22
4-30-75
cp to NTIS
20
S-3578

SDS-910 UTILITIES MANUAL

MASTER

Leonard C. Moon

and Thomas G. Pinter

AMES LABORATORY, USERDA

IOWA STATE UNIVERSITY

AMES, IOWA

Date Transmitted: March 1975

PREPARED FOR THE U. S. ENERGY RESEARCH AND DEVELOPMENT ADMINISTRATION
UNDER CONTRACT W-7405-eng-82

DISTRIBUTION OF THIS DOCUMENT UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

SDS-910 UTILITIES MANUAL

Leonard C. Moon and Thomas G. Pinter

Ames Laboratory, ERDA
Iowa State University
Ames, Iowa 50010

Date Transmitted: March 1975

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research and Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

PREPARED FOR THE U. S. ENERGY RESEARCH AND
DEVELOPMENT ADMINISTRATION UNDER CONTRACT W-7405-eng-82

MASTER

DISTRIBUTION OF THIS DOCUMENT UNLIMITED

—NOTICE—

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research and Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

Available from: National Technical Information Service
U. S. Department of Commerce
P.O. Box 1553
Springfield, VA 22161

Price: Microfiche \$2.25
 Paper Copy \$4.25

TABLE OF CONTENTS

	PAGE
ABSTRACT	iv
INTRODUCTION	1
CARD READER ROUTINE	2
PDP-15 LOADER ROUTINE	17
SDS-910 SYSTEM AND PROGRAM LOADER ROUTINES	21
RAID--THE SDS-910 DEBUG PACKAGE	41

ABSTRACT

This report describes the various utility programs developed by the software systems group of the Ames Laboratory, for the SDS-910 computer.

Distribution List

L. Moon	12
T. Pinter	12
OR	27
ERDA Chicago Patent Office	1
Ames Laboratory Library	8
	—
Total	60

INTRODUCTION:

The various utility programs described in this report were developed by the software systems group of the Ames Laboratory, for the two SDS-910 computers. The differences between the two computers are hardware differences. The software was made to be as transparent as possible to the user. Program differences are noted in the sequence numbers of the listings. The "J" stands for those programs that operate on Jacobson's SDS-910 -- PDP-15 computer system. The "R" denotes those programs that operate on the Reactor SDS-910 -- PDP-15 computer system.

Even though these utilities are for the SDS-910, some of them operate on the PDP-15. This is because the SDS-910 does not have a card reader, magnetic tape drive, or magnetic disk as peripherals.

CARD READER ROUTINE:

This routine is used to build a program data set which will be executed in the SDS-910.

HOW TO USE:

1. On the PDP-15 computer use DISKORG to create a norm data set with record length = 44. The number of sectors needed is figured by counting the number of cards of the SDS-910 program to be put out on disk and dividing by 5. Add 1 to the number of sectors if there is any fraction left over from the division.

On any PDP-15 teletype type in a

CNTL A

System responds REQ, INQ, OR ZAP?

Reply REQ CNTL D

System responds TYPE IN USER LIBRARY NAME?

Reply DISK CNTL D

System responds TYPE IN TASK NAME?

Reply DISKORG CNTL D

System responds USER ID IS ##### CORE ADDRESS IS #####
TASK DISPATCHABLE

DISKORG asks TYPE IN DATE-YYMMDD?

Reply RESPOND WITH CURRENT YEAR, MONTH, DAY CNTL D

DISKORG asks TYPE IN OPCODE?

Reply ADS CNTL D

DISKORG asks DIRECTORY NAME?

Reply RESPOND WITH YOUR LIBRARY NAME CNTL D

DISKORG asks	DATA SET NAME?
Reply	RESPOND WITH THE NAME OF YOUR SDS-910 PROGRAM CNTL D
DISKORG asks	DATA SET TYPE?
Reply	NORM CNTL D
DISKORG asks	RECORD LENGTH?
Reply	44 CNTL D
DISKORG asks	NUMBER OF SECTORS?
Reply	RESPOND WITH NUMBER CALCULATED ABOVE CNTL D
DISKORG asks	CREATOR'S NAME?
Reply	YOUR INITIALS CNTL D
DISKORG asks	UPDATE?
Reply	YES CNTL D
DISKORG asks	GET INPUT FROM CARD READER?
Reply	NO CNTL D
DISKORG asks	TYPE IN OPCODE?
Reply	If you want to create more data sets reply ADS CNTL D and go through the sequence again. If not reply CNTL D.

DISKORG should terminate normally at this point.

2. Use CARDREAD routine to read in SDS-910 program cards through the PDP-15 card reader.

On a PDP-15 teletype type in a

CNTL A

System responds	REQ, INQ, OR ZAP?
Reply	REQ CNTL D

System responds TYPE IN USER LIBRARY NAME?

Reply Respond with name of library that the SDS-910
program will reside in CNTL D

System responds TYPE IN TASK NAME?

Reply CARDREAD CNTL D

System responds USER ID IS ##### CORE ADDRESS IS #####
TASK DISPATCHABLE

CARDREAD asks DATA SET NAME?

Reply Reply with above created data set name CNTL D

Put SDS-910 program cards in the PDP-15 card reader hopper.

Push the reset and start buttons. CARDREAD will then read in the cards.

