0	Data		1	(a)	(c)	
	0	11	12	13-14	15	17
1 - 377	Data		1	(a)	(b)	

(a) High Speed Multiplexer Number - 00, 01, 10, 11_2

(b) Data Point Mod 10_8 for this Multiplexer

(c) Block Sequence Number for this Multiplexer = 0, 1, 2, 3_8

12 through 17. The data obtained from each of the subpoints 0, 1, 2, 3 (mod 4) are in the computer memory. Word 1 (mod 4) contains the scan class in which the major point was scanned for the period indicated in bits 15 through 17 of this word. Word 2 (mod 4) contains in bits 12 to 17 a scan class index (mod 100_8). This information simply indicates the number of times this point has been scanned in this class during this period. From this information and the period time indicated in PERT table, and the scan class time indicated in the SCTM table, the actual time after some arbitrary zero point that this point was scanned, can be computed. Word 3 bits 13-16 contains the least significant bits of the current clock reading CLKL. Time 0 is flagged in the data stored by the scanner in bit 17 of word 3 (mod 4). Previous to the blowdown rupture, bit 17 of word 3 (mod 4) is zero. After the rupture, this bit is set to 1. From the information in the table SCTM and PERT and the scan class index, time zero can be determined within 1 to 2 msec.

In order to distinguish the data coming from the high speed multiplexer from that coming from the low speed multiplexer, bit 12 of word 3 (mod 4) of the data from the low speed multiplexer is zero, whereas data coming from the high speed multiplexer has bit 12 of all data words set to 1, by hardware.

The format of the data words coming from the high speed multiplexer is also shown in Table III. All of the data in a given block of 400 words come from either a low speed multiplexer or from one of the high speed units. Bits 0 to 11 for each of the high speed multiplexers contain the data obtained from the analogue-to-digital converter. Bit 12 is set to 1 for all of the multiplexers. Bits 13 and 14 identify which of the three multiplexers is being used, and bits 15 to 17 identify the data point (mod 8) on the individual multiplexers. The high speed data are accumulated automatically as was described in the hardware section. Upon interrupt from the rupture of the wires located on the rupture disk, the three high speed multiplexers are enabled and proceed to store data into memory areas 10000_8 to 11777_8 , 12000_8 to 13777_8 , 14000_8 to 15777_8 , respectively. The code, as designed, allows for a fourth multiplexer which will store data into 16000_8 to 17777_8 .

CONTROL OF THE DECTAPE UNITS

The DECTAPE Units are verified to be in the "write enabled" condition by the initialization program. If this is not the case, a continuous rocking motion will be noted by the operator. This can be corrected by setting the write switch to the enable position. Units 1, 2, 3, and 4 are rewound to a position approximately over block 0 on the magnetic tape. Thus, the amount of time to find and begin writing block 5 on any tape unit is minimized.

As was pointed out above, the operation of the DECTAPE is controlled by the magnetic tape function program (No. 3) through use of a magnetic tape queue. As entry made into the magnetic tape-queue causes the magnetic tape function program to be turned on, and the subsequent block number to be searched for, and the data to be written out on to DECTAPE. The relative timing of the DECTAPE operations and the data accumulation by means of the low speed and high speed multiplexers are shown schematically in Figure 3. Here, it has been assumed that the high speed multiplexer data is collected essentially instantaneously and the low speed multiplexer data is collected at a rate of 256 words in 125 msec. Fifty milliseconds are required to write one block of words, 150 msec either to start or to stop movement of the DECTAPE unit, and 300 msec for the tape unit to reverse directions. As is seen from the schematic, normal operation permits several blocks to be written onto magnetic tape at one time because of the simultaneous accumulation of data with that of writing the data out to magnetic tape.

If a magnetic tape error occurs during the writing of the data on to DECTAPE, the message MTE is typed followed by five digits which uniquely identify the DECTAPE unit and the block number where the error occurred. The magnetic tape handler bypasses this block, goes on, and tries to write the same data on the block following. Thus, if a bad spot is encountered on the magnetic tape, it is simply bypassed and a rewrite over the same area is not attempted. This was done in order to minimize the amount of time spent in trying to rewrite over a defective spot and the possibility of tying up the computer in an endless loop.

ASSUMPTIONS

Time to Collect 1 Block Data = 125 Msec

Time to Write 1 Block Data = 50 Msec

Time to Start (or Stop) Tape = 150 Msec

12 Blocks from High Speed Multiplexer at 0 Msec

Blocks Added
to Queue

DECTAPE
Searching

DECTAPE
Writing
Blocks

Time (Msec)

FIGURE 3. Comparison of Data Accumulation and Data Transfer to DECTAPE

When all of the possible information has been written onto a given DECTAPE unit, the message REPLACE TAPE followed by the number of the unit is typed on the teletype. At this time, the operator should remove the tape, replace it with another tape, position it approximately at the beginning, and set the write switch to WRITE ENABLE. If the magnetic tape handler program senses that the tape is "unable" it will type out the message UNABLE, and the DECTAPE will rock back and forth to indicate to the operator that this is the off ending unit.

STARTUP PROGRAM

The startup program is used for two purposes: First to read-in all of the control tables such as the startup control, the CLPER table, the SCTM table, and the PERT table. This is initiated by typing a letter T on the teletype keyboard. The computer will halt to allow a paper tape in "read-in" format to be mounted in the optical reader. It is then read-in when CONTINUE is pressed. This tape is prepared by the MPGEN2 program to be described later.

Secondly, the startup program is used to control the sequence of events leading up to the rupture and the blowdown experiment. Through the commands, which are described in detail in Table IV, the startup program controls external events by means of the relay buffer. There are currently six commands which have been specified. An octal zero is used as a pure time delay. Its use delays the sequence of operations by a number of milliseconds which are specified by the two succeeding entries into the startup table treated as a double precision integer. An octal 1 is used to set the relay buffer to the bit configuration which is shown in the next succeeding word. The octal 2 command requires knowledge of the memory locations utilized by the CSEMPLEX program. These can be obtained from the paper tape assembler at the last step.⁽²⁾ Principally, this command is used while debugging the program because by its use various memory locations can be altered during the course of startup. The octal command 3 can also be used in this manner. This command allows the operator to execute a series of three instructions which are

TABLE IV. Definition of Startup Table (SUT) Commands

<u>Command</u> (mod 8)	<u>Definition</u>
000000 XXXXXX XXXXXX	Delay sequence of startup operations XXXXXX XXXXXX millisec where X...X is a double precision integer value.
000001 XXXXXX	Set contents relay buffer to bit configuration of XXXXXX. (1)
000002 XXXXXX YYYYYY	Set contents of memory location XXXXXX to YYYYYY.
000003 XXXXXX YYYYYY ZZZZZZ	Execute the three following instructions XXXXXX, YYYYYY, ZZZZZZ after placing them in <u>SJ03I</u> to <u>SJ03I+2</u> , respectively.
000004	Terminate startup.
000005 XXXXXX	Restart clock with CLKCON = XXXXXX. XXXXXX should be equal to LAM - A*5 + 1 where A is the basic clock increment in millisec.
000006 000007	) Undefined, will cause computer to halt.

entered successively in the table after the command. These instructions are executed at symbolic locations SJ03I, SJ03I+1, and SJ03I+2, respectively. They might be used to carry on various IOT instructions such as the example shown in Table V, which is used to arm the time zero flip-flop a very few milliseconds before the blowdown is initiated. Octal command 5 is used to set the interrupt frequency of the real time clock. This command would be used only in the case it was desired to have some interrupt frequency other than 1 msec which is currently specified. Commands 6 and 7 are currently undefined and will cause the computer to halt in the startup program.

A typical example of a startup table which would be prepared by the MPGEN2 code is shown in Table V. As is seen, it is used to sound alarms, turn on various instruments, strip charts, cameras,

etc. It is also used to enable the primary blowdown circuit and to fire the shot. Finally, after the shot has been fired, the circuits are disabled which control the various recorders. Thus, using a similar set of startup commands, the complete blowdown operation can be controlled by the computer in a stepwise sequence independent of operator error. In the example, the relay buffer connections which are shown in Table VI are assumed. These are not currently connected but are merely used for example purposes.

TABLE V. Typical Startup Table (SUT)

<u>Memory Location</u>	<u>Octal Word</u>	<u>Remarks</u>
3500 + 0	000005*	Start internal computer clock
1	777774*	to interrupt at 1 msec intervals
2	000002*	Start scanning
3	020022*	
4	000001*	
5	000000	Delay 10 minutes = $600,000_{10}$ msec
6	000002	= 2223700_8 msec
7	223700	
10	000001	Enable klaxon
11	004000	
12	000000	Delay 4 minutes - $240,000_{10}$ msec
13	000000	
14	724600	= $724,600_8$ msec
15	000001	Turn on neutron beam and close valves
16	034000	
17	000000	Delay 59.5 sec = $59,500_{10}$ msec
20	000000	= $164,154_8$ msec
21	164154	
22	000001	Enable blowdown circuit, strip charts,
23	374000	camera
24	000000	Delay 0.5 sec = 500_{10} msec
25	000000	= 764_8 msec
26	000764	
27	000003	Arm "time zero" interrupt
30	704202	
31	740000	
32	740000	
33	000001	Fire shot
34	774000	
35	000000	Delay 2.0 sec = $2,000_{10}$ msec
36	000000	= 3720_8 msec
37	003720	
40	000001	Disable fire and blowdown circuits,
41	030000	strip charts, camera, and klaxon
42	000004	Terminate Startup Procedure

* First five table entries are placed automatically by MPGEN2 code.

TABLE VI. Relay Buffer Control Circuits

<u>ORS With Accumulator Bit = 1</u>	<u>Circuit Description</u>
0	Initiates blowdown
1	Enables blowdown circuit
2	Enables strip chart recorders
3	Enables high speed motion picture camera
4	Enables neutron beam generator
5	Closes off high pressure loop valves
6	Enables klaxon
7 - 17	Unspecified

MPGEN2

As noted already, the control of the multiplexer code is effected through a series of tables, in particular, the CLPER, the SUT, the PERT, and the SCTM tables. These tables determine which points are scanned, in what classes, and at what frequency; and the initial startup sequence of events. All information in these tables is in packed binary configurations which would cause some difficulty in supplying the information to the code. In order to alleviate this difficulty, a second code entitled MPGEN2 (Table B-I) was written using the DECSYS system.⁽⁵⁾ The purpose of this code is to accept information in an easy format from the operator and prepare a binary "read-in" tape to be used in conjunction with the CSEMPLEX code.

Examples of the computer typed output and the operator input for MPGEN2 are shown in Table B-II. The amount of computer output can be controlled by setting or clearing the accumulator switches, numbers 1, 2, and 3. Results of the positioning of these switches are shown in Table B-II. As noted, the number of periods desired is requested first. This can be any value up through 6. Then the length of the period is requested. In the unique Period 0, scan Class 1 is the only one used. The length of Period 1 is then requested and the scanning of intervals

for each of the seven Classes requested thereafter. Any Class which is not bypassed by typing a minus 1 then has the points to be scanned in this Class requested. The end of the number of points in a given scan Class is designated by typing a minus 1. This process is repeated for each class until all Classes which have been assigned a positive scanning interval have been covered. Then, the same information is requested for Period 2 and successive periods.

After all information for the desired number of periods has been entered, the startup table commands, expressed as six octal digits, are then requested. Five of the startup commands as shown in Table V are entered automatically by the MPGEN2 routine. These start a scanner operation and initiate a real time clock to interrupt at 1 msec intervals. For the example in Table B-II, the table of startup commands shown in Table V has been used. Finally the delays for the high speed multiplexer are entered.

OPERATIONAL DETAILS

Previous to use of the CSEMPLEX code, the operation of the multiplexer should be verified using the multiplexer test code which is listed in Table B-III. Using this code in the manner described by the comments at the beginning of the listing, the gains of the four amplifiers can be adjusted when a reference voltage standard is used. The code takes 128 samples of the analogue-to-digital converter readings from the point numbers specified upon input. The interval between successive readings of the converter is adjusted by means of the accumulator switches. The clock switch (labeled EXTEND) is set to the 5KHZ position (OFF), and the octal value of the interval desired in milliseconds is entered.

After completion of the low-speed multiplexer checkout, the startup table and the scanner control tables are prepared using MPGEN2. MPGEN2 is called in from the "DECTAPE Master Reader" following the procedure given in Reference 6. The computer will halt after read-in at location 100. When CONTINUE is pressed, the computer will type out

according to the setting of the accumulator switches as is shown in example in Table B-II. At this point, the operator merely follows the typed requests that are typed out by the MPGEN2.

After the message "REMOVE TAPE" is typed, the operator then takes the punched paper tape from the high speed punch. At this time, the "DECTAPE Master Reader" is removed from the DECTAPE unit, and scratch tapes are mounted onto all of the DECTAPE units to be used for data storage from the experimental run. The units are dialed to 1, 2, 3, and 4, respectively, and the write-switch should be set to the WRITE position. CSEMPLEX is then loaded using a "RIM loader" and the "funny format" tape which has been obtained from the assembling of the code listed in Table B-IV. When the load is complete, the computer will halt with the accumulator displaying maximum memory address that was used by CSEMPLEX.

At this point the "EXTEND" switch is set to the 5KHz position, the address switches are set to 20, and START is pressed to initiate operation. Normally, all four of the tape units should be rewound to home position in the order 2, 3, 4, 1. If this does not occur, and if a tight computer loop is noted at about location 2756_8 , one of the tape units is probably off. If a rocking motion is noted on one of the tape units, this tape unit has a disabled WRITE. Normally, the computer will then proceed to the executive control loop at about 542_8 .

In order to load the tables generated by MPGEN2, the character T is typed on the teletype keyboard, which produces a message requesting that the read-in tape be mounted in the optical reader, and that CONTINUE be pressed, after which the computer halts at location 2126_8 . In order for the message to be typed out, the accumulator switches must not all be in the cleared position. A double safety feature has been incorporated in this code to control use of typed input commands. That is, both the typed commands and a non-cleared condition of the accumulator are required to execute the command. After the message is typed, the paper tape which resulted from MPGEN2 is mounted and the CONTINUE is pressed.

To begin the sequence of operations which have been specified in the startup table, an F command is now typed with the accumulator switches not zero. When this is done, the time of day and date should be noted for future reference, especially if it is anticipated that the code will be stopped and restarted at a later time.

As the run proceeds, the DECTAPES will be filled up with data in the sequence 1, 2, 3, 4, 1, etc. Thus, when tape unit 1 is full, a message will be typed to remove tape 1. At this point, the operator should remove the tape and replace it with a new fresh tape, labeling the removed tape with some adequate identification. Again, the tape should be mounted with the reading head somewhere near the beginning of the tape and the write switch enabled. If this is not done when the multiplexer code finds it necessary to record data on tape unit 1 again, a rocking motion will ensue, and the message TAPE 1 UNABLE will be typed. When the last period has elapsed, the message RUN COMPLETED will be typed. At this point, all tapes should be removed in order and labeled appropriately.