If any errors are encountered while reading the cards the program will respond with the appropriate error message. If the READ CHECK light comes on on the PDP-15 card reader, simply take all of the cards out of the stacker and put them on the front of the hopper again. The program will respond with

DATA SET NAME?

just reply with the appropriate program data set name, CNTL D. On successful completion the program will ask

DATA SET NAME?

If you have more programs to load onto the PDP-15 disk reply with the next program name, CNTL D. If not reply CNTL D.

The program should end normally (completion code 000000).

CARDREAD: PROC;		RCRD0010
/* THIS PROCEDURE READS INPUT CARDS IN COLUMN BINARY FOR		RCRC0020
THE 910 SYSTEM TO BE LOADED ACROSS THE INTERFACE.		RCRC0030
/*		RCRC0040
*/		RCRD0050
DCL J FIXED;		RCRD0060
DCL BUFSTP FIXED, /* INDICATES THE LAST WORD IN THE BUFFER.	*/RCRC0070	
CDREG FIXED, /* CONTAINS STATUS WORD FROM THE EXPERIMENT.	*/RCRD0080	
COLCT FIXED, /* COUNTS THE COLUMNS OF THE INPUT CARD.	*/RCRD0090	
D1 FIXED,D2 FIXED, /* POINTERS FOR BUFFERS BEING USED.	*/RCRD0100	
I FIXED, /* LCOP PARAMETER.	*/RCRD0110	
IB FIXED, /* POINTER FOR PACKED ARRAY.	*/RCRC0120	
ICL FIXED, /* PCINTER FOR ARRAY CL.	*/RCRD0130	
IL FIXED, /* STARTING POINTER FOR CARD COLUMNS.	*/RCRC0140	
K FIXED, /* BUFFER POINTER.	*/RCRD0150	
LIM FIXED, /* BUFFER MARKER.	*/RCRD0160	
MSG# FIXED, /* PCINTER TO MESSAGE ARRAY FOR OUTPUT.	*/RCRC0170	
RM FIXED, /* CCUNTS NUMBER OF LEADING NON-ZERO WORDS.	*/RCRD0180	
TEMP FIXED, /* TEMPORARY STORAGE LOCATION.	*/RCRC0190	
TYPE FIXED; /* INDICATES THE PACKING TYPE.	*/RCRD0200	
/*		RCRC0210
*/		RCRD0220
DCL CD910 FILE OUTPUT,		RCRD0230
/* DISK FILE FOR STORING 910 CODE.	*/RCRC0240	
CR15 FILE EXPT; /* INPUT FILE FOR CARD READER.	*/RCRD0250	
/*		RCRC0260
*/		RCRD0270
DCL B(44) FIXED, /* CCONTAINS PACKED CARD COLUMNS.	*/RCRD0280	
CL(66) FIXED, /* ARRAY FOR STORAGE OF THE CARD COLUMNS.	*/RCRC0290	
PT(4) FIXED INIT(28,38,9,32);	RCRC0300	
/* MESSAGE LENGTH POINTER.	*/RCRC0310	
/*		RCRD0320
*/		RCRD0330
DCL DUMMY CHAR(9), /* CCONTAINS DATA SET NAME OF OUTPUT FILE.	*/RCRC0340	
MSG(4) CHAR(38) INIT(	RCRD0350	
'HCPPR EMPTY, MISSING CARDS. ',	RCRC0360	
/* 2 */ 'COLUMNS WERE MISSING ON THE LAST CARD.',	RCRC0370	

'EAC CARD.	'	RCRD0380
/* 4 */ 'NORMAL END OF JOB, ALL CARDS IN.	'	RCRC0390
);	/* MESSAGE STRING LENGTH.	*/RCRD0400
DCL BUF_FULL BIT(1);	/* SET TRUE WHEN OUTPUT BUFFER IS FULL.	*/RCRD0410
DCL BUF_EMP EVENT;	/* EVENT VARIABLE TO DENOTE THAT BUFFER	RCRC0420
	HAS BEEN WRITTEN ONTO DISK.	*/RCRD0430
DCL CARD_DONE EVENT;	/* EVENT VARIABLE SIGNIFYING THAT A CARD	RCRC0440
	HAS BEEN READ.	*/RCRD0450
/*		RCRC0460
DCL LOAD ENTRY ((*) FIXED, (*) FIXED);		RCRC0470
*/		RCRD0480
COMPLETION (CARD_DONE)='0'B;		RCRC0490
OPEN FILE (CR15);		RCRD0500
/* THIS IS THE ON-BLOCK FOR THE CARD READER INTERRUPTS.		RCRD0510
*/		RCRC0520
CN EXPT (CR15) BEGIN;		RCRD0530
DCL DATA FIXED;	/* INPUT FROM CARD COLUMN.	*/RCRD0540
CDREG=ONCODE;		RCRC0550
/* GET THE STATUS WORD FOR CHECKING.		RCRD0560
CLEAR INTERRUPT AND STATUS.		RCRD0570
*/		RCRD0580
TEMP=(0+20)+EXEC_CRCS;		RCRD0590
/* CHECK BIT 16 TO SEE IF THE CARD IS FINISHED.		RCRD0600
*/		RCRD0610
IF CDREG & '0000000000000000010'B		RCRC0620
THEN DO;		RCRD0630
/* CHECK FOR MISSING COLUMNS.		RCRC0640
*/		RCRD0650
/*		RCRD0660
MISSING COLUMNS.		RCRC0670
*/		RCRD0680
IF COLCT=80 THEN MSG#=2;		RCRC0690
COMPLETION (CARD_DONE)='1'B; GO TO INT_EXIT;		RCRC0700
END;		RCRD0710
/* TEST FOR CARD DONE INTERRUPT MISSED */		RCRC0720
IF COLCT>=80 THEN DO;		RCRD0730
COMPLETION(CARD_DONE)='1'B; GO TO INT_EXIT; END;		RCRD0740

/*	BIT 7 TESTS FOR HOPPER EMPTY.	RCRD0750
*/		RCRD0760
	IF CDREG&'00000001000000000'B	RCRD0770
	THEN DO;	RCRD0780
	IF MSG#=4 THEN; ELSE MSG#=1;	RCRD0790
	COMPLETION (CARD_DONE)='1'B;	RCRD0800
	GO TO INT_EXIT;	RCRD0810
	END;	RCRD0820
/*	CHECK FOR FOLLOWING ERRORS:	RCRD0830
	BIT 4-PHOTO ERROR	RCRD0840
	BIT 5-MOTION ERROR	RCRD0850
	BIT 6-PICK ERROR	RCRD0860
	BIT 8-DATA MISSED	RCRD0870
	BIT 9-TROUBLE	RCRD0880
*/		RCRD0890
	IF CDREG &'000011101100000000'B	RCRD0900
	THEN DO;	RCRD0910
/*	BAD CARD.	RCRD0920
*/		RCRD0930
	MSG#=3; GO TO INT_EXIT; END;	RCRD0940
/*	CHECK BIT 17 TO SEE IF DATA READY.	RCRD0950
*/		RCRD0960
	IF CDREG & 1	RCRD0970
	THEN ;	RCRD0980
	ELSE DO; MSG#=3; GO TO INT_EXIT; END;	RCRD0990
/*	LOAD DATA.	RCRD1000
*/		RCRD1010
	READ FILE(CR15) INTO (DATA) KEY (0);	RCRD1020
/*	ONLY TWELVE LEAST SIGNIFICANT BITS ARE NECESSARY.	RCRD1030
*/		RCRD1040
	DATA=DATA & '0000001111111111'B;	RCRD1050
/*	INCREMENT COLUMN COUNTER.	RCRD1060
*/		RCRD1070
	COLCT=COLCT+1;	RCRD1080
/*	NO DATA STORED IN FIRST SIX COLUMNS OR LAST SIX COLUMNS.	RCRD1090
*/		RCRD1100
	IF COLCT<7 COLCT>72	RCRD1110

LOOP:	RCRD1490
IF COLCT<9 THEN GO TO LOOP;	RCRC1500
IF CL(1)=2860 & CL(2)=2860	RCRD1510
THEN DO;	RCRC1520
LCOP1: IF COLCT<80 THEN GO TO LOOP1; /* WAIT UNTIL ENTIRE CARD IN*/	RCRD1530
BUF_FULL='1'B; MSG#=4; CALL PACK;	RCRD1540
WRITE FILE (CD910) FROM (B) EVENT(BUF_EMP);	RCRC1550
WAIT(BUF_EMP);	RCRD1560
CLOSE FILE (CD910);	RCRC1570
GO TO GET_NEXT;	RCRD1580
END;	RCRD1590
ELSE DO;	RCRC1600
CALL PACK;	RCRD1610
/* PRINT OUT ERROR MESSAGE AND TERMINATE READ.	RCRC1620
*/	RCRD1630
FINISH:	RCRD1640
WAIT (CARD_DONE);	RCRC1650
IF MSG#	RCRD1660
THEN DO;	RCRC1670
PUT SKIP LIST(MSG(MSG#));	RCRD1680
GO TO END;	RCRD1690
END;	RCRC1700
END;	RCRD1710
WRITE FILE (CD910) FROM (B) EVENT(BUF_EMP);	RCRC1720
WAIT (BUF_EMP);	RCRD1730
GO TO START;	RCRC1740
/*	RCRC1750
*/	RCRD1760
PACK: /* THIS PROCEDURE PACKS THE 12 BIT INPUT WORDS INTO 18	RCRC1770
BIT 15 WORDS FOR STORING ON DISK.	*/RCRD1780
PROC;	RCRD1790
IB=1;	RCRC1800
DO I= 1 TO 66;	RCRD1810
PAUSE:	RCRC1820
IF COLCT<=1 THEN GO TO PAUSE;	RCRD1830
IF TYPE-1	RCRD1840
THEN IF TYPE-2	RCRC1850