If it is desired to limit the tape units used to numbers 1 and 2 only, the octal number 010000 is deposited into MGSW at memory location 1240₈.

HALTING AND RESTARTING

Occasionally, it might be desirable to interrupt the data collection being performed by CSEMPLEX in order to use the computer for another application. To do this, the accumulator switches are set to 1 and an H is typed; a message RUN COMPLETED is typed, and immediately writing occurs on one of the tape units. All tapes written up to this time should be then removed and labeled. Also, the time of day and date should be noted. At this time, the computer is ready to be used for another application.

To restart, CSEMPLEX is loaded with four fresh tapes mounted on the four units. The scanning and startup tables are then read-in by means of the T command and by pressing CONTINUE. The time at which it is desired to begin data logging again is noted, and from this time, values of

CLKL and CLKM are computed as is described in Appendix A. The value for CLKL is stored using an L command from the value entered into the accumulators switches, and then the value of CLKM is stored into memory using the M command. Finally, operation is then restarted at the time of day which has been used in the above calculation by typing an X command with the accumulator switches not cleared. Data logging then proceeds as before with a gap present in the data due to the interruption.

ERROR MESSAGES

Several error messages have been incorporated into CSEMPLEX. First, when the typewriter types, for example, MTEO1453, this is interpreted as a magnetic tape error occurring on unit 1 at block 453 as has been described above. This block is then bypassed and contains no useful data. Information which declares that this block is not valid must be supplied to any later codes which are used to process the data resulting from CSEMPLEX. TAPE 01 UNABLE means that the WRITE switch on Unit 1 is in the WRITE LOCK position, and it should be placed into the WRITE position. SCAN FAIL indicates that some malfunction of the low speed multiplexer has occurred, and it has taken too long for the analogue-to-digital conversion to take place. ILLEGAL INTERRUPT indicates that some pulse has entered into the interrupt facility which is illegal. Presumably, this could come from one of the multichannel analyzers connected to the computer. IPE is typed whenever an illegal character is typed into the keyboard for control purposes. A message which should never occur, and which indicates severe and irrecoverable trouble, is STORAGE QUEUE EMPTY. This can only result if one has had extreme trouble with the magnetic tapes, or if one of the magnetic tape units has been UNABLE for an excessive length of time.

DETAILED DESCRIPTION OF CSEMPLEX

The CSEMPLEX code has been broken down by logical function into segments numbered 01 through 19.

CSEEMPLX 01

This is the teletype interrupt service routine. When the interrupt service program determines that the teletype flag is up, it turns control over to location TELTYP. At this point, a character is removed from the teletype buffer TELBUF and is typed on the teletype. If the buffer is empty, return is made directly to the function program.

CSEEMPLX 02

This is the storage request and return subroutines which must be entered in an inhibited mode. STOREQ supplies a block of 400₈ words to the requesting function program. When that use of that storage is complete by the function program, it is returned to the storage queue by means of the STORAD subroutine. The queue is initially set to twelve entries by the initialization program to allow room for automatic storage by hardware of the data coming from the high speed multiplexer. When the queue is found to be empty, upon a storage request, control is turned over to the STREMP routine wherein the message STORAGE QUEUE EMPTY is caused to be typed.

CSEEMPLX 03

These are the magnetic tape output queue handler subroutines which are entered from a function program in the inhibited note. When data is ready to be written onto DECTAPE, the MGQAD routine is entered by the scan program and the initial address of the block is added to the magnetic tape queue, MQUEUE. The magnetic tape function program and driver routines use the MGQREQ routine to obtain the next block of memory to be written out onto DECTAPE from this queue.

CSEEMPLX 04

This routine is the interrupt handler. It consists of a series of skip instructions which test the various flags that are present in the interrupt vector. If an interrupt occurs and no flag is set, this routine causes the message ILLEGAL INTERRUPT to be typed. The CONTIN routine restores the accumulator and returns control to the function program. It is used by all of the interrupt service routines upon their exit.

CSEMPLEX 05

This is the teletype output queue handler. When any of the function programs or interrupt driven programs need a message typed, they enter TYPOUT with the initial address of the message. This initial address is then entered into TQUEUE and the IOP function program No. 2 is turned on simultaneously.

CSEMPLEX 06

These routines consist of the EXIT, WAIT, and SARR (save all registers) subroutines used by the function programs to terminate operation or to allow delays. SARR is used by the interrupt driven routines, which must either use the registers normally assigned to the function programs (location 10) or that require control be turned over to another function program. The WAIT subroutine causes the return address to the function program to be decremented by 2. This allows the automatic return to the function program at the point where the test on the waiting condition is made.

CSEMPLEX 07

This is the executive control program number 0 and consists of the idling loop EXEC, wherein the PROG entry for each of the programs is tested. If a program is turned on, indicated by PROG for this program being equal to 1, control is turned over to RARR wherein the index registers are restored and control is turned over to the function program. The PROG array consists of the storage for the accumulator, link, return address, initial entry address, and the auto-index register 10 for each of the function programs.

CSEMPLEX 08

This is the input-output program number 3 which is used to decode the messages in the teletype queue, TQUEUE. The messages are stored three characters per computer word using the Anelex code.⁽²⁾ The end of the message is indicated by a zero character (octal 00). The IOP routine

separates out a character, then using the conversion table BTAT, converts this to its ASCII equivalent and stores this into the teletype queue, TELBUF. When the zero character is encountered, indicating the end of message, a line feed and carriage return are stored in TELBUF, and the queue, TQUEUE, is examined for further messages.

CSEMPLEX 09

This is the magnetic tape output function program number 2 and its associated driver. When the entry is made into the magnetic tape output queue handler, MGQAD, the magtape function program is turned on. The program begins by requesting the memory block to be written through MGQREQ and exits if the queue is empty. It then searches for the block upon which the data is to be written, and when found, proceeds to write the data following a normal system of writing to DECTAPE.

The error flag can be raised by several causes. First, the tape might be in the end of tape area, requiring removal and switching to the next greater numbered unit(1, 2, 3, 4, 1, ...). If this occurs, the switching is automatic, and the message REPLACE TAPE is typed. A second error flag might occur if the tape is in beginning of tape area, at which position the tape motion must be reversed and the unit put into the search-forward mode. A third reason for error could be that the tape WRITE switch was in the WRITE LOCK position. If this is the case the TAPE UNABLE message is typed and the writing action is tried again. This results in the rocking motion that has been described earlier. If none of the above is true, then a true error must have occurred; the message MTE with the associated unit and block number is typed out and the data is written on the next higher numbered block. Tape units 1 and 2 only are used if MGSW = 010000.

CSEMPLEX 10

This is the scanner control function program number 1 which is used to control the operation of the low speed multiplexer. Upon entry, buffer storage is requested. Then, the end of the Period is checked,

and if the Period is complete, suitable changes are made in the CLPER table and the PERT table to set up for operation in the next Period.

After the end of Period check, the times to scan the various scan Classes are checked to see if it is time to operate any one of the individual Classes. If the scan Class time is negative, it is assumed that there are no points to be scanned in this Class and it is bypassed. If it is time to scan this Class the next point in order is scanned. During the first Period, only one Class is being scanned and all points connected to the low speed multiplexer are scanned. During later periods, only those points which are so indicated in the CLPER table are scanned in the Class designated. Information as to what scan Class is being operated, the Period number, and the sequence number showing that this class has been scanned in this Period, are stored in the buffer array in a manner that is indicated in Table III. At this time, the multiplexer is enabled, a diagnostic countdown counter is set, and the scan program waits until the conversion of the four subpoints is complete. When this is done, the buffer is checked to insure that there is still room, and if not, action is taken to write out the buffer to magnetic tape.

At the end of the last Period, control is turned over to location TERM wherein the rest of the block of data currently being stored is filled with 777777₈, if this is the true end of the run, or it is filled with 777000 if this is an intermediate interruption. Then, the block of data is written out to DECTAPE and the message typed indicating that this is the end of the run.

CSEMPLEX 11

This is the scan driver program which is entered upon completion of an analogue-to-digital conversion from the interrupt service vector. Here, the analogue data are combined with the information already stored by the scan program in the data buffer, the next subpoint is selected, and conversion is initiated. Upon completion of the last conversion the diagnostic counter is reset, the flag SDCTR is set to zero to indicate to the scan program that the conversion of all the subpoints is complete.

CSEEMPLX 12

This is the real time clock interrupt service routine. Initially, it restarts the clock so as to interrupt 1 millisecond later. Then, the diagnostic counter for the scanner is examined to see if a failure has occurred. If the low speed multiplexer has not completed its conversion within 2 msec, the multiplexer is cycled around and the conversion tried again. The message SCANNER FAILED is caused to be typed, also.

Right after the "time 0" interrupt occurs the location CLOKSW is set to a NOP by CSEEMPLX 14. This causes the counters for the high speed multiplexer delays to be examined. When a delay is completed, the associated multiplexer is enabled. When all four multiplexer delays are completed, the CLOKSW is reset to the normal JMP CLOK2.

CSEEMPLX 13

This is the startup program previously described, which serves two functions. It is used, first, to read-in the startup tables SUT, CLPER, PERT, and SCTM, and second, to process the commands stored in the table SUT during the startup sequence of the experimental run.

CSEEMPLX 14

These routines handle the incoming data from the high speed multiplexers. First, when the "time zero" interrupt occurs, the four high speed multiplexers' delay periods are begun. The program then waits until the high speed multiplexers successively complete their delays. The after data collection is written out on magnetic tape.

CSEEMPLX 15

This routine is used to service the interrupts occurring from the typewriter keyboard; that is, to process commands that are entered through this keyboard. If an "R" is typed, the relay buffer is reset to the ACCUMULATOR switch value. If a "T" is typed, the reading of the tables through the startup routine is effected. If an "H" is typed, the

termination procedure for interrupting the data logging is followed. If an "F" is typed, the sequence of instructions in the startup table are processed. If an "L" (or "M") is typed, a new value for the least (or most) significant half of the real time clock is set to the value in the ACCUMULATOR switches. If an "X" is typed, it is assumed that this is a restart of a previous run, and then the time zero flip-flop is locked out and the storage originally set aside for the high speed multiplexer is returned to the storage queue. The scanner is started by turning on the program number 1, the scanner program.

CSEEMPLX 16

This is a subroutine used whenever a magnetic tape error occurs in order to convert the three digit octal block number to a code similar to that used in an alpha-numeric message.

CSEEMPLX 19

This is the initialization routine, and it is used to restore all segments of the code to the values at which they were before operation. However, it does not undo the CLPER table which has been modified by the scan program (No. 1). Therefore, if during debug operation, it is desired to restart the program for test purposes, it will be necessary to read-in again the tables which had been prepared by the MPGEN2 code.

REFERENCES

1. G. J. Rogers. Program for Containment Systems Experiment, General Electric Company, Richland, Washington, HW-83607. September, 1964.
2. PDP-7, USERS HANDBOOK, pamphlet F-75. Digital Equipment Corporation, Maynard, Massachusetts, June, 1965.
3. R. A. Walker. Modifications to the CSE PDP-7 Computer, BNWL-CC-713. Pacific Northwest Laboratory, Richland, Washington, July, 1966.
4. James Martin. Programming Real-Time Computer Systems, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1965.
5. PDP-7 DECSYS Operating Manual, pamphlet 7-5-S, Digital Equipment Corporation, Maynard, Massachusetts, March, 1966.
6. W. D. Draper. Interim Report on Utility Programs for the FFTF Instrumentation Prototype, BNWL-CC-659. Pacific Northwest Laboratory, Richland, Washington, June, 1966.

APPENDIX A

ENTRY OF CLKM AND CLKL UPON RESTART

APPENDIX A

ENTRY OF CLKM AND CLKL UPON RESTART

Assume that the following sequence of events has occurred:

1. The procedure described in Section IV of this report was followed and the "F" was typed at 10:00:00.000 a.m. on a given day.
2. At 14:23 the data logging was interrupted by typing an "H", the DECTAPES removed, and the computer turned over to another application.
3. At 15:10 it was desired to commence data logging again. Previous to this time CSEMPLEX was loaded into core, fresh DECTAPES mounted, a "T" was typed and the tables generated by MPGEN read-in. It is now necessary to compute the octal numbers to be inserted into CLKM and CLKL by means of the typed "M" and "L" commands respectively. Since CLKM and CLKL consist of a 36 bit (or 12 octal digit) internal clock which measures millisec, it is first necessary to convert the elapsed time since CLKM=CLKL=0 at 10.--a.m. to millisec, then to its octal equivalent.

15:10 minus 10.-- = 5 hours 10 mins = 18,600,000 millisec.

The conversion of this number to octal is most easily effected by considering CLKM and CLKL to be the integral and fractional parts respectively of some number X where:

$$X = \frac{18600000}{262144} = 70.953369$$

Then, using the tables on pages 181 through 187 of Reference (2)

$$\text{CLKM} = 70_{10} = 000106_8$$

$$\begin{array}{rcl} \text{CLKL} & = 0.5_{10} & = 0.4_8 \\ & + 0.453125_{10} & + 0.350_8 \\ & + 0.000244_{10} & + 0.000100_8 \\ \hline & = 0.953369_{10} & = 0.750100_8 \end{array}$$

Thus, [000106] is entered into the accumulator switches and "M" typed; [750100] entered and "L" typed; and at 15:10,[000001] entered and "X" typed.

BNWL-597

APPENDIX B

PROGRAM LISTING

Table B-I. Listing of MPGEN2 Code

Table B-I. Listing of MPGEN2 Code

```
MPLXGEN
/PROGRAM TO GENERATE TABLES FOR CSEMPLEX
FIDDEC
EXTERNAL .ID1A,.ID2,VARI1H
HLT
JMP .AAA

PERT.,    JMS CALSB
          LAW 2
          17661

SCTM.,    JMS CALSB
          LAW 7
          LAW 1
          17677

CLPER.,   JMS CALSB
          LAW 1
          17700

SUT.,     JMS CALSB
          LAW 1
          17500

MSKP.,    JMS CALSB
          LAW 1
          .MSKP
```

```

.AAF,      LAC ICL.
           TITLE3

           READ      ZGET
           DAS .IDIA
           1         ZSCN
           ROR .52
           ARF X1.   ZINTERVAL
           ENDD
           LAC ACS   ZIS INTERVAL

           SZ4       ZNEGATIVE
           JMP .AAFT ZYES
           CNVISP    ZCONVERT IT
.AAFE,     ERROR    ZERROR
           JMP .AAF  ZTRY AGAIN

           CAL SCIM. ZGET
           ARX IPERI. ZSCIM
           ARX ICL.  ZADDR

           LAC X1.   ZSTORE
           DAC I TEMAD ZSCIM
           SNA       ZINTERVAL ZERO
           JMP .AAFE ZYES ERROR

           LAC IPER. ZIS
           SNA       ZTHIS PERIOD
           JMP .AAL   ZZERO

.AAFT,     ISZ ICL.  ZARE WE
           ISZ ICLM. ZTHRU
           JMP .AAF   ZND

.AAB,      LAC X1
           DAC ICL.

           LAM -6
           DAC ICLM.

           ITL06
           ITL09

           DZM SUM.
           DZM SUM. 1 ZSUM=4.