THEN DO; /* TYPE=3	*/RCRD1860
B(IB)=B(IB) ((CL(I))&'0000001111111111'B);	RCRC1870
IB=IB+1; TYPE=1;	RCRD1880
END;	RCRC1890
ELSE DO; /* TYPE=2	*/RCRC1900
B(IB)=B(IB) ((CL(I)/64)&63);	RCRD1910
IB=IB+1;	RCRC1920
B(IB)=(CL(I)*4096)&'111111000000000000'B;	RCRC1930
TYPE=3;	RCRD1940
END;	RCRC1950
ELSE DO; /* TYPE=1	*/RCRD1960
B(IB)=((CL(I)*64)&'111111111111000000'B);	RCRC1970
TYPE=2;	RCRC1980
END;	RCRD1990
END;	RCRC2000
END;	RCRD2010
END PACK;	RCRD2020
/*	RCRC2030
*/	RCRD2040
END;	RCRC2050
WAIT(BUF_EMP);	RCRC2060
CLOSE FILE (CD910);	RCRD2070
GO TO GET_NEXT;	RCRD2080
END_RUN:	RCRD2090
CLOSE FILE (CR15);	RCRC2100
END CARDREAD;	RCRC2110

CARDREAD: PROC;		JCRD0010
/* THIS PROCEDURE READS INPUT CARDS IN COLUMN BINARY FOR		JCRC0020
THE 910 SYSTEM TO BE LOADED ACROSS THE INTERFACE.		JCRC0030
/*		JCRD0040
*/		JCRC0050
DCL J FIXED;		JCRD0060
DCL BUFSTP FIXED, /* INDICATES THE LAST WORD IN THE BUFFER.	*/	JCRC0070
CDREG FIXED, /* CONTAINS STATUS WORD FROM THE EXPERIMENT.	*/	JCRC0080
COLCT FIXED, /* CCUNTS THE COLUMNS OF THE INPUT CARD.	*/	JCRD0090
D1 FIXED,D2 FIXED, /* POINTERS FOR BUFFERS BEING USED.	*/	JCRC0100
I FIXED, /* LCOP PARAMETER.	*/	JCRD0110
IB FIXED, /* POINTER FOR PACKED ARRAY.	*/	JCRC0120
ICL FIXED, /* POINTER FOR ARRAY CL.	*/	JCRD0130
II FIXED, /* STARTING POINTER FOR CARD COLUMNS.	*/	JCRD0140
K FIXED, /* BUFFER POINTER.	*/	JCRC0150
LIM FIXED, /* BUFFER MARKER.	*/	JCRD0160
MSG# FIXED, /* PCINTER TO MESSAGE ARRAY FOR OUTPUT.	*/	JCRC0170
RM FIXED, /* COUNTS NUMBER OF LEADING NON-ZERO WORDS.	*/	JCRD0180
TEMP FIXED, /* TEMPORARY STORAGE LOCATION.	*/	JCRC0190
TYPE FIXED; /* INDICATES THE PACKING TYPE.	*/	JCRD0200
/*		JCRD0210
*/		JCRC0220
DCL CD910 FILE OUTPUT,		JCRD0230
/* DISK FILE FOR STORING 910 CODE.	*/	JCRC0240
CRO3B FILE EXPT;		JCRD0250
/*		JCRD0260
*/		JCRC0270
DCL E(44) FIXED, /* CONTAINS PACKED CARD COLUMNS.	*/	JCRD0280
CL(66) FIXED, /* ARRAY FOR STORAGE OF THE CARD COLUMNS.	*/	JCRC0290
PT(4) FIXED INIT(28,38,9,32);		JCRD0300
/* MESSAGE LENGTH POINTER.	*/	JCRC0310
/*		JCRD0320
*/		JCRD0330
DCL DUMMY CHAR(9), /* CONTAINS DATA SET NAME OF OUTPUT FILE.	*/	JCRC0340
MSG(4) CHAR(38) INIT(		JCRD0350
'HOPPER EMPTY, MISSING CARDS.',		JCRC0360
/* 2 */ 'COLUMNS WERE MISSING ON THE LAST CARD.',		JCRD0370

'EAD CARD.'	JCRD0380
/* 4 */ 'NORMAL END OF JOB, ALL CARDS IN.'	JCRC0390
); /* MESSAGE STRING LENGTH.	*/JCRD0400
DCL BUF_FULL BIT(1);/* SET TRUE WHEN OUTPUT BUFFER IS FULL.	*/JCRD0410
DCL BUF_EMP EVENT; /* EVENT VARIABLE TO DENOTE THAT BUFFER	JCRD0420
HAS BEEN WRITTEN ONTO DISK.	*/JCRD0430
DCL CARD_DONE EVENT; /* EVENT VARIABLE SIGNIFYING THAT A CARD	JCRC0440
HAS BEEN READ.	*/JCRD0450
/*	JCRC0460
DCL LOAD ENTRY ((*) FIXED, (*) FIXED);	JCRC0470
*/	JCRD0480
COMPLETION (CARD_DONE)='0'B;	JCRC0490
OPEN FILE(CR03B);	JCRD0500
/* THIS IS THE ON-BLOCK FOR THE CARD READER INTERRUPTS.	JCRD0510
*/	JCRC0520
ON EXPT(CR03B) BEGIN;	JCRD0530
DCL DATA FIXED; /* INPUT FROM CARD COLUMN.	*/JCRD0540
CDREG=ONCODE;	JCRC0550
/* GET THE STATUS WORD FOR CHECKING.	JCRD0560
CLEAR INTERRUPT AND STATUS.	JCRD0570
*/	JCRD0580
TEMP=EXEC_CRCS;	JCRD0590
/* CHECK BIT 16 TO SEE IF THE CARD IS FINISHED.	JCRD0600
*/	JCRD0610
IF CDREG & '000000000000000010'B	JCRC0620
THEN DO;	JCRD0630
/* CHECK FOR MISSING COLUMNS.	JCRC0640
*/	JCRD0650
/*	JCRD0660
MISSING COLUMNS.	JCRC0670
*/	JCRD0680
IF COLCT=80 THEN MSG#=2;	JCRC0690
COMPLETION (CARD_DONE)='1'B; GO TO INT_EXIT;	JCRD0700
END;	JCRD0710
/* TEST FOR CARD DONE INTERRUPT MISSED */	JCRC0720
IF COLCT>=80 THEN DO;	JCRD0730
COMPLETION(CARD_DONE)='1'B; GO TO INT_EXIT; END;	JCRC0740

/* HOPPER EMPTY CHECK */	JCRD0750
IF CDREG&'010J00000000000000'B THEN	JCRD0760
DO;	JCRD0770
IF MSG#=4 THEN; ELSE MSG#=1;	JCRD0780
COMPLETION (CARD_DONE)='1'B;	JCRD0790
GO TO INT_EXIT;	JCRD0800
END;	JCRD0810
/* CHECK THE FOLLOWING:	JCRD0820
0 NOT READY	JCRD0830
2 NO PASS (CARD)	JCRD0840
3 BAD DATA */	JCRD0850
IF CDREG&'101100000000000000'B THEN	JCRD0860
DO;	JCRD0870
/* EAC CARD.	JCRD0880
*/	JCRD0890
MSG#=3; GO TO INT_EXIT; END;	JCRD0900
/* CHECK BIT 17 TO SEE IF DATA READY.	JCRD0910
*/	JCRD0920
IF CDREG & 1	JCRD0930
THEN ;	JCRD0940
ELSE DO; MSG#=3; GO TO INT_EXIT; END;	JCRD0950
/* LOAD DATA.	JCRD0960
*/	JCRD0970
/* ONLY TWELVE LEAST SIGNIFICANT BITS ARE NECESSARY.	JCRD0980
*/	JCRD0990
DATA=(CDREG/4)& '000000111111111111'B;	JCRD1000
/* INCREMENT COLUMN COUNTER.	JCRD1010
*/	JCRD1020
COLCT=COLCT+1;	JCRD1030
/* NO DATA STORED IN FIRST SIX COLUMNS OR LAST SIX COLUMNS.	JCRD1040
*/	JCRD1050
IF COLCT<7 COLCT>72	JCRD1060
THEN GO TO INT_EXIT;	JCRD1070
ICL=ICL+1; CL(ICL)=DATA;	JCRD1080
INT_EXIT:	JCRD1090
/* END OF THE ON-BLOCK.	JCRD1100
*/	JCRD1110

END;	JCRD1120
/* THIS ROUTINE INITIALIZES VARIABLES AND READS A CARD.	*/JCRD1130
GET_NEXT:	JCRD1140
PUT LIST('DATA SET NAME');	JCRD1150
GET LIST (DUMMY);	JCRD1160
IF DUMMY=' ' THEN GO TO END_RUN;	JCRD1170
OPEN FILE (CD910) OUTPUT TITLE (DUMMY);	JCRD1180
TYPE=1; MSG#=0;	JCRD1190
BUF_FULL='0'B; BUFSTP=C; LIM=0;	JCRD1200
START:	JCRD1210
CDREG=0;	JCRD1220
COLCT=0; ICL=0;	JCRD1230
WRITE FILE (CRC3B) FROM(CDREG) KEYFROM(0);	JCRD1240
COMPLETION (CARD_DONE)='0'B;	JCRD1250
LCCP:	JCRD1260
IF COLCT<9 THEN GO TO LOOP;	JCRD1270
IF CL(1)=2860 & CL(2)=2860	JCRD1280
THEN DO;	JCRD1290
LOOP1: IF COLCT<80 THEN GO TO LOOP1; /* WAIT UNTIL ENTIRE CARD IN*/	JCRD1300
BUF_FULL='1'B; MSG#=4; CALL PACK;	JCRD1310
WRITE FILE (CD910) FROM (B) EVENT(BUF_EMP);	JCRD1320
WAIT (BUF_EMP);	JCRD1330
CLCSE FILE (CD910);	JCRD1340
GO TO GET_NEXT;	JCRD1350
END;	JCRD1360
ELSE DO;	JCRD1370
CALL PACK;	JCRD1380
/* PRINT OUT ERROR MESSAGE AND TERMINATE READ.	JCRD1390
*/	JCRD1400
FINISH:	JCRD1410
WAIT (CARD_DONE);	JCRD1420
IF MSG#	JCRD1430
THEN DO;	JCRD1440
PUT SKIP LIST(MSG(MSG#));	JCRD1450
GO TO END;	JCRD1460
END;	JCRD1470
END;	JCRD1480