```

```

.AAH,      CAL SCTM.      /GET
           ARX IPER1.     /SCTM
           ARX ICL.       /ADDR

           LAC I TEMAD    /IS
           SPA            /IT NEG
           JMP .AAK       /YES - CLASS BYPASSED

           LAC I TEMAD
           FLD            /TI=2./FLOATF(SCTM(IPER,ICL))
           DAC TI.
           LAC TWO
           FDV TI.
           DAC TI.
           LFM
           LAC ICL.
           TIIL08

.AAI,      LAC ICL.
           DAC ICL1.
           RDD            /GET OCIAL POINT NO.
           SPA            /END OF INPUT
           JMP .AAK       /YES

           DAC I0.
           CMA            /IS
           TAD C1         /PT
           TAD C57        />57
           SMA
           JMP .+3        /NO
           ERROR          /YES
           JMP .AAI

           ISZ I0.

```

```

CAL CLPER.  /GET CLPER
ARX I4.     /ADDR

LAC IEMAD   /SAVE
DAC I1.     /II

CAL MSKP.   /GET
ARX IPER.

LAC I IEMAD /MASK
AND I I1.   /DELETE BITS
DAC I I1.   /FROM CLPER

LAM -0      /IS IT
TAD IPER.   /PERIOD 1
SNA
JMP .AAJ    /YES - BYPASS SHIFT

LAC IPER.
CMA
TAD C2
DAC IT.

LAC ICL1.   /POSITION
RCL
RTL         /ICL
ISZ IT.
JMP .-3
DAC ICL1.

.AAJ,       LAC ICL1.   /STORE ICL
            XOR I I1.   /IN
            DAC I I1.   /CLPER

EFM
LAC SUM.
ADD II.
DAC SUM.
FSB ONE
CAS
JMP .AAJ0
JMP .AAJ0
LFM          /SUM>1.0
WRITE
JMS .ID2
2
FOR .53
ARX IPER.
ENDID
ERROR
JMP .AAE

```

```

.53,      320007      /C7H
          TEXT
MULTIPLEXOR SCANNING CAPACITY EXCEEDED FOR PERIOD
          020003      /,13
          600000      /)

.AAM,      LFM
          JMP .AAI
.AAK,      ISZ ICL.
          ISZ ICLM.
          JMP .AAH

.AAL,      TITL06

          ISZ IPER.
          ISZ IPER1.
          ISZ IPERM.
          JMP .AAE

/
/ROUTINE TO BUILD SUT

.AAM,      TITL11
          LAW SUTE-1
          DAC I0
          LAW-10577+SUTE
          DAC I1.

.AAN,      JMS R060      /GET WORD
          SAD C707370 /TERMINATION
          JMP .AAO      /YES
          DAC I 10      /STORE WORD
          ISZ I1.      /END OF TABLE
          JMP .AAN      /NO
          ERROR      /YES
          JMP .AAM      /TRY AGAIN

.AAO,      TITL12      /HISPD DELAY INPUT
          LAW 10757      /INIT. HISPD DELAY ADDR. -1
          DAC I0      /=3757 IN CSEMPLEX
          LAW -3
          DAC I1.
          DZM I0.

.AAP,      LAC I0.      /TYPE "DELAY"
          TITL13
          READ      /GET
          JMS .I010
          I      /DELAY
          FOR .52
          ARE X1.
          ENDDO      /TIME

```

```

LAC ACS      /IS II
SZA          /NEGATIVE
JMP .AAPE    /YES-ERROR
CNVTSP       /CONVERT II

```

```

.AAPE,      ERROR
JMP .AAP
DAC I 10     /STORE DELAY TIME IN MSEC
ISZ I0.
ISZ I1.
JMP .AAP

```

```

.AAQ,

```

```

/SCAN CLPER TO FIND FIRST AND LAST PIS.

```

```

LAC 03676    /=CLPER-2 IN CSEMPLEX
DAC 10764    /INITIAL ICLPI=CLPER-2
IAD 02       /
DAC I0.      /INITIAL ICLPF=CLPER

```

```

LAM-60       /SET UP LOOP CNTRS.
DAC I.
LAC CLPER.+2
IAD (-1)
DAC I0

```

```

.AAR,      ISZ I.      /INCR. CNTR.
            ISZ 10764   /INCR. ICLPI
            ISZ I0.     /INCR. ICLPF
            LAC I 10     /GET CLPER WORD
            SAD 00       /IF .NE. ZERO, ICLPI FOUND
            JMP .AAR

```

```

.AAS,      LAC I0.      /SET ICLPF
            DAC 10765
            ISZ I0.
            LAC I 10
            ISZ I.       /ARE WE THRU
            SKP          /NO
            JMP .AAT     /YES
            SAD 00
            JMP .AAS + 2
            JMP .AAS

```

```

.AAT,

```


/PUNCH QUI AND TERMINATION

```

      LAM -400      /PUNCH
      DAC I.        /LEADER
      CLA
      PSA
      PSF
      JMP .-1
      ISZ I.
      JMP .ABA
      LAM -277
      DAC I1.

      CAL SUB.
      ARX C1

      .ABB,        LAM -2
                  DAC I0.

      LAC I TEMAD /GET BINARY WORD
      RCL          /1 POS. LEFT

      .ABC,        RTL          /PUNCH IT
                  RTL
                  RTL
                  PSB
                  PSF
                  JMP .-1
                  ISZ I0.
                  JMP .ABC      /WORD

      ISZ TEMAD
      ISZ I1.
      JMP .ABB
      LAM -400      /PUNCH
      DAC I.        /LEADER
      LAC C274      /PUT
      JMS .ABE      /HLT
      LAC C200      /IN
      JMS .ABE      /READ
      LAC C340      /IN
      JMS .ABE      /TAPE

      .ABD,        CLA          /PUNCH
                  JMS .ABE      /LEADER
                  ISZ I.
                  JMP .ABD

      TITL10

      HLT          /END OF
      JMP .-1      /PROGRAM

```

```
.ABE,      0
           PSA
           PSF
           JMP  --1
           JMP I .ABE
.ACA,      CAL PERT.
           ARX IPER.    /PERT (IPER+1)
           LAC TEMAD
           DAC IT.

           CAL PERT.
           ARX IPER1.
           LAC I IT.
           DAC I TEMAD
           ISZ IT.
           ISZ TEMAD
           LAC I IT.
           DAC I TEMAD /=PERT(IPER)

           JMP .AAL
```

/SUBROUTINES TO TYPE OUT TITLES
EXIT=JMP 1 -JMS

TIT01, 0
 SW3
 EXIT TITL01
 WRITE
 JMS .102
 2
 FOR .1
 ENDIO
 EXIT TITL01

.1, 320007 /(7H
 TEXT
PROGRAM TO PUNCH READ IN TAPE TO CONTROL
 320007 /(7H
 TEXT
MULTIPLEXERS (SA=3500)
 400000
 520007 //7H
 TEXT
LOW SPEED MULTIPLEXER SPECIFICATIONS
 400000 //
 600000 /)

TIT02, 0
 SW1
 EXIT TITL02
 WRITE
 JMS .102
 2
 FOR .2
 ENDIO
 EXIT TITL02

.2, 320007 /(7H
 TEXT
NO. OF PERIODS DESIRED
 600000 /16L

```

TIT03,      0
            DAC IT.
            SW2
            EXIT TITL03
            WRITE
            JMS .I02
            2
            FOR .3
            ARX IT.
            ENDDO
            EXIT TITL03

.3,          320007      /C7H
            TEXT
*****PERIOD
            020002      /I2
            320007      /,7H
            TEXT
*****
            600000      /)

TIT04,      0
            SW1
            EXIT TITL04
            WRITE
            JMS .I02
            2
            FOR .4
            ENDDO
            EXIT TITL04

.4,          320007      /C7H
            TEXT
LENGTH
            600000      /)

```

```

TIT05,      0
            SW1
            EXIT TITL05
            WRITE
            JMS .102
            2
            FOR .5
            ENDIO
            EXIT TITL05

.5,          320007      /C7H
            TEXT
UNITS-H,M,S,T
            600000      /)

TIT06,      0
            SW3
            EXIT TITL06
            WRITE
            JMS .102
            2
            FOR .6
            ENDIO
            EXIT TITL06

.6,          320007      /C7H
            TEXT
-----
            600000      /)

```

```

TIT07,      0
            SW2
            EXIT TITL07
            WRITE
            JMS .I02
            2
            FOR .7
            ENDIO
            EXIT TITL07

.7,          320007      /C7H
            TEXT
SCANNING INTERVALS
            600000      /)

TIT08,      0
            DAC IT.
            SW1
            EXIT TITL08
            WRITE
            JMS .I02
            2
            FOR .8
            ARX IT.
            ENDIO
            EXIT TITL08

.8,          320007      /C7H
            TEXT
CLASS
            020002      /,12
            600000

```

```

TIT09,      0
            SW2
            EXIT TITL09
            WRITE
            JMS .I02
            2
            FOR .9
            ENDIO
            EXIT TITL09

.9,          320007      /C7H
            TEXT
POINT ASSIGNMENT
            600000      /)

TIT10,      0
            SW3
            EXIT TITL10
            WRITE
            JMS .I02
            2
            FOR .10
            ENDIO
            EXIT TITL10

.10,         320007      /C7H
            TEXT
REMOVE READ IN TAPE FROM PUNCH
            600000      /)

TIT11,      0
            SW3
            EXIT TITL11
            WRITE
            JMS .I02
            2
            FOR .11
            ENDIO
            EXIT TITL11

.11,         320007      /C7H
            TEXT
TYPE IN START UP TABLE COMMANDS AS 6 OCTAL DIGITS,L/F,C/R
            400000      /
            120007      /,7H
            TEXT
TERMINATE WITH -1,L/F,C/R
            600000      /)

```

```

TIT12,      0
            SW3
            EXIT TITL12
            WRITE
            JMS .I02
            2
            FOR .12
            ENDDO
            EXIT TITL12

.12,        200000      /C
            520007      //7H
            TEXT
HIGH SPEED MULTIPLEXER SPECIFICATIONS
            520007
            TEXT
DELAYS
            600000

TIT13,      0
            DAC IT.
            SW1
            EXIT TITL13
            WRITE
            JMS .I02
            2
            FOR .13
            ARX IT.
            ENDDO
            EXIT TITL13

.13,        320007      /C7H
            TEXT
MPX
            020002      /,12
            600000

```


/ SUBROUTINES TO BYPASS PROGRAM PORTIONS
/ WHEN ACS ARE SET

SWT1, 0
 LAS
 AND (200000
 SZA
 SW1 EXIT
 ISZ SW1-JMS
 EXIT SW1

SWT2, 0
 LAS
 AND (100000
 SZA
 EXIT SW2
 ISZ SWT2
 EXIT SW2

SWT3, 0
 LAS
 AND (400000
 SZA
 EXIT SW3
 ISZ SWT3
 EXIT SW3

/SUBROUTINE ENTERED ON INPUT ERROR

```
ERR,      0
          WRITE
          JMS .102
          2
          FOR .51
          ENDIO
          EXIT ERROR
.51,      320007      /C7H
          TEXT
--- TRY AGAIN
          600000      /)
```

/COMMON TIME CONVERSION ROUTINE TO
/CONVERT INPUT TO FLT. PT. MSEC (X1)

```
CNVT,      0
          TITL05
CNVT0,      JMS RDA
          DAC IT.      /GET CHAR.
          JMS RDA      /LF
          JMS RDA      /CR
          LAC IT.
          SAD (310      /H
          JMP CNVTH
          SAD (315      /M
          JMP CNVTM
          SAD (323      /S
          JMP CNVTSC
          SAD (324      /T
          JMP CNVTT
          ERROR      /TYPEIN
          JMP CNVT0      /ERROR
CNVTH,      EFM
          LAC X1.
          FMP SIXTY
          DAC X1.
CNVTM,      EFM
          LAC X1.
          FMP SIXTY
          DAC X1.
CNVTSC,      EFM
          LAC X1.
          FMP THOU
          DAC X1.
CNVTT,      LFM
          JMP I CNVT
RDA,      0
          KSF
          JMP .-1
          KRB
          JMP I RDA
```

/CNVT FLT. PT. X1. TO SING. PREC. INTEGER

```

CNVTS,      0
            JMS CNVT      /X1 TO MSEC
            EFM
            LAC TWO17     /IS X1
            FSB X1.       />2**17
            CAS
            JMP CNVTSE    /- = YES
            JMP CNVTSE    /0 = YES
            LAC X1.       /+ = NO
            FXA
            DAC X1.
            ISZ CNVTS
            ISZ CNVTS
            EXIT CNVTSP

```

```

CNVTSE,     LFM
            EXIT CNVTSP

```

/CONVERT FLT. PT. X1. TO D.P. INTEGER
/STORED IN IT AND IT0

```

CNVTD,      0
            EFM
            LAC X1.
            FDV TWO18
            FXA
            DAC IT.
            FLO
            FMP TWO18
            DAC X2.
            LAC X1.
            FSB X2.
            FDV TWO
            FXA
            RCL
            DAC IT0.
            EXIT CNVTD

```

/VARIABLES

X1.,	0	0
X2.,	0	0
X3.,	0	0
SUM.,	0	0
IT.,	0	0
I.,	0	
IT0.,	0	
IT1.,	0	
IT.,	0	
IT0.,	0	
NPER.,	0	
IPER.,	0	
IPER1.,	0	
IPERM.,	0	
ICL.,	0	
ICLM.,	0	
ICL1.,	0	

```

ONE,      001400      /1.0
0
TWO,      002400      /2.0
0
/FLOATING POINT CONSTANTS
TWO17,    022400      /2**17
0
SIXTY,    006740      /60.
0
THOU,     012764      /1000.
0
TWO18,    023400      /2**18
0

/MASKS
.MSKP,     777770      777707      777077
          770777      707777      077777

/FORMATS
.52,       260017      /(E15.0
          600000      /)
TITL01 = JMS TIT01
TITL02 = JMS TIT02
TITL03 = JMS TIT03
TITL04 = JMS TIT04
TITL05 = JMS TIT05
TITL06 = JMS TIT06
TITL07 = JMS TIT07
TITL08 = JMS TIT08
TITL09 = JMS TIT09
TITL10 = JMS TIT10
TITL11=JMS TIT11
TITL12=JMS TIT12
TITL13=JMS TIT13
SW1 = JMS SWT1
SW2 = JMS SWT2
SW3 = JMS SWT3
RDI = JMS READI
RDO = JMS READO
ERROR = JMS ERR
CNVTSP=JMS CNVTS
CNVIDP=JMS CNVID
10500/
          5          /SUI
          LAM - 4     /SET CLOCK INTERRUPT
          2          /TO 1 MSEC
          20022       /START SCAN
          1
SUTE,      0

10577/     0
.52+2/
START