WRITE FILE (CD910) FROM (B) EVENT(BUF_EMP);	JCRD1490
WAIT (BUF_EMP);	JCRC1500
GO TO START;	JCRD1510
/*	JCRC1520
*/	JCRD1530
PACK: /* THIS PROCEDURE PACKS THE 12 BIT INPUT WORDS INTO 18	JCRC1540
BIT 15 WORDS FOR STORING ON DISK.	*/JCRC1550
PROC;	JCRD1560
IB=1;	JCRD1570
DO I= 1 TO 66;	JCRD1580
PAUSE:	JCRC1590
IF COLCT<=I THEN GO TO PAUSE;	JCRC1600
IF TYPE=1	JCRD1610
THEN IF TYPE=2	JCRD1620
THEN DO; /* TYPE=3	*/JCRD1630
B(IB)=B(IB) ((CL(I))&'0000001111111111'B);	JCRC1640
IB=IB+1; TYPE=1;	JCRC1650
END;	JCRD1660
ELSE DO; /* TYPE=2	*/JCRC1670
B(IB)=B(IB) ((CL(I)/64)&63);	JCRD1680
IB=IB+1;	JCRC1690
B(IB)=(CL(I)*4096)&'111111000000000000'B;	JCRC1700
TYPE=3;	JCRD1710
END;	JCRC1720
ELSE DO; /* TYPE=1	*/JCRD1730
B(IB)=((CL(I)*64)&'111111111111000000'B);	JCRD1740
TYPE=2;	JCRC1750
END;	JCRD1760
END;	JCRD1770
END:	JCRD1780
END PACK;	JCRD1790
/*	JCRC1800
*/	JCRD1810
END:	JCRC1820
WAIT(BUF_EMP);	JCRD1830
CLOSE FILE (CD910);	JCRD1840
GO TO GET_NEXT;	JCRD1850

END_RUN:
CLOSE FILE (CRC3B);
END CARDREAD;

JCRC1860
JCRD1870
JCRD1880

PDP-15 LOADER ROUTINE:

This routine (in conjunction with the SDS-910 System and Program loader, see page 21 in this manual) loads SDS-910 programs from the PDP-15 disk. For information on how to put a SDS-910 program on the PDP-15 disk see CARD READ ROUTINES page 2 in this manual.

If the user's program has a branch on the end card the loader will cause a branch to that location on the end of the transfer. This option should only be used by the SDS-910 system.

HOW TO USE:

On a PDP-15 teletype press CNTL A

System responds REQ, INQ, OR ZAP?

Reply REQ CNTL D

System responds TYPE IN USER LIBRARY NAME?

Reply REPLY WITH LIBRARY WHERE THE SDS-910 PROGRAM
RESIDES CNTL D

System responds TYPE IN TASK NAME?

Reply REPLY WITH LOADER CNTL D

System responds USER ID ##### CORE ADDRESS IS #####
TASK DISPATCHABLE

The loader routine on the PDP-15 side will now ask:

DATA SET NAME?

Reply with name of data set which contains the users program to be loaded. The loader will now load the above data set into the SDS-910.

Upon completion it will ask

NORMAL END, NEXT DATA SET NAME?

If there are more programs to be loaded across the interface specify the next one at this time. Otherwise, reply with a CNTL D. The loader routine should end normally.

LOADER: PROC;	LDR00010
/* THIS PROCEDURE READS 910 CODE FROM DISK AND SENDS IT TO	LDR00020
THE 910 TO LOAD THE SYSTEM AND USER CODES.	LDR00030
*/	LDR00040
DCL CD910 FILE, /* INPUT FILE CONTAINING THE 910 CODE.	*/LDR00050
LD910 FILE EXPT; /* OUTPUT FILE TO THE 910 THRU THE INTERFACE.	*/LDR00060
/*	LDR00070
*/	LDR00080
/* INPUT BUFFER FOR CD910.	*/LDR00090
DCL BUFIN (44) FIXED;	LDR00100
DCL READ_DONE EVENT;	LDR00110
DCL DUMMY CHAR(9); /* CCNTAINS DATA SET NAME OF THE INPUT FILE.	*/LDR00120
/*	LDR00130
*/	LDR00140
DCL ENDLOAD(4) BIT(18) INIT('101100101100101100'B,	LDR00150
'10110000000000000000'B,'00000000000000000000'B,	LDR00160
'00000000000000000000'B);	LDR00170
DCL J FIXED;	LDR00180
/*	LDR00190
*/	LDR00200
ON ENDFILE (CD910) GO TO L3;	LDR00210
/*	LDR00220
*/	LDR00230
OPEN FILE (LD910);	LDR00240
FUT SKIP;	LDR00250
L05:	LDR00260
PUT LIST(' DATA SET NAME');	LDR00270
GET LIST (DUMMY);	LDR00280
IF DUMMY=' ' THEN GO TO L4;	LDR00290
OPEN FILE (CD910) INPUT TITLE (DUMMY);	LDR00300
L2:	LDR00310
/* READ IN ONE RECCRD FROM DISK.	LDR00320
*/	LDR00330
READ FILE (CD910) INTO (BUFIN);	LDR00340
IF BUFIN(1)='101100101100101100'B	LDR00350
THEN DC; DO J=1 TO 4;	LDR00360
ENDLOAD(J)=BUFIN(J); END;	LDR00370

WRITE FILE (LD910) FROM (ENDLOAD) KEYFROM (1) EVENT	LDR00380
(READ_DONE);	LDR00390
WAIT (READ_DONE);	LDR00400
GO TO L3; END;	LDR00410
/* SEND CODE ACROSS THE INTERFACE TO THE 910	LDR00420
*/	LDR00430
WRITE FILE (LD910) FROM (BUFIN) KEYFROM (1) EVENT(READ_DONE);	LDR00440
WAIT(READ_DONE);	LDR00450
GO TO L2;	LDR00460
L3: CLOSE FILE(CD910);	LDR00470
PUT SKIP LIST('NORMAL END, NEXT');	LDR00480
GO TO L05;	LDR00490
L4:	LDR00500
CLOSE FILE(LD910);	LDR00510
END LOADER;	LDR00520

SDS-910 SYSTEM AND PROGRAM LOADER ROUTINES:

These routines load SDS-910 program into the SDS-910 computer when used in conjunction with the PDP-15 LOADER explained on page 17 of this manual. There are two modes of operation;

1. initial (system and program) load,
2. load under system control.

When the initial load mode is chosen, the program zeroes core, sets-up the interrupt vectors, and loads the system and users programs into the SDS-910. If a load under system control is done the program just loads the user program into the SDS-910. The program is on both paper tape and PDP-15 disk.

HOW TO USE INITIAL SYSTEM LOADER (IPL):

1. Read in the SDS-910 system loader paper tape on the SDS-910.
 - a. Put RUN/IDLE switch to idle.
 - b. Turn register select switch to A register. Push start button.
 - c. Turn register select switch to C register. Push start button.
 - d. Turn register select switch to B register. Push start button.
 - e. Turn register select switch to X register. Push start button.
 - f. Turn register select switch to C register.
 - g. Put paper tape loader in paper tape reader. Make sure the load run knob on the paper tape reader is in LOAD when loading and RUN when you move to the next step.

- h. Turn power on to paper tape reader.
- i. Move RUN/IDLE switch to RUN.
- j. Lift FILL switch.

Paper tape should read in.

- 1. Turn power off to paper tape reader.
 - m. Re-wind paper tape.
2. On the PDP-15 side:
- a. Zero PDP-15 core. See systems operation book.
 - b. Set address register to 17720₈ by using the console switches. Read in PDP-15 system using the system loader tape.
 - c. On the PDP-15 console:
 - 1. J100 CR
 - 2. Press continue on the console.
 - d. CNTL A, the system will respond with
REQ, INQ, OR ZAP?
 - e. Reply REQ CNTL D, the system will respond with
TYPE IN USER LIBRARY NAME?
 - f. Reply with your SDS-910 program library name CNTL D.
The system will respond with
TYPE IN TASK NAME?
 - g. Reply LOADER CNTL D.
The system will respond
USER ID IS ##### CORE ADDRESS IS #####
TASK DISPATCHABLE

- h. The loader routine on the PDP-15 side will ask:

DATA SET NAME?

Reply OS910 CNTL D.

- i. The loader will load the system into the SDS-910

and then return with:

NORMAL END, NEXT DATA SET NAME?

Reply with your program data set name, CNTL D.

- j. The loader will load that program into the SDS-910

and return again with:

NORMAL END, NEXT DATA SET NAME?

Reply with your next program (if you have one) CNTL D.

If you don't have any more programs then reply CNTL D.

This will cause the loader in both computers to end normally.

Now you can start any program by entering CNTL A. The system will respond again

REQ, INQ, OR ZAP?

Reply REQ CNTL D and reply to the rest of the questions with the library and data set that you want to execute.

TO USE THE LOADER UNDER SYSTEM CONTROL:

On the PDP-15 side:

1. On any teletype press CNTL A.

The system will respond

REQ, INQ OR ZAP?

Reply REQ CNTL D.

The system will respond

TYPE IN USER LIBRARY NAME?

Reply with the library name that the SDS-910 program

resides in, CNTL D. The system will respond with

TYPE IN TASK NAME?

Reply LOADER CNTL D. The system will respond with

USER ID IS ##### CORE ADDRESS IS #####
TASK DISPATCHABLE

The PDP-15 loader will ask:

DATA SET NAME?

Reply with the name of the SDS-910 program data set, CNTL D.

The loader will load the data set with the SDS-910 and return with:

NORMAL END, NEXT DATA SET NAME?

If you have more data sets to be loaded across the interface,
specify them at this time. Otherwise, reply with a CNTL D.

This will end the loader program.

NOTE: It is assumed that the user will be using the system location
which has a zero in it and not have a location in his own code for
this purpose. Therefore, the loader does not place zeroes into a
users area. This can be a problem if the user is loading over an area
that has not been zeroed and he wants the constant zero in his own area.