```

Table B-II. Typical Teletype Input-Output When Using MPGEN2

- A. Accumulator Switches Cleared
- B. Switch 1 Set
- C. Switches 1, 2, Set
- D. Switches 1, 2, 3 Set

TABLE B-II. Typical Teletype Input-Output When Using MPGEN2

A. Accumulator Switches Cleared

PROGRAM TO PUNCH READ IN TAPE TO CONTROL MULTIPLEXERS (SA=3500)

LOW SPEED MULTIPLEXER SPECIFICATIONS

NO. OF PERIODS DESIRED

2

*****PERIOD 0*****

LENGTH

899900

UNITS-H,M,S,T

T

SCANNING INTERVALS

CLASS 1

2

UNITS-H,M,S,T

M

*****PERIOD 1*****

LENGTH

1100

UNITS-H,M,S,T

T

SCANNING INTERVALS

CLASS 1

50

UNITS-H,M,S,T

T

CLASS 2

-1

CLASS 3

-1

CLASS 4

-1

CLASS 5

-1

CLASS 6

-1

CLASS 7

-1

POINT ASSIGNMENT

CLASS 1

021

035

-1

*****PERIOD 2*****

LENGTH

5

UNITS-H,M,S,T

S

SCANNING INTERVALS

CLASS 1

100

UNITS-H,M,S,T

T

CLASS 2

500

UNITS-H,M,S,T

T

CLASS 3

-1

CLASS 4

-1

CLASS 5

-1

CLASS 6

-1

CLASS 7

-1

POINT ASSIGNMENT

CLASS 1

021

-1

CLASS 2

035

-1

TYPE IN START UP TABLE COMMANDS AS 6 OCTAL DIGITS,L/F,C/R
TERMINATE WITH -1,L/F,C/R

000000

000002

223700

000001

004000

000000

000000

724600

000011

034000

000000
000000
164154
000001
374000
000000
000000
000764
000003
200355
040354
740000
000001
774000
000000
000000
003720
000001
030000
000004
- 1

HIGH SPEED MULTIPLEXER SPECIFICATIONS
DELAYS

MPX 0

0

UNITS-H,M,S,T

T

MPX 1

100

UNITS-H,M,S,T

T

MPX 2

200

UNITS-H,M,S,T

T

MPX 3

300

UNITS-H,M,S,T

T

REMOVE READ IN TAPE FROM PUNCH

B. Switch 1 Set

PROGRAM TO PUNCH READ IN TAPE TO CONTROL MULTIPLEXERS (SA=3500)

LOW SPEED MULTIPLEXER SPECIFICATIONS

2

*****PERIOD 0*****

899900

I

SCANNING INTERVALS

2

M

*****PERIOD 1*****

1100

I

SCANNING INTERVALS

5

I

-1

-1

-1

-1

-1

-1

POINT ASSIGNMENT

021

035

-1

*****PERIOD 2*****

5

S

SCANNING INTERVALS

100

I

500

I

-1

-1

-1

-1

-1

POINT ASSIGNMENT

021

-1

035

-1

TYPE IN START UP TABLE COMMANDS AS 6 OCTAL DIGITS, L/F, C/R
 TERMINATE WITH -1, L/F, C/R

000000

000002

023700

000001

004000

000000

000000

724600

000001

034000

000000

000000

000764

000003

000355

740000

740000

000001

774000

000000

000000

000000

003720

000001

030000

000004

-1

HIGH SPEED MULTIPLEXER SPECIFICATIONS
 DELAYS

M

I

100

I

200

I

300

I

REMOVE READ IN TAPE FROM PUNCH

C. Switch 1, 2 Set

PROGRAM TO PUNCH READ IN TAPE TO CONTROL MULTIPLEXERS (SA=3500)

LOW SPEED MULTIPLEXER SPECIFICATIONS

2

899900

T

2

M

1100

T

50

T

-1

-1

-1

-1

-1

-1

021

035

-1

5

S

100

T

500

T

-1

-1

-1

-1

-1

021

-1

035

-1

TYPE IN START UP TABLE COMMANDS AS 6 OCTAL DIGITS,L/F,C/R
 TERMINATE WITH -1,L/F,C/R

000000
 000002
 223700
 000001
 004000
 000000
 000000
 724600
 000001
 034000
 000000
 000000
 000764
 000003
 200355
 040354
 740000
 000001
 774000
 000000
 000000
 003720
 000000
 030000
 000004

- 1

HIGH SPEED MULTIPLEXER SPECIFICATIONS

DELAYS

0

T

100

T

200

T

300

T

REMOVE READ IN TAPE FROM PUNCH

B-30

BNWL-597

D. Switch 1, 2, 3 Set

2
899900
T
2
M
1100
T
50
T
-1
-1
-1
-1
-1
-1
021
035
-1
5
S
100
T
500
T
-1
-1
-1
-1
-1
021
-1
035
-1
000000
000002
223700
000001
004000
000000
000000
724600
000001
034000
000000
000000
164154
000001
374000

000000
000000
000764
000003
200355
040354
740000
000001
774000
000000
000000
003720
000001
030000
000004
- 1
0T
T
100
T
200
T
300
T

Table B-III. Listing of Multiplexer Test Code

Table B-III. Listing of Multiplexer Test Code

```

MULTIPLEXER TEST LO-DIFF-HI-VAR-AVG
/SET ADS TO 22
/PUSH START
/SET REAL TIME CLOCK TO SKC POSITION
/SET ACS TO OCTAL VALUE IN MSEC DESIRED BETWEEN SCANS
/TYPE STARTING MULTIPLEXER ADDRESS (00-57)
/TYPE FINAL MULTIPLEXER ADDRESS (00-57)--MUST BE > STARTING
/
/PROGRAM WILL TYPE OUT
/"N" ADDR (0) LO  DIFF  HI  VAR  AVG
/          (1) LO  DIFF  HI  VAR  AVG
/          (2) LO  DIFF  HI  VAR  AVG
/          (3) LO  DIFF  HI  VAR  AVG
/"N+1" ADDR ETC
/
/ALL VALUES TYPED OUT WILL BE IN OCTAL WITH THE 12 BITS
/FROM THE A/D CONVERTER RIGHT ADJUSTED
      LAW 17774
      DAC FORCOU
      DZM SCANUM
      DZM TEST
      LAC (SA
      DAC ADR
      KRB
      LAW 215
      TLS
KB,   JMS CR
      JMS LF
      LAC CORE
      TAD (1
      DAC VAR1
      DAC VAR2
      KSF
      JMP --1
      KRB
      AND BITS
      CLL
      RTL
      RAL
      DAC I ADR
      KSF
      JMP --1
      KRB
      AND BITS
      XOR I ADR
      RTL
      RAL
      DAC I ADR
      KSF
      JMP --1

```

```

KRB
AND BITS
XOR I ADR
DAC I ADR
LAC TEST
SZA
JMP .+5
ISZ TEST
LAC (FA
DAC ADR
JMP KB
DZM TEST
LAC (SA
DAC ADR
JMS CR
JMS LF
JMS TTEN
JMP MPX
777600
0
LAC (11
TAD SA
CMA
ADD FA
SPA
JMP .+4
LAC (11
TAD SA
DAC FA
LAC FA
JMP I TTEN
0
701301
JMP .-1
701312
RAL
CML
RAR
CLL
AND MASK1
RTR
RTR
RTR
DAC TEMP1
DAC I VAR2
LAC SCANUM
SZA
JMP .+5
LAC TEMP1
DAC I CURLO

```

MASK1,
TTEN,

READ,

DAC I CURHI
DZM I CURAVE
LAC TEMP1
CMA
DAC TEMP2
ISZ TEMP2
NOP
LAC TEMP2
TAD I CURLO
SMA
JMS STORLO
LAC TEMP2
TAD I CURHI
SPA
JMS STORHI
LAC TEMP1
CLL
RAR
TAD I CURAVE
DAC I CURAVE
LAC VAR2
TAD (200
DAC VAR2
ISZ FORCOU
JMP .+2
JMP .+6
703042
ISZ CURLO
ISZ CURAVE
ISZ CURHI
JMP READ+1
LAW 17774
DAC FORCOU
JMP I READ
LAC SA
DAC CURLOC
LAC (400
XOR CURLOC
CLL
RTL
DAC CURLO
LAC (600
XOR CURLOC
RTL
DAC CURAVE
LAC (1000
XOR CURLOC
RTL
DAC CURHI
LAC CURLOC

MPX,

```

703004
JMS READ
LAC CURLOC
SAD FA
JMP ++2
JMP ++11
ISZ SCANUM
LAC SCANUM
TAD VAR1
DAC VAR2
LAC SCANUM
SAD (200)
JMP ++4
JMP MPXT
ISZ CURLOC
JMP MPX+2
DZM SCANUM
LAC CORE
DAC 13
JMP RO
MPXT, LAS
SNA
JMP MPX
CMA
DAC LF
ISZ LF
TIME, LAM -4
DAC 7
CLON
CLSF
JMP --1
ISZ LF
JMP TIME
CLOF
RO, JMP MPX
LAC SA
DAC CURLOC
LAC (400
XOR CURLOC
CLL
RTL
TAD (777777
DAC 10
LAC (600
XOR CURLOC
CLL
RTL
TAD (777777
DAC 11
LAC (1000

```

AGAIN,

XOR	CURLOC			
CLL				
RTL				
TAD	(777777			
DAC	12			
LAC	CURLOC			
JMS	TYPE3			
JMS	SPACE			
LAC	I 10			
DAC	DIFF			
JMS	TYPE6			
JMS	SPACE			
LAC	I 11			
RCR		RCR	RCR	RCR
RCR		RCR		
DAC	SAVE1			
LAC	I 12			
DAC	HIGH			
LAC	DIFF			
CMA				
TAD	HIGH			
TAD	(1			
AND	(7777			
JMS	TYPE6			
JMS	SPACE			
LAC	HIGH			
JMS	TYPE6			
JMS	SPACE			
JMS	VAR			
JMS	SPACE			
JMS	SPACE			
LAC	VAR2			
JMS	TYPE6			
JMS	SPACE			
DZM	S1			
DZM	S2			
DZM	S3			
LAC	SAVE1			
JMS	CMY			
LAC	(12			
DAC	SAVE1			
AND	BITA			
DAC	S6			
LAC	SAVE1			
RCR		RCR	RCR	RCR
RCR		RCR		
DAC	SAVE1			
LAC	S6			
JMS	CMY			
LAC	(20			

```
ADD S3
DAC S3
ADD (-11
SMA
JMS CARRY1
LAC S2
ADD (-11
SMA
JMS CARRY2
LAC S1
DAC SAVE2
JMS PRNT1
LAC S2
DAC SAVE2
JMS PRNT1
LAC S3
DAC SAVE2
JMS PRNT1
ENDP, JMS CR
JMS LF
ISZ FORCOU
JMP ++7
LAW 17774
DAC FORCOU
LAC CURLOC
SAD FA
JMP KB
JMP ++5
JMS SPACE
JMS SPACE
JMS SPACE
JMP AGAIN
ISZ CURLOC
JMP R0+2
LF, 0
LAW 212
TSF
JMP --1
TLS
JMP I LF
CR, 0
LAW 215
TSF
JMP --1
TLS
JMP I CR
CARRY1, 0
SNA
JMP I CARRY1
LAC C1
```



```

      ADD S2
      DAC S2
      LAC S3
      ADD (-12
      DAC S3
      ADD (-11
      SMA
      JMP .-8
      JMP I CARRY1
CARRY2, 0
      SNA
      JMP I CARRY2
      LAC (1
      ADD S1
      DAC S1
      LAC S2
      ADD (-12
      DAC S2
      ADD (-11
      SMA
      JMP .-8
      JMP I CARRY2
PRNTI, 0
      LAC SAVE2
      XOR TYPCOD
      TSF
      JMP .-1
      TLS
      JMP I PRNTI
VAR, 0
      DZM TM
      DZM VAR2
STRT, LAC I 13
      DAC TEMP1
      CMA
      ADD SAVE1
      SPA
      CMA
      DAC VAR1
      JMS CMY
      LAC VAR1
      TAD VAR2
      DAC VAR2
      ISZ TM
      LAC TM
      SAD (200
      JMP FIN1
      JMP STRT
FINI, LAC VAR2
      RCR          RCR          RCR

```

	RCR	RCR	RCR
	RCR		
	DAC VAR2	/VARIANCE	
PRNTDP,	JMP I VAR		
	0		
	LAC C56		
	TSF		
	JMP --1		
	TLS		
	JMP I PRNTDP		
INC,	LAC S6		
	ADD C-65		
	SMA		
	JMS PIO		
	LAC SAVE1		
PIO,	JMP IDX		
	0		
	LAC C1		
	DAC SAVE2		
	JMS PRNTI		
	DZM SAVE2		
	JMS PRNTI		
	JMS PRNTDP		
	JMP ENDP		
SPACE,	0		
	LAW 240		
	TSF		
	JMP --1		
	TLS		
	JMP I SPACE		
STORLO,	0		
	LAC TEMP1		
	DAC I CURLO		
	JMP I STORLO		
STORHI,	0		
	LAC TEMP1		
	DAC I CURHI		
	JMP I STORHI		
TYPE6,	0		
	DAC BUK		
	LAW 17772		
	DAC SIXCO		
	LAC BUK		
	RTL		
	RAL		
	DAC STOR		
	RAL		
	AND BITS		
	XOR TYPCOD		
	TSF		

```

      JMP -- 1
      TLS
      LAC STOR
      ISZ SIXCO
      JMP TYPE6+5
      JMP I TYPE6
BITS, 7
BITA, 77
BITB, 7700
BITC, 770000
CORE, 5777
TYPCOD, 260
TYPE3, 0
      DAC BUK
      LAW 17775
      DAC TYPE6+2
      LAC BUK
      RTL
      RTL
      RTL
      RTL
      RAL
      JMS TYPE6
      LAW 17772
      DAC TYPE6+2
      JMP I TYPE3
DTAB, 000000
      000175
      000372
      000567
      000764
      001161
      001356
      001553
DTAB1, 000000
      000020
      000037
      000057
      000076
      000116
      000136
      000155
TTAB, 000000
      050102
HTAB, 000000
      000604
      010210
      011102
      020506
      030200

```

B-43

BNWL-597

	031004
	040410
TENTAB.	000000
	000010
	000106
	000204
	000302
	000400
	000410
	000506
PAUSE	

Table B-IV. Listing of CSEMPLEX Code

Table B-IV. Listing of CSEMPLEX Code

```

CSEMPLEX01
/TELETYPE INTERRUPT SERVICE ROUTINE
      BUF=4000
20/      JMP INIT
      HLT
      PRG 10
      HLT
TELTYP,   TCF          /CLEAR TELETYPE FLAG
      LAM -0          /DECREMENT
      TAD TELCTR      /NO. OF CHARS.
      DAC TELCTR      /IN BUFFER
      SNA             /IS IT ZERO
      JMP CONTIN      /YES-RETURN
      LAC TELOUT      /GET BUFFER INDEX
      SAD TELMAX      /IS IT MAX.
      LAC TELMIN      /YES-RESET TO MIN.
      TAD C1          /INCR.
      DAC TELOUT      /AND STORE
      LAC I TELOUT    /GET CHAR.
      TIS             /TYPE IT
      JMP CONTIN      /AND RETURN
TELMIN,   TELBUF-1     /MIN. ADDR. FOR BUFFER
TELMAX,   TELBUF+77    /MAX.ADDR. IN BUFFER
TELOUT,   TELBUF-1     /INDEX TO PICK UP NEXT CHAR.
TELIN,    TELBUF-1     /INDEX TO STORE NEXT CHAR.
TELBUF,   TELBUF+1000 /TELETYPE BUFFER
TELCTR,   0           /NO. OF CHARS. IN BUFFER
START

```

CSEMPLEX32

/STORAGE REQUEST AND RETURN SUBROUTINES-INHIBITED ENTRY

```

STORED,      0
              LAM -4          /DECREMENT
              TAD SQCTR       /STORAGE BLOCK CNTR.
              SPA             /IS IT EMPTY
              JMP STREMP      /YES-GO TO EMERGENCY ROUTINE
              DAC SQCTR       /AND-STORE CNTR.
              DTM QMF         /ZERO QUEUE EMPTY MESS INDICATOR
              LAC SQIND       /GET QUEUE INDEX
              SAG SQMAX       /IS IT MAX
              LAC SQMIN       /YES-SET IT TO MIN
              TAD C1          /INCR.
              DAC SQIND       /STORE INDEX
              LAC I SQIND     /GET BLOCK ADDR.
              JMP I STORED    /RETURN WITH BLK. ADDR.

```

```

STORAD,      0
              DAC BLKADD     /SAVE BLK. ADDR.
              ISZ SQCTR       /INCR. STORAGE BLK. CNTR.
              LAC SQINI       /GET QUEUE INDEX
              SAG SQMAX       /IS IT MAX
              LAC SQMIN       /YES-SET IT TO MIN
              TAD C1          /INCR.
              DAC SQINI       /AND STORE
              LAC BLKADD     /P.U.ADDR.
              DAC I SQINI     /AND STORE
              JMP I STORAD    /RETURN

```

```

QMF,         0          /QUEUE EMPTY MESSAGE TYPED OR NOT
SQCTR,       14         /NO. OF BLKS AVAILABLE AS STORAGE
SQMIN,       SQUEUE-1   /MIN. ADDR. OF QUEUE
SQMAX,       SQUEUE+27   /MAX. ADDR. OF QUEUE
SQINI,       SQUEUE+7    /INDEX TO RETURN BLKS--RESERVES 6400 OCTAL LOC
SQMIN4,      SQUEUE+7    /FOR HISPEED MULTIPLEXER
SQIND,       SQUEUE+23   /INDEX TO GET BLKS.

```

```

SQUEUE,     BUF
            BUF 400
            BUF 1000
            BUF 1400
            BUF 2000
            BUF 2400
            BUF 3000
            BUF 3400
            BUF 4000
            BUF 4400
            BUF 5000
            BUF 5400
            BUF 6000

```



```

BUF 6400
BUF 7000
BUF 7400
BUF 10000
BUF 10400
BUF 11000
BUF 11400
BUF 12000
BUF 12400
BUF 13000
BUF 13400
STREMP, LAM -1 /DECR. RETURN
TAD STOREQ /ADDR. TO IOF
DAC 0 /SAVE IT
JMS SARR /SAVE REGISTERS
LAC QMF /HAS MESSAGE
SZA /BEEN TYPED
JMP EXEC /YES
JMS TYP OUT /NO TYPE OUT
JMP .+2 /QUEUE
MESS7 /EMPTY
ISZ QMF
JMP EXEC

```

```

MESS7, TEXT
STORAGE QUEUE EMPTY

```

```

START

```

CSEMPLEX03

/MAGTAPE OUTPUT QUEUE HANDLER SUBROUTINES-ENTER INHIBITED

```

MGQREQ,      0
              LAM -0          /DECR.
              TAD MQCTR      /MAG. TAPE QUEUE CNTR.
              SPA           / IS IT EMPTY
              JMP MQRET      /YES-RETURN WITH NEG. ADDR.
              DAC MQCTR      /NO-STORE CNTR.
              LAC MQINO      /GET QUEUE INDEW
              SAD MQMAX      /IS IT MAX.
              LAC MQMIN      /YES-SET IT TO MIN.
              TAD (1        /INCR.
              DAC MQINO      /STORE IT
              LAC I MQINO    /GET ADDR.
MQRET,        NOP
              JMP I MGQREQ   /RETURN

```

```

MGQAD,        0
              DAC BLKADD     /SAVE BLK. ADDR.
              ISZ PROG 20    /TURN ON MAG. TAPE HANDLER
              ISZ MQCTR      /INCR. CTR.
              LAC MQINI      /GET QUEUE INDEX
              SAD MQMAX      /IS IT MAX
              LAC MQMIN      /YES-SET IT TO MIN.
              TAD (1        /INCR.
              DAC MQINI      /AND STORE
              LAC BLKADD     /P.U. ADDR.
              DAC I MQINI    /AND STORE
              JMP I MGQAD    /RETURN

```

```

MQCTR,        0
MQMIN,        MQUEUE-1
MQMAX,        MQUEUE 27
MQINI,        MQUEUE-1
MQINO,        MQUEUE-1
BLKADD,       LAM -0
MQUEUE,       MQUEUE+30/

```

START

CSEMPLEX04
/ INTERRUPT HANDLER

```

                                T0SF=704201 /SKIP IF WIRE IS BROKEN
                                /ON RUPTURE DISC
                                ADSF=701301
INTER,  DAC SAVEAC
                                MMEF /DECTAPE ERROR
                                SKP
                                JMP MMERR
                                MMDF /DECTAPE DATA
                                SKP
                                JMP I MMDATA
                                MMBF /DECTAPE BLOCK
                                SKP
                                JMP WRTBLK
                                ADSF /ANALOG-DIG. CONV. COMPLETE
                                SKP
                                JMP SCD
                                CLSF /CLOCK OVFL
                                SKP
                                JMP CLOK
                                TSF /TELETYPE COMPLETE
                                SKP
                                JMP TELTYP
                                KSF /KEYBOARD ENTRY
                                SKP
                                JMP KEYB
                                T0SF /IS IT TIME 0
                                SKP
                                JMP TIME0

                                JMS TYPOUT /TYPE OUT ILLEGAL INTERRUPT
                                JMP . 2 /QUEUE FULL-DELETE MESSAGE
                                MESS1 /MESS. ADDR.

                                RRB /CLEAR READER FLAG
                                PCF /CLEAR PUNCH FLAG
                                723001 /CLEAR ANALYZER FLAG
                                NOP

CONTIN, LAC 0 /RESTORE
                                RCL /LINK
                                LAC SAVEAC /RESTORE AC
                                ION
                                JMP I 0 /RETURN TO INTERRUPTED PROGRAM

SAVEAC, 0

```

1/ JMP INTER /JUMP TO INTERRUPT SERVICE ROUTINE
SAVEAC+1/

MESS1, TEXT
ILLEGAL INTERRUPT

START

CSEMPLX05

/TELETYPE OUTPUT QUEUE HANDLER-INHIBIT BEFORE ENTRY

```

TYP0UT,      0
              LAC TQCTR    /IS QUEUE
              SAD (20      /FULL
              JMP I TYP0UT          /YES-RETURN
              ISZ TQCTR    /INCR. CTR.
              ISZ TYP0UT   /SET MESSAGE PTR.
              LAC TQINI    /GET QUEUE INPUT INDEX
              SAD TQMAX    /IS IT MAX
              LAC TQMIN    /YES-SET TO MIN.
              TAD (1       /INCR.
              DAC TQINI    /AND STORE
              LAC I TYP0UT          /STORE
              DAC I TQINI /MESS. ADDR.
              ISZ TYP0UT /AND
              JMP I TYP0UT          /RETURN

```

```

TQMIN,        TQUEUE-1    /MIN ADDR. OF QUEUE
TQMAX,        TQUEUE 17   /MAX ADDR. OF QUEUE
TQINI,        TQUEUE-1    /INDEX TO STORE MESSAGE
TQINO,        TQUEUE-1    /INDEX TO P.U. MESSAGE

```

```

TQUEUE,      TQUEUE+20/

```

START

CSEMPLEX06

/WAIT AND EXIT MACROS AND REGISTER STORE

```

EXIT=JMP .
IOF
LAM -0      /SET
TAD PROG1   /PROG.
TAD (PROG
DAC 17      /STORE INDEX
DZM I 17    /TURN OFF PROGRAM
DZM I 17    /ZERO AC STORE
LAC I 17    /P.U. INIT. ADDR.
DAC I 17    /STORE RETURN ADDR.-ZERO LINK
DZM I 17    /ZERO 10
JMP EXEC    /GO TO ECP

```

WAIT=JMS .

```

0
IOF
DAC SAVEAC  /SAVE ACCUM.
LAC (PROG   /SET
TAD PROG1   /PROG.
DAC 17      /INDEX
LAC SAVEAC  /STORE
DAC I 17    /AC
ISZ 17      /BY PASS INIT. PROG. ADDR.
CLA         /GET
RAR         /LINK
TAD WAIT -JMS          /ADD PROG. RETURN ADDR.
TAD (-2          /BACK IT UP 3 TO WAIT TEST
DAC I 17        /STORE IT
LAC 10          /SAVE REGISTERS
DAC I 17
JMP EXEC        /GO TO ECP

```

SARR,

Ø
LAC (PROG
TAD PROG1
DAC 17
LAC SAVEAC
DAC I 17
ISZ 17
LAC Ø
DAC I 17
LAC 1Ø
DAC I 17
JMP I SARR

START

CSEMPLEX07

/EXECUTIVE CONTROL PROGRAM (#0)

```

EXEC,      LAC PROG1    /SET
           TAD (PROG
           DAC PROGX     /INDEX
EXEC1,     DZM PROG1     /SET RUNNING PROGRAM TO ZERO
           IDN           /ALLOW
           NOP
           IOF           /FOR INTERRUPT
           LAC PROGX     /INCR.
           SAD PROGMX
           LAC (PROG
           TAD (10       /PROG. CNTR.
           DAC PROGX
           LAC I PROGX   /IS PROG N
           SNA           /TURNED ON
           JMP EXEC1     /NO-LOOP
           IDN           /ALLOW
           NOP           /FOR
           IOF           /INTERRUPTS

RARR,      LAC PROGX     /RESTORE
           DAC 17        /REGISTERS
           LAC I 17      /FOR
           DAC SAVEAC    /RUNNING PROGRAM
           ISZ 17
           LAC I 17
           DAC 0
           LAC I 17
           DAC 10
           LAC PROGX
           TAD (-PROG+1
           DAC PROG1     /SET RUNNING PROG NO.
           JMP CONTIN    /GO TO PROGRAM

PROGMX,    PROG 60
PROG1,     PROG
PROGX,     PROG

PROG,      0
.+7/       0

          0
          SCN           /PROGRAM #1
.+5/       0

```


B-54

BNWL-597

	Ø	
	MAG	/PROGRAM #2
•+5/	TQCTR,	Ø
	Ø	
	IOP	/PROGRAM #3
•+5/	Ø	
	Ø	
	SSW	/PROGRAM #4
•+5/	Ø	
	Ø	
	EMER	/PROGRAM #5
•+5/	Ø	
	Ø	
	HISPD	/PROGRAM #6
•+5/	Ø	

START

CSEMPLEX08

/INPUT OUTPUT APPLICATION PROGRAM (#3) FOR TELE TYPE

```

IOP,      LAC TQIND    /GET QUEUE INDEX
          SAD TQMAX    /IS IT MAX
          LAC TQMIN    /YES-SET IT TO MIN
          TAD C1       /INCR.
          DAC TQIND    /AND STORE
          LAM -0       /GET MESSAGE ADDR.
          TAD I TQIND  /AND SET
          DAC I0       /INDEX

```

```

IOP1,     LAC I I0     /P.I. MESSAGE WORD
          DAC WORD     /NO-SAVE WORD
          LAM -2       /SET
          DAC CHARCT   /CHAR. CNTR.

```

```

IOP2,     LAC WORD
          RTL          /ROTATE
          RIL          /LEFT
          RTL          /SIX
          DAC WORD     /SAVE FOR NEXT
          AND C37      /MASK IT
          SNA          /END OF MESS.
          JMP IOP3     /YES
          TAD CBTAT-1  /SET
          DAC BTAT     /CONV. INDEX
          LAC I BTAT   /GET
          SNL          /EQUIV. ASCII
          RR9          /BAUDOT U.C.
          JMS TELSTR   /PUT IT IN QUEUE
          ISZ CHARCT
          JMP IOP2     /GET NEXT CHAR.
          JMP IOP1     /GET NEXT WORD

```

```

TELSTR,   0
          AND C377     /MASK IT
          DAC BTAT     /SAVE IT
          LAC TELCTR   /IS
          SNA          /BUFFER EMPTY
          JMP TELST    /YES-GO TYPE CHAR.
          IOF
          LAC TELCTR   /IS
          SAD C100     /BUFFER
          WAIT         /FULL--YES
          LAC TELIN    /GET BUFFER INDEX
          SAD TELMAX   /IS IT MAX
          LAC TELMIN   /SET IT TO MIN
          TAD C1       /INCR.
          DAC TELIN    /AND STORE

```

```

LAC BTAI      /STORE
ISZ TELCTR
DAC I TELIN /CHAR.
ION
JMP I TELSTR

TELST,        ISZ TELCTR  /=1
LAC BTAI      /GET CHAR
ILS          /TYPE IT
JMP I TELSTR      /RETURN

IOP3,        LAW 212      /LF
JMS TELSTR
LAW 215
JMS TELSTR /CR
LAM -0        /DECR. CNTR
IOF
IAD TQCTR
DAC TQCTR
SNA          /IS IT ZERO
EXIT         /YES-QUIT
ION          /NO-GET
JMP IOP       /NEXT MESSAGE

RR9=JMS .
Ø
RCR
RTR          RTR          RTR          RTR
JMP I RR9-JMS

CHARCT,      Ø
BTAI,        Ø
WORD,        Ø

BTAT,        324265
              215215
              317271
              240240
              310243
              316254
              315256
              212212
              314251
              322264
              307246
              311270
              320260
              303272
              326273
              305263
              332242

```

B-57

BNWL-597

304244
302277
323211
331266
306241
330257
301255
327262
312247
200200
325267
321261
313250
200200