1				RLD00010
*				RLD00020
*	THIS IS THE INITIAL LOADER ROUTINE			RLD00030
*				RLD00040
	ACRG	2		RLD00050
IDLB00	PZE		/RESET INTERRUPTS	RLD00060
	BRU*	*-1	/GO BACK TO ROUTINE	RLD00070
TIME	OCT	0		RLD00080
	ACRG	331		RLD00090
NTHETA	OCT	0		RLD00100
NOMEGA	OCT	0		RLD00110
NFHI	OCT	0		RLD00120
NCHI	OCT	0		RLD00130
CTHETA	OCT	0		RLD00140
DCMEGA	OCT	0		RLD00150
DPHI	OCT	0		RLD00160
DCHI	OCT	0		RLD00170
AVAIL	OCT	0		RLD00180
	AORG	6300		RLD00190
	DIR		/DISABLE INTERRUPTS	RLD00200
	LDA	INT1		RLD00210
	STA	INT		RLD00220
	LDA	LCC1		RLD00230
	STA	LOC		RLD00240
	LDA	ADDR3		RLD00250
	STA	ADDR2		RLD00260
	CLR		/CLEAR REGISTERS	RLD00270
	STA	ADDR		RLD00280
	LDX	COUNT	/AMOUNT OF CORE TO ZERO	RLD00290
	STA*	ADDR	/ZERO CORE	RLD00300
	MIN	ADDR	/INCREMENT ADDRESS	RLD00310
	BRX	*-2	/BRANCH UNTIL ALL CORE IS ZEROED	RLD00320
	LDX	COUNT2		RLD00330
	STA*	ADDR2		RLD00340
	MIN	ADDR2		RLD00350
	BRX	*-2		RLD00360
***				RLD00370

***				RLD00380
	LDA	LOC		RLD00390
	MRG	BIT9		RLD00400
	STA	LOC		RLD00410
	LDX	CNT1	/LOAD X WITH NUMBER OF LOADS	RLD00420
INIT	LCA*	INT	/INITIALIZE INTERRUPT VECTORS	RLD00430
	STA*	LOC	/FOR CLOCKS AND RTERROR	RLD00440
	MIN	INT	/INCREMENT INT	RLD00450
	MIN	LOC	/INCREMENT LOC	RLD00460
	BRX	*-4	/BRANCH UNTIL ALL INITIALIZATION IS DONE	RLD00470
	EIR		/ENABLE INTERRUPTS	RLD00480
	BRU	WAIT		RLD00490
***				RLD00500
***				RLD00510
TRANS	PZE		/TRANSFER INFORMATION FROM 15	RLD00520
	ECM	32300	/RESET OUTBOARD REGISTER TO WAIT FOR	RLD00530
	POT	ZEROES	/NEXT INTERRUPT	RLD00540
	ECM	30301	/READ WORD 1 FROM INTERFACE	RLD00550
	PIN	DATA	/	RLD00560
	EOM	30302	/READ WORD 2 FROM INTERFACE	RLD00570
	PIN	DATA+1		RLD00580
	EOM	30303	/READ WORD 3 FROM INTERFACE	RLD00590
	PIN	DATA+2		RLD00600
	LDA	DATA	/CHECK FIRST WORD FOR	RLD00610
	LCB	MINUS1	SYSTEM SEND?	RLD00620
	SKM	SYSMAS		RLD00630
	BRU	NONSND	/NO, GO TO NON-SEND	RLD00640
SYSSND	ECM	32304	/YES, SYSTEM SEND, RETURN INFORMATION	RLD00650
	POT	C1	/SET REQUEST	RLD00660
	EOM	36307	/TEST 15 LOCK-UP	RLD00670
	SKS	30004		RLD00680
	BRU	*-2	/WAIT UNTIL LOCK-UP NOT SET	RLD00690
	ECM	32305	/SET BUSY	RLD00700
	POT	C1		RLD00710
	EOM	32304		RLD00720
	POT	ZEROES		RLD00730
	EOM	32307	/SET 910 INITIATE	RLD00740

POT	C40		RLDC0750
LDB	DATA+1		RLDC0760
LCY	6		RLDC0770
XAB			RLDC0780
ETR	KEYMS4		RLDC0790
STA	SENDT1		RLDC0800
LDB	DATA+1		RLDC0810
LDA	DATA+2		RLDC0820
RCY	12		RLDC0830
STB	DATA+1		RLDC0840
LCA	DATA		RLDC0850
RCY	6		RLDC0860
ETR	STAT		RLDC0870
STA	DATA		RLDC0880
LDB	DATA+2		RLDC0890
LCY	6		RLDC0900
LDA	DATA		RLDC0910
LCY	6		RLDC0920
STA	DATA		RLDC0930
LDA	SENDT1		RLDC0940
STA	DATA+2		RLDC0950
EOM	32301	/SEND INFORMATION BACK TO 15	RLDC0960
POT	DATA	/FIRST WORD	RLDC0970
ECM	32302		RLDC0980
POT	DATA+1	/SECOND WORD	RLDC0990
ECM	32303		RLDC1000
POT	DATA+2	/THIRD WORD	RLDC1010
BRU	WAIT	/WAIT UNTIL 15 GIVES MORE INFORMATION	RLDC1020
***			RLDC1030
***			RLDC1040
NCNSND	SKN	DATA	RLDC1050
	BRU	*+2	RLDC1060
	BRU	WAIT	RLDC1070
	CLR		RLDC1080
	LDB	DATA+1	RLDC1090
	LCY	12	RLDC1100
	ADD	MINUS1	RLDC1110

	STA	SENDT1		RLD01120
	SKN	SENDT1		RLD01130
	BRU	*+2		RLD01140
	BRU	WAIT		RLD01150
	LDA	DATA	/TEST FOR PROPER SYSTEM STATUS WORD	RLD01160
	SKA	STATUS		RLD01170
	BRU	*+2		RLD01180
	BRU	WAIT		RLD01190
	LCA