START

CSEMPLX09

/MAG. TAPE OUTPUT PROGRAM (#2) AND DRIVER

MAG,	IOF	/GET ADDR.
	JMS MGOREO	/OF BLK
	ION	/TO WRITE
	SPA	
	EXIT	/QUEUE EMPTY
	DAC CSAV	
	DAC CSA	/SAVE MEMORY ADDR.
MAG1,	ISZ BLOCK	/INCR. DECTAPE BLOCK CNTR.
	LAC UNIT	/SELECT
	MMSE	/DECTAPE
	LAW SVC2	/SET DATA
	DAC MMDATA	/FLAG RETURN
	LAW 61	/SEARCH
	MMLC	/REVERSE
	DAC MTBSY	/SET MAG. TAPE BUSY
MAG2,	LAC MTBSY	/WAIT
	SZA	/FOR
	WAIT	/BLK. TO BE WRITTEN
	JMP MAG	
MAG3,	CLA	/DESELECT
	MMLC	/AND STOP
	MMSE	/UNIT
	DISMIS	
MAG4,	LAM-200	/SET DESELECT DELAY
	DAC CHECK	/200 MICROSEC/PASS THRU WAIT EIC
	NDP	/WAIT RETURN
	ISZ CHECK	/DELAY ELAPSED
	WAIT	/NO-WAIT
	JMP MAG1	/YES-GO SELECT UNIT
SVC0,	LAW SVC1	/SET DATA
	DAC MMDATA	/FLAG RETURN
	LAW 41	/SEARCH FORWARD
	MMLC	
	DISMIS	
SVC1,	MMRD	/READ BLOCK NO.
	SAD BLOCK	/IS IT RIGHT
	JMP SVC3	/YES-GO TO WRITE
	CMA	/NO-IS
	TAD BLOCK	/THE NO.
	SMA	/LESS
	DISMIS	/YES

```

LAW SVC2      /NO, OVERSHOT
DAC MMDATA    /SEARCH
LAW 61        /REVERSE
MMLC
DISMIS

SVC2,         MMRD      /READ BLK.
              CMA       /IS IT
              TAD BLOCK /LESS THAN
              SPA       /DESIRED
              DISMIS    /NO
              JMP SVC0   /YES-GO SRCH. FORWARD
SVC3,         LAW SVC5   /SET DATA
              DAC MMDATA /FLAG RETURN
              LAW 43     /WRITE
              MMLC      /FORWARD
              CLC       /SET
              DAC CHECK  /CHECK SUM
              DISMIS
SVC5,         LAC I CSA   /GET WD. TO WRITE
              ISZ CSA    /INCR. ADDR.
              MMWR      /WRITE IT
              ADD CHECK  /ADD TO
              DAC CHECK  /CHECKSUM
              DISMIS
WRTBLK,       LAC CHECK  /WRITE
              CMA
              MMWR      /CHECKSUM
              DZM MTU    /RESET UNABLE INDICATOR
              LAC CSAV   /RETURN BLOCK
              JMS STORAD /TO STORE QUEUE
              JMS MGOREQ /GET NEXT
              SPA       /MEM. ADDR. FROM QUEUE
              JMP MQEMP  /EMPTY
              DAC CSAV
              DAC CSA
              ISZ BLOCK
              JMP SVC3+4 /GO WRITE NEXT BLOCK

MQEMP,        LAW OUT    /SET DATA FLAG RETURN
              JMP SVC0 1  /AND SEARCH FORWARD

MTBSY,        0
BLOCK,        0          /CURRENT BLOCK ON DECTAPE DESIRED
UNIT,         010000     /CURRENT DECTAPE UNIT SELECTED
MMDATA,       XX
CHECK,        0
CSAV,         0
CSA,          0
DISMIS=JMP CONTIN

```

```

OUT,      DZM MTBSY
          JMP MAG3

MMERR,    MMRS      /GET TAPE STATUS
          AND (040000 /IS TAPE IN
          SNA      /EOT AREA
          JMP MMERR1 /NO
          MMRS      /IS TAPE
          AND (010000 /GOING
          SZA      /FORWARD
          JMP SVC0   /NO-SEARCH FORWARD
          LAC UNIT   /YES-PUT UNIT #
          RCL      RTL      RTL      RTL
          JMS OCVT   /INTO
          MESS2 4     /MESSAGE
          JMS TYP0UT /TYPE
          JMP . 2     /MESSAGE
          MESS2      /TO REMOVE
          CLA        /DESELECT
          MMLC       /UNIT
          MMSE       /AND STOP
          JMS SARR    /SAVE ALL REG.
DIAG0,    LAC UNIT   /COUNTDOWN RETURN
          AND MSGW    /SWITCH
          ADD (010000 /TAPE
          DAC UNIT    /UNIT
          LAC (4      /RESET BLK CNTR.
          DAC BLOCK
          LAC (20     /SET RUNNING PROGRAM
          DAC PROG1   /TO #2
          ION
          JMP MAG4    /GO SELECT ALT. UNIT

MSGW,     030000
          JMP MAG4    /GO SELECT ALT. UNIT

MMERR1,   MMRS      /IS IT A
          AND (022000 /TRUE ERROR
          SNA
          JMP MMERR3 /NO

```

```

MMERR2,  LAC BLOCK      /GET BLOCK #
          ADD UNIT      /ADD UNIT AND SHIFT RIGHT 9
          RTR           RTR           RTR           RTR           RAR
          AND (777      /MASK
          JMS CCVT      /ADD TO MESSAGE
          MESS3 1
          LAC BLOCK      /GET BLOCK # LEAST
          AND (777      /SIG. DIGITS
          JMS CCVT      /ADD TO MESSAGE
          MESS3 2
          JMS TYP OUT    /IYPOUT
          JMP . 2        /MAG TAPE
          MESS3          /ERROR MESSAGE
          ISZ BLOCK      /INCR. BLK. CNTR.
          LAC CSAV       /RESET MEMORY
          DAC CSA        /ADDR.
          JMP SVCM       /GO WRITE IT ON NEXT BLK

MMERR3,  MMRS           /IS TAPE
          AND (001000    /UNABLE
          SVA
          JMP MMERR2     /NO-TREAT AS A TRUE ERROR
          LAC MTU        /HAS
          SZA           /UNABLE MESSAGE BEEN TYPED
          JMP . 12       /YES
          ISZ MTU        /NO-SET SW.
          LAC UNIT      /PUT UNIT #
          RCL           RTL           RTL           RTL
          JMS CCVT      /IN MESSAGE
          MESS4 2
          JMS TYP OUT    /IYPE
          JMP . 2        /II
          MESS4
          JMS SARR
          LAC (20        /GO
          DAC PROG1      /TRY
          JMP MAG1 1     /AGAIN

```


B-62

BNWL-597

MESS2, TEXT
REPLACE TAPE
Ø
MESS3, TEXT
MTE
Ø
Ø
MESS4, TEXT
TAPE XX
MESS4A, TEXT
UNABLE
MTU, Ø
START

CSEMPLEX10

/SCANNER CONTROL PROGRAM (#1)

```

SCN,      JMS BUFREN  /GET BUFFER STORAGE

SCN1,      JMS TIMCHK
           TAD I IPERT I
           TAD I IPERT
           JMP EOSPER  /TIME FOR NEXT PERIOD

SCN2,      LAC ISCHII  /INITIALIZE---
           DAC ISCHII  /SCAN INTERVAL INDEX
           LAC ICLTMI  /CLASS TIME
           DAC ICLTMI  /INDEX
           LAC IPTII   /CLASSPOINTI
           DAC IPTII   /INDEX
           LAC C1      /CLASS
           DAC ICL      /INDEX
           LAC ISCIMI  /CLASS INTERVAL
           DAC ISCIMI  /INDEX
           LAM-7       /SET CLASSES OFF
           DAC ICL0    /COUNTER

SCN3,      LAC I ICLTM /IS CLASS
           ISZ ICLTM
           SPA         /TURNED OFF
           JMP SCNO    /YES
           DAC ICLP
           JMS TIMCHK
           TAD I ICLTM
           TAD ICLP
           SKP         /TIME FOR THIS CLASS
           JMP SCNO

SCN3B,     LAC I IPTI  /YES-SET
           DAC ICLP    /CLPER PNTR.

SCN4,      ISZ ICLP
           LAC ICLP    /IS THIS
           SAD ICLPF    /LAST PT.
           JMP SCN5

SCN4A,     JMP SCN4B   /BECOMES NOP AFTER FIRST PERIOD

           LAC I ICLP  /P.O. CLPER WD.
           AND C7      /MASK IT
           SAD ICL      /THIS CLASS
           SKP         /YES-GO SCAN II
           JMP SCN4     /NO LOOP

```

```

SCN4B,  LAC HSPFL  /ANY HISPD UNITS RUNNING
        SZA
        JMP SCN5   /YES, BYPASS REST OF THIS SCAN CLASS
        LAC SDCTR  /IS
        SNA        /SCANNER THRU
        JMP SCN4C  /YES
        LAC TQCTR  /ANY MESSAGES
        SZA        /TD TYPE
        WAIT       /YES
        LAC MTBSY  /IS MAG TAPE
        SNA        /BUSY
        SKP        /NO GO CHECK IT
        JMP SCN4B  /YES
        NOP
        WAIT

SCN4C,  LAC BUFAD  /IS
        SAD BUFADF /BUFFER FULL
        SKP
        JMP SCN4D  /NO
        LAC BUFADI /YES WRITE IT
        IOF
        JMS MGOAD  /OUT TO TAPE
        ION
        JMS BUFREQ /GET BUFFER STORAGE

SCN4D,  LAC ICLP   /SET
        DAC I IPTI /POINT INDEX
        AND C77
        703004    /START SCANNER
        DAC I BUFAD /STORE ID
        LAC BUFAD   /SET DRIVER
        DAC DBUFAD  /BUFFER ADDR.
        LAM -3      /INITIALIZE
        DAC SDCTR   /DRIVER COUNTER
        LAC ICL     /STORE INFO.
        RCL         /IN
        RTL         /BUFFER
        TAD IPER
        ISZ BUFAD
        DAC I BUFAD
        LAC I ISCI
        AND C77
        ISZ BUFAD
        DAC I BUFAD
        ISZ BUFAD
        LAC CLKL    /LEAST SIG. CLOCK
        AND C17     /LEAST SIG. 4 BITS
        RCL         /LEFT ONE POSITION
        DAC I BUFAD

```

```

EOSPER,    ISZ IPER
            LAM -6      /IS JOB
            TAD IPER    /DONE
            SNA
            JMP TERM    /YES-TERMINAL PROCEDURE
            ISZ IPERT    /NO
            ISZ IPERT    /INCR. PERT
            ISZ IPERT 1  /POINTERS
            ISZ IPERT 1
            LAC ICLTM
            DAC ICLTM
            LAC ISCHH
            DAC ISCHH
            LAC ISCTM    /INCR.
            TAD C7      /SCM
            DAC ISCTM    /PNTR.
            DAC ISCTM
            LAC IPTII    /RESET
            DAC IPTI     /POINT TABLE PNTR.
            LAM -6      /INITIALIZE
            DAC ICL      /LOOP INDEX
EOP1,       LAC I ISCTM  /SET
            SMA          /UP
            LAC CLKM     /CLTM,
            DAC I ICLTM  /SCII
            ISZ ICLTM
            DZM I ISCHH
            LAC CLKL
            DAC I ICLTM
            LAC ICLPI
            DAC I IPTI
            ISZ IPTI
            ISZ ICLTM
            ISZ ISCTM
            ISZ ISCHH
            ISZ ICL
            JMP EOP1
EOP2,       JMP EOP0     /ST TO NOP AFTER FIRST PASS
            LAC ICLPT
            DAC ICLP
EOP3,       ISZ ICLP
            LAC ICLP
            SAD ICLPF
            JMP EOP0
            LAC I ICLP   /ROTATE
            RCR          /CLPER
            RTR          /3
            DAC I ICLP   /RIGHT
            JMP EOP3

```

```

ISZ BUFAD
LAC K4      /START
DAC COUNT 1 /DIAG. CNTR.
JMP SCN1

SCN5,      LAC ICLPI
           DAC I IPTI
           ISZ I ISCTI /INCR. SCII
           CLL          /DOUBLE PRECISION
           LAC I ICLTM /INCR.
           TAD I ISCTM /CLTM
           DAC I ICLTM
           GLK          /GET OVFL
           DAC 10       /SAVE IT
           LAM -2       /ADJUST
           TAD ICLTM    /PNTR.
           DAC ICLIM
           LAC 10
           TAD I ICLIM
           DAC I ICLIM
           ISZ ICLTM
           JMP SCN0 1   /BYPASS INCR. CLASS OFF CNTR.

BUFREQ,    0
           IOF
           JMS SIORON  /GET BUFFER ADDR.
           IOV
           DAC BUFADI  /SET
           DAC BUFAD   /UP
           TAD K400    /BUFFER
           DAC BUFADF  /CONTROL
           JMP I BUFREQ          /WORDS

TIMCHK,    0
           LAC CLKM    /ROUTINE TO CHECK
           CMA         /IF TIME LIMIT
           DAC CLKMC    /EXCEEDED
           LAC CLKL    /IN DOUBLE PRECISION
           CMA!CLL
           XCI I TIMCHK
           ISZ TIMCHK
           CLA!RAL
           TAD CLKMC
           XCI I TIMCHK
           ISZ TIMCHK
           SMA
           ISZ TIMCHK /NOT EXCEEDED
           JMP I TIMCHK /EXCEEDED

```

EDP0,	LAC CNOP	/SET
	DAC EDP2	/SWITCHES
	DAC SCN4A	/TO NOP
	JMP SCN1	
SCN0,	ISZ ICLO	/INCR. CLASSES OFF CNTR.
	ISZ ICLIM	
	ISZ ICL	
	LAC ICL	/HAVE ALL
	AND C7	/CLASSES BEEN
	SNA	/CHECKED
	JMP SCN6	/YES-GO WAIT
	ISZ ISCTI	
	ISZ ISCTM	
	ISZ IPTI	
	JMP SCN3	
SCN6,	ISZ ICLO	/ARE ALL CLASSES OFF
	JMP SCN1	/NO-GO CHECK WHICH ONE ON
	WAIT	/YES WAIT
TERM,	LAM -0	/FILL
	DAC I BUFAD	/REST
	ISZ BUFAD	/OF
	LAC BUFAD	/BUFFER
	SAD BUFADF	
	SKP	
	JMP TERM	
	LAC BUFADT	/WRITE IT
	JMS MGDAD	/OUT TO TAPE
	IDF	
	JMS TYPDUT	/TYPE OUT
	WAIT	/TERMINATION
	MESS6	/MESSAGE
	ION	
	EXIT	/END OF SCANNING
IPERT,	PERT	/INDEX TO PERT
	PERT+1	
IPER,	0	/CURRENT PERIOD

ISCTMI,	SCTM	/INDEX TO SCTM			
ISCTM,	SCTM				
ICL,	Ø	/CURRENT SCAN CLASS			
ICLO,	Ø	/SCAN CLASSES OFF CNTR.			
ICLTM,	CLTM	/INDEX TO CLTM			
ICLTM,	CLTM				
ISCHII,	SCII	/INDEX TO SCII			
ISCH,	SCII				
ICLP,	CLPER				
IPTH,	IPT	/INDEX TO IPT			
IPTI,	IPT				
BUFAD,	Ø				
BUFADI,	LAM - Ø	/INDEX TO IPT			
BUFADF,	Ø				
DBUFAD,	Ø				
SCII,	Ø	/SCAN CLASS INTERVAL NUMBER			
	Ø	Ø	Ø	Ø	
	Ø	Ø	Ø	Ø	
CLTM,		/DOUB. PREC. TIMES TO SCAN			
		/CLASS CLTM(ICL)			
	LAM	LAM	LAM	LAM	LAM
	LAM	LAM	LAM	LAM	LAM
	LAM	LAM	LAM	LAM	LAM
IPT,	CLPER-1	/PT. LAST SCANNED IN CLASS			
	CLPER-1	CLPER-1	CLPER-1		
	CLPER-1	CLPER-1	CLPER-1		
BUF-200/					
SCTM,	SCTM+61/	/SCANNING INTERVALS MSEC			
		/SCTM (7,7), SCTM (IPER,ICL)			

B-69

BNWL-597

PERT, PERT+16/ /DOUB. PREC. TIMES TO SW. PERIODS
 /PERT (IPER)

CLKMC, 0

CLPER, 0

CLPER+64/

ICLPI, CLPER-1

ICLPE, CLPER+60

IPT+7/

MESS6, TEXT

RUN IS COMPLETED

START

CSEMPLEX11
/SCAN DRIVER

```
      ADRB=701312
SCD,   ADRB      /READ CONV
      TAD I DBUFAD      /COMBINE
      DAC I DBUFAD      /WITH DATA ID
      ISZ SDCTR /ARE WE THRU
      JMP SCD1  /NO-GO SELECT NEXT PT.
      TAD T0FL
      DAC I DBUFAD
      LAM -0      /TURN OFF DIAG. CNTR.
      JMP SCD2

SCD1,   703042      /SELECT AND INIT. A TO D
      LAC C4      /START
SCD2,   DAC COUNT 1 /DIAG. CNTR.
      ISZ DBUFAD
      JMP CONTIN

SDCTR,   0

START
```

CSEMPLEX12

/CLOCK SERVICE ROUTINE

```

CLOK,      LAC CLKCON /RESTART
           TAD 7
           DAC 7 /CLOCK
           CLON
           ISZ CLKL /UPDATE
           SKP /CLOCK
           ISZ CLKM /CELLS
CLOKSW,    JMP CLOK2 /SW SET BY HISPD (#6) TO NOP
           LAM -5
           DAC 17 /SCAN
           LAC COUNT-1 /DIAG
           DAC 16 /COUNTERS
           DAC 15

CLOK1,     LAC I 16 /IS COUNTER ON
           SPA
           JMP CLOK3 /NO LOOP
           TAD (-0) /DECR.
           SMA!SZA /TROUBLE
           JMP CLOK3 /NO-LOOP
           LAM-0 /TURN OFF COUNT
           DAC I 15
           LAC 17 /YES-GENERATE
           TAD (JMP CLOKV /DIAGNOSTIC
           DAC . 1 /BRANCH
           XX
           JMP .
           JMP SCFL /SCAN FAIL
           JMP DEL0 /HISPD SCAN 0 DELAY COMPLETED
           JMP DEL1 /HISPD SCAN 1 DELAY COMPLETED
           JMP DEL2 /HISPD SCAN 2 DELAY COMPLETED
           JMP DEL3 /HISPD SCAN 3 DELAY COMPLETED
CLOKV,CLOK2, LAC COUNT 1 /IS SCN CNTR
           SPA /ON
           JMP CONTIN /NO
           TAD (-0) /YES-DECR.
           DAC COUNT 1 /AND SAVE IT
           SMA!SZA /FAILURE
           JMP CONTIN /NO

```

```

SCFL,      703002      /CYCLE
           703002      /MULTIPLEXOR
           703002      /AROUND AND
           703042      /TRY AGAIN
           JMS TYP0UT   /TYPE OUT
           JMP . +2     /FAILURE
           MESS5       /MESSAGE
           LAC (4      /RESTART
           DAC COUNT+1 /COUNTER
           JMP CONTIN

COUNT,    LAM          LAM          LAM          LAM
           LAM          LAM

CLKL,      0
CLKM,      0
CLKCON,    0
MESS5,     TEXT
SCANNER FAILED
CLOK3,     DAC I 15     /SAVE COUNTER
           ISZ 17      /THRU
           JMP CLOK1    /NO LOOP
           JMP CONTIN   /YES

EHSMP0=704002      /ENABLE HI SPD MPX 0
EHSMP1=704042      /ENABLE HI SPD MPX 1
EHSMP2=714002      /ENABLE HI SPD MPX 2
EHSMP3=724002      /ENABLE HI SPD MPX 3

DEL0,      EHSMP0      /TURN ON HISPD MPX 0
           ISZ HSPFL    /INCR. MPX ON FLAG
           ISZ DELCTR   /INCR. DELAYS COMPLETE CNTR.
           JMP CLOK3 1  /GO LOOP
           JMP RSCLSW   /RESET SWITCH

DEL1,      EHSMP1      /DITTO MPX 1
           JMP DEL0 1

DEL2,      EHSMP2      /DITTO MPX 2
           JMP DEL0 1

DEL3,      EHSMP3      /DITTO MPX 3
           JMP DEL0 1

RSCLSW,    LAC (JMP CLOK2      /RESET CLOKSW
           DAC CLOKSW
           JMP CONTIN

DELCTR,    0

START

```

CSEMPLEX13
/ START UP PROGRAM (#4)

```
SSW,      JMS TYP0UT
           JMP . 2      /TYPE OUT MESSAGE TO
           MESS8        /MOUNT READIN TAPE
           LAC IQCTR    /WAIT
           SZA          /FOR
           WAIT         /MESSAGE
           LAC TELCTR   /TO
           SZA          /BE
           WAIT         /TYPED
           IOF
           HLT
           LAC (3477    /SET
           DAC 17       /INDEXES
           LAM -277     /TO STORE
           DAC 16       /TABLES
SSWA,      RSB          /READ
           RSF          /IN
           JMP .-1
           RRB          /SUT
           DAC I 17
           ISZ 16       /PERT
           JMP SSWA     /AND CLPER
           EXIT
```

MESS8, TEXT
PLACE READIN TAPE FROM MPGEN IN READER, PRESS CONTINUE

```

                                ORS=742104
SSW4,    LAC SUTI      /INITIALIZE
          DAC 10       /INDEX TO S.U. TABLE

SSW1,    LAC I 10      /P.U. S.U. COMMAND
          AND C7       /MASK IT
          TAD C JMP SVECT      /GENERATE
          DAC SBR       /BRANCH

SBR,     XX

SVECT,   JMP SJ00      /00
          JMP SJ01      /01
          JMP SJ02      /02
          JMP SJ03      /03
          JMP SJ04      /04
          JMP SJ05      /05
          XX            /06
          XX            /07

SJ00,    LAC I 10
          TAD CLKM
          DAC SJ03I     /TIME MOST SIG.
          CLL
          LAC I 10
          TAD CLKL
          DAC SJ03I 1
          SZL
          ISZ SJ03I     /TIME LEAST SIG.

SJ00T,   LAC CLKM      /HAS DELAY ELAPSED
          CMA
          DAC SJ03I 2 /DOUBLE PRECISION
          LAC CLKL
          CMA!CLL
          TAD SJ03I 1
          CLA!RAL
          TAD SJ03I 2
          TAD SJ03I
          SKP
          JMP SJ03I     /WAIT RETURN
          BMA

SJ00W,   WAIT         /NO-WAIT
          JMP SSW1      /YES-GET NEXT COMMAND

SJ01,    LAC I 10      /SET RELAY
          740501       /BUFFER
          JMP SSW1

```

SJ02, LAC I 10 /GET ADDR.
 IAD CDAC /GEN. INSTR.
 DAC .+2
 LAC I 10 /XEC
 XX /STORE
 JMP SSW1

SJ03, LAC I 10 /STORE
 DAC SJ03I
 LAC I 10 /INSTR.
 DAC SJ03I 1
 LAC I 10
 DAC SJ03I 2

SJ03I, XX /XEC
 XX
 XX /INSTR.
 JMP SSW1

SJ04, EXIT
 IOF
 DZM CLKL
 DZM CLKM
 LAC I 10
 DAC CLKCON
 DAC 7
 CLON
 ION
 JMP SSW1

SUTI, BUF-301

BUF-300/

SUT, SUT+100/

SUTI+1/

START

CSEMPLEX14

/ROUTINES TO HANDLE HIGH SPEED MULTIPLEXERS-PROGRAM #6

```

T0FF0=704204      /INHIBIT FURTHER INTERRUPTS FROM RUPTURE DISC
S HSMP0=704021     /SKIP ON HI SPEED MPX 0 ENABLED
SHSMP1=704041     /SKIP ON HI SPEED MPX 1 ENABLED
SHSMP2=714001     /SKIP ON HI SPEED MPX 2 ENABLED
SHSMP3=724001     /SKIP ON HI SPEED MPX 3 ENABLED

```

```

TIME0,      T0FF0      /RESET AND LOCKOUT TIME 0 FLIP FLOP
             ISZ T0FL   /INDICATE TIME ZERO TO SCAN PROGRAM
             LAC CLPER+60      /START
             DAC COUNT+2 /DELAY
             LAC CLPER+61      /PERIODS
             DAC COUNT+3 /FOR
             LAC CLPER+62      /HISPD
             DAC COUNT+4 /MPXS
             LAC CLPER+63
             DAC COUNT+5
             LAC (NOP) /SET CLOK SWITCH
             DAC CLOKSW /SO AS TO TEST HISPD DELAYS
             LAM-3 /SET CNTR TO
             DAC DELCTR /RESET CLOKSW
             LAC (1) /TURN ON HI SPEED MULTIPLEXER
             DAC PROG+60 /SERVICE PROGRAM
             JMP CONTIN

```

```

HISPD,      LAM -3      /INITIALIZE NO. OF MULTIPLEXERS ON
             DAC HSPE   /COUNTER

```

```

HISPD0,     LAM-0      /HAS DELAY
             SAD COUNT+2 /PERIOD 0 ELAPSED
             SKP        /YES
             JMP .+3     /NO
             SHSMP0     /MPX0 DONE

```

```

HSSW0,      JMP DHS0    /YES
             LAM-0      /HAS DELAY
             SAD COUNT+3 /PERIOD 1 ELAPSED
             SKP        /YES
             JMP .+3     /NO
             SHSMP1     /MPX 1 DONE

```

```

HSSW1,      JMP DHS1    /YES
             LAM-0      /HAS DELAY
             SAD COUNT+4 /PERIOD 2 ELAPSED
             SKP        /YES
             JMP .+3     /NO
             SHSMP2     /MPX 2 DONE

```

```

HSSW2,    JMP DHS2      /YES
          LAM-0         /HAS DELAY
          SAD COUNT+5   /PERIOD 3 ELAPSED
          SKP           /YES
          JMP .+3       /NO
          SHSMP3        /MPX 3 DONE
HSSW3,    JMP DHS3      /YES

          SKP
          JMP HISPD0    /WAIT RETURN
          NOP
          WAIT

DHS0,     LAC (NOP
          DAC HSSW0
          JMS DMP
          BUF 3400

DHS1,     LAC (NOP
          DAC HSSW1
          JMS DMP
          BUF 5400

DHS2,     LAC (NOP
          DAC HSSW2
          JMS DMP
          BUF 7400

DHS3,     LAC (NOP
          DAC HSSW3
          JMP DMP1      /CHG. TO "JMS DMP" IF MPX3 ADDED
          BUF 11400

DMP,      0
          LAC I DMP     /GET BUFFER
          DAC DMP       /ADDR
          LAM -3        /SET LOOP
          DAC DCTR      /CNTR

```



```

DMP0,    LAC DMP      /INCR
          TAD (400)    /BUFFER
          DAC DMP      /ADDR
          RTR          /GET
          RTR
          RTR
          RTR
          AND (3       /BLOCK NO.
          DAC 10
          LAC I DMP     /PUT IT
          AND (77777)  /IN DATA
          TAD 10
          DAC I DMP
          LAC DMP       /WRITE
          IOF           /BLOCK
          JMS MGOAD     /OUT ONTO
          ION           /TAPE
          ISZ DCTR
          JMP DMP0
DMP1,    LAM-0         /DECR. MPX ON FLAG
          IOF
          TAD HSPFL
          DAC HSPFL
          ION
          ISZ HSPE      /ALL MULTIPLEXERS SERVICED
          JMP HISPD0    /NO
          EXIT          /YES-EXIT

HSPE,    0
DCTR,    0
T0FL,    0

START

```

CSEMPLEX15

/KEYBOARD INTERRUPT SERVICE ROUTINE

```

KEYB,      KRB
           SAD (322      /IS IT AN "R"
           JMP RELAY     /GO SET RELAY BUFFER
           SAD (324      /IS IT A "I"
           JMP READIN    /READ INITABLES
           SAD (310      /IS IT A "H"
           JMP HALT      /HALT DATA LOGGING
           SAD (314      /IS IT AN "L"
           JMP SCLKL     /SET CLKL
           SAD (315      /IS IT AN "M"
           JMP SCLKM     /SET CLKM
           SAD (330      /IS IT AN "X"
           JMP SSCN      /START SCN AGAIN
           SAD (306      /IS IT AN "F"
           JMP FIRE      /START BLOWDOWN
           JMS TYPOUT     /NONE OF THESE ERROR
           JMP . 2        /IN TYPED INPUT
           MESS9
           JMP CONTIN

RELAY,      LAS          /SET RELAY
           707501        /BUFFER TO ACS CONFIG
           JMP CONTIN

READIN,     LAS          /TURN ON
           DAC PRG 40    /START UP
           LAC (SSW      /SET INIT ENTRY
           DAC PRG 43    /TO READ IN TAPE
           JMP CONTIN

HALT,       LAC PERT 14  /SET CLOCK TO TIME
           DAC CLKM      / OF LAST PERIOD
           LAC PERT 15   /TERMINATION
           DAC CLKL
           LAM -77       /SET MAG TERM. CHAR.
           DAC TERM      /TO INTERMEDIATE
           JMP CONTIN

FIRE,       LAS          /TURN ON
           DAC PRG 40    /START UP
           LAC (SSW0     /SET INIT
           DAC PRG 43    /ENTRY
           JMP CONTIN

SCLKL,      LAS
           DAC CLKL      /SET CLOCK LEAST HALF
           JMP CONTIN    /TO ACS

```

```
SCLKM,  LAS
        DAC CLKM      /SET CLOCK MOST HALF
        JMP CONTIN    /TO ACS

SSCN,   LAS          /START
        DAC PROG 14   /SCANNER
        T0FF0
        LAC C30       /ELIMINATE BUFFER
        DAC SQCTR      /FOR HISPD MULTIPLEXER
        LAC SQINO
        DAC SQINI
        DAC SQMIN0
        JMP CLOK      /GO START CLOCK

MESS9,  TEXT
IPE
START
```

CSEMPLEX16

/ROUTINE TO CONVERT 3 DIGIT OCTAL # TO BAUDOT STORED IN ADDR

/CALL LIST

```

/      LAC #
/      JMS OCVT
/      ADDR
/      RETURN
OCVT,  4
      DAC ONUM      /SAVE NUMBER
      LAC I OCVT    /GET ADDR
      DAC OADR      /OF OUTPUT
      DZM I OADR    /AND ZERO IT
      ISZ OCVT      /INCR FOR RETURN
      LAM -2        /INIT LOOP
      DAC OCTR      /CNTR

```

```

OLP,   LAC ONUM      /GET NUMBER
      AND C7        /MASK
      IAD COTB      /CALC.
      DAC OPIR      /PNTR
      LAC I OPIR    /GET EQUIV. BAUDOT
      XOR I OADR    /ADDT
      DAC I OADR    /TO OUTPUT
      ISZ OCTR      /THRU
      SKP           /NO
      JMP I OCVT    /YES-RETURN
      LAC I OADR    /SHIFT OUTPUT
      RCR          RTR          RTR          RAR
      DAC I OADR    /6 RIGHT
      LAC ONUM      /SHIFT NUMBER
      RTR          RAR
      DAC ONUM      /RIGHT
      JMP OLP       /LOOP

```

```

OTB,   330000
      730000
      630000
      410000
      250000
      030000
      530000
      710000
      310000
      070000

```

```

OPIR,  0
OADR,  0
OCTR,  0
ONUM,  0
START

```

CSEMPLX19

/INITIALIZATION ROUTINE

INIT, DZM TELCTR /INITIALIZE CSEMPLX01 (01)

LAC TELMIN /01

DAC TELDUT /01

DAC TELIN /01 END

LAM-27 /02

DAC 10

LAC SQMIN

DAC 11

LAC (BUF-400

TAD (400

DAC I 11

LSZ 10

JMP *-3

LAC SQMIN0 /02

DAC SQINI /02

LAC (SQQUEUE+23 /02

DAC SQINO /02

LAC (14 /02

DAC SQCTR /02 END

DZM QMF /03

DZM MQCTR /03

LAC MQMIN /03

DAC MQINI /0

DAC MQINO /03 END

DZM TQCTR /05

LAC TQMIN /05

DAC TQINI /05

DAC TQINO /05 END

CAF /09

LAC (4

DAC BLOCK

LAC (010000

DAC UNIT

DZM MTU /09 END

DZM IPER /10

LAC (JMP EOP0

DAC EOP2

LAC (JMP SCN4B

DAC SCN4A

LAC (PERT

DAC IPERT

ADD (1

DAC IPERT+1

LAC (SCTM

DAC ISCTMI

LAM -0

DAC TERM

DAC BUFADI

TAD ICLTMI

DAC 10

```

INIT0,
    LAM -0
    TAD ISCI11
    DAC 11
    LAM -0
    TAD IPT11
    DAC 12
    LAM -6
    DAC ICLTM
    LAC ICLPI
    DAC I 12
    DZM I 11
    LAM -0
    DAC I 10
    DAC I 10
    ISZ ICLTM
    JMP INIT0
    DZM T0FL
    DZM SDCTR /RESET SCANNER ON FLAG
    DZM CLTM /TURN ON CLASS 1
    DZM CLTM 1 /10 END
    LAM -0 /12
    DAC COUNT
    DAC COUNT 1
    DAC COUNT 2
    DAC COUNT 3
    DAC COUNT 4
    DAC COUNT 5 /12 END
    LAC (JMP DHS0 /16
    DAC HSSW0
    DZM HSPFL
    LAC (JMP DHS1
    DAC HSSW1
    LAC (JMP DHS2
    DAC HSSW2
    LAC (JMP DHS3
    DAC HSSW3 /16 END
    LAC (PROG-1 /07
    DAC 17
    LAM -6
    DAC PROG1
INIT1,
    DZM I 17
    DZM I 17
    LAC I 17
    DAC I 17
    DZM I 17
    DZM I 17
    DZM I 17
    ISZ PROG1
    JMP INIT1 /07 END
    LAM -3 /09
    DAC 17

```

```

INIT2,    CLA          /DESELECT
          MMLC         /TAPE
          MMSE         /UNITS
          DELAY
          LAC UNIT     /SELECT
          AND MGSW
          ADD (014400
          DAC UNIT
          MMSE         /UNIT
INIT2A,   LAW 60       /MOVE
          MMLC         /REVERSE
          MMEF         /WAIT
          JMP 2-1       /FOR EDT
          LAW 41       /SEARCH
          MMLC         /FORWARD
          MMDF         /WAIT FOR
          JMP 2-1       /FIRST BLOCK
          LAW 43       /WRITE
          MMLC         /FORWARD
          MMDF         /WAIT
          JMP 2-1       /FOR DATA FLAG
          MMWR         /TRY TO WRITE
          MMDF         /WAIT FOR
          JMP 2-1       /FLAG
          MMEF         /REPEAT
          SKP          /ON ERROR
          JMP INIT2A
          ISZ 17
          JMP INIT2
          CLA          /DESELECT
          MMLC         /TAPE
          MMSE         /UNITS
          700501       /DROP ALL RELAYS
          DELAY
          LAC (BUF-1   /ZERO OUT BUFFER
          DAC 10
          DZM I 10
          LAC 10
          SAD (20000
          SKP
          JMP 2-4
          T0FFO        /DISABLE I0 INTERRUPT ENABLE FLIP FLOP
          JMP EXEC
DELAY=JMS .
          0
          LAW 0
          DAC 16
          ISZ 16
          JMP 2-1
          JMP I DELAY-JMS

HSPFL,    0           /HISPD UNIT RUNNING FLAG
START

```


Table B-V. Octal Dump of DECTAPE Data From Example

Table B-V. Octal Dump of DECTAPE Data From Example

000001							
400000	400210	400000	400000	400001	400210	400000	400000
400002	400210	400000	400000	400003	400210	400000	400000
400004	400210	400000	400000	400005	400010	400000	402200
400006	400210	400000	400000	400007	400410	400000	400000
400010	400010	400000	400000	402211	400210	400000	400000
400012	400210	400000	400000	400013	400210	400000	400000
ETC							

000002							
061620	062010	062201	063400	400421	400410	400001	400000
400022	400410	400001	400000	400023	400210	400001	400000
400024	400010	400001	400000	400025	400210	400001	400000
400026	400210	400001	400000	400027	400210	400001	400000
400030	400210	400001	400000	400031	400210	400001	400000
400032	400410	400001	400000	400033	400210	400001	400000
ETC							

000007							
400040	400041	400042	400043	400044	400045	400046	400047
400040	400041	400042	400043	400044	400045	400046	400047
400040	400041	400042	400043	400044	400045	400046	400047
400040	400041	400042	400043	400044	400045	400046	400047
400040	400041	400042	400043	400044	400045	400046	400047
400040	400041	400042	400043	400044	400045	400046	400047
ETC							

000010							
400041	400041	400042	400043	400044	400045	400046	400047
400040	400041	400042	400043	400044	400045	400046	400047
400040	400041	400042	400043	400044	400045	400046	400047
400040	400041	400042	400043	400044	400045	400046	400047
400040	400041	400042	400043	400044	400045	400046	400047
400040	400041	400042	400043	400044	400045	400046	400047
ETC							

000026							
400073	400071	400072	400073	400074	400075	400076	400077
400070	400071	400072	400073	400074	400075	400076	400077
400070	400071	400072	400073	400074	400075	400076	400077
400070	400071	400072	400073	400074	400075	400076	400077
400070	400071	400072	400073	400074	400075	400076	400077
400070	400071	400072	400073	400074	400075	400076	400077
ETC							

030027							
062020	061611	062400	063203	063635	062011	063200	064200
063020	060611	062601	062400	063035	061211	063001	063600
062620	061011	062402	062201	065035	061611	063602	063401
062420	061411	062403	062301	063635	062611	063203	062401
062620	061611	062404	062301	063635	062611	063004	062401
062420	061611	062205	062301	062035	063011	063005	063601
062420	062211	062406	062401	063035	063411	062606	062601
062420	062011	062207	062201	062035	063211	063407	063601
062420	062211	062410	061601	063635	063411	063210	062401
062220	062411	062211	061601	062435	063411	063411	063601
062020	062211	062412	062201	062235	063211	062612	063601
062220	062411	062013	062301	062435	063611	062613	063601
062220	062411	062214	061601	062435	063611	063414	063601
062220	062611	062415	062401	063635	063211	063215	062401
062220	062211	062216	062201	063635	063611	063216	062601
062220	062411	062017	062001	062635	063011	064017	063001
062220	062611	062420	062401	063035	063411	063020	062401
062220	062411	061621	062201	063435	063611	062621	063001
062020	062211	062222	062201	063035	063611	063222	062601
062620	063011	062623	062601	063035	063011	063023	062601
062420	062611	062424	062401	063035	063411	063024	062601
063020	063411	063025	062601	063035	063311	063025	062401
062620	062611	062426	062601	063420	063012	063400	063001
063635	063022	062200	062401	063220	062612	063201	063001
063420	063012	063202	063001	063220	063012	063203	063001
063220	062612	063004	062601	063220	062612	063005	062601
063435	062622	062601	062401	063220	063012	063206	062601
063020	062412	063007	062601	063220	063012	063210	063001
063220	062612	063211	063001	063420	063012	063412	063001
063035	062422	062602	062601	063220	062612	063013	062601
063220	063012	063414	063001	063420	062612	063415	063001
063220	062612	063016	063001	063220	063012	063217	062601

[illegible]

DISTRIBUTIONNo. of
Copies

1	<u>AEC Chicago Operations Office</u> D. M. Gardner
2	<u>AEC Chicago Patent Group</u> G. H. Lee R. K. Sharp
2	<u>AEC Idaho Operations Office</u> D. Williams
44	<u>AEC Library, Washington</u> <u>Advisory Committee on Reactor Safeguards</u> R. F. Fraley (17)
	<u>Division of Reactor Development and Technology</u> J. W. Crawford R. L. Ednie J. A. Lieberman (5) E. E. Sinclair M. J. Whitman
	<u>Division of Compliance</u> L. Kornblith, Jr.
	<u>AEC Energy Commission</u> <u>Division of Compliance, Region IV</u> P.O. Box 15266 Denver 15, Colorado 80215 John W. Flora
	<u>Division of Operational Safety</u> H. Gilbert
	<u>Division of Production</u> George B. Pleat
	<u>Division of Reactor Standards</u> M. Bolotsky E. G. Case (10) A. B. Holt

No. of
CopiesNaval Reactors BranchDivision of Reactor Development and Technology

R. S. Brodsky (2)

1	<u>AEC Oak Ridge Operations Office</u> W. L. Smalley
2	<u>AEC RDT Site Representative - PNL</u> P. G. Holsted
3	<u>AEC Richland Operations Office</u> C. L. Robinson A. S. Waterhouse Technical Information Library
1	<u>AEC San Francisco Operations Office</u> John B. Radcliffe
5	<u>Argonne National Laboratory</u> C. E. Dickerman S. Fistedis R. O. Ivins P. Lottes R. C. Vogel
1	<u>Atomic Energy Commission</u> <u>USAEC Technical Representative</u> Whiteshell Branch Pinawa, Manitoba, Canada Glenn D. Boyer
1	<u>Babcock & Wilcox Company</u> Robert Wascher
6	<u>Battelle Memorial Institute</u> A. R. Duffy D. L. Morrison S. Paprocki (2) D. N. Sunderman (2)
1	<u>Brookhaven National Laboratory</u> A. W. Castleman
265	<u>AEC Division of Technical Information Extension</u>

No. of
Copies

- 2 California, University of Institute of Engineering Research
 Berkeley 4, California 94704
 H. A. Johnson
 V. E. Schrock
- 1 Combustion Engineering, Inc.
 Nuclear Division
 M. F. Valerino
- 1 Douglas United Nuclear, Inc.
 Technical Information
- 1 du Pont Company, Aiken
 A. H. Peters
- 2 General Electric Company
 2151 South First Street
 San Jose, California 95125
 G. E. Wade
 S. Vandenberg
- 3 General Electric Company, San Jose (Trumbull)
 P. Bray
 M. Siegler
 W. A. Sutherland
- 1 Harvard Air Cleaning Laboratory
 Harvard University
 665 Huntington Avenue
 Boston, Massachusetts 02190
- 2 IIT Research Institute
 E. Gallagher
 T. A. Zaker
- 1 Los Alamos Scientific Laboratory
 J. H. Russel, K Div.
- 1 MPR Associates, Inc.
 815 Connecticut Avenue, N.W.
 Washington, D.C. 20006
 T. Rockwell

No. of
Copies

- 1 National Bureau of Standards (Koch)
 G. Muehlhause
- 1 Naval Ordnance Laboratory (Reil)
 James Proctor
- 1 North Carolina State University
 Department of Mechanical Engineering
 M. N. Ozisik
- 1 Nuclear Fuels Services
 West Valley, New York 14171
 Russel P. Wischow
- 7 Union Carbide Corporation (ORNL-Y-12)
 Nuclear Safety Information Center
 Joel Buchanan
- Chemical Technology Division
 R. Blanco
 D. Ferguson
 W. B. Cottrell (4)
- 9 Phillips Petroleum Company
 S. Forbes (2)
 Curt Haire
 N. K. Sowards
 T. R. Wilson (2)
- Water Reactor Safety Program Office
 Frank Schroeder (3)
- 2 TRW Space Technology Laboratories
 TRW Systems Group
 D. B. Langmuir
 S. M. Zivi
- 3 Westinghouse Electric Corp.
 E. Beckjord
 D. Fletcher
 R. A. Wiesemann

No. of
Copies

31	<u>Battelle-Northwest</u>
	R. T. Allemann
	J. M. Batch
	J. Campbell
	L. D. Coffin
	G. M. Dalen
	R. Y. Dean
	J. L. Deichman
	G. E. Driver
	C. E. Linderoth
	R. Nightingale (2)
	G. W. Riedeman
	G. J. Rogers
	J. T. Russell
	L. Schwendiman
	J. C. Spanner
	W. L. Thorne
	R. A. Walker
	W. R. Weissenberger
	E. L. Wells
	N. P. Wilburn (3)
	M. E. Witherspoon
	Technical Information Files (5)
	Technical Publications (2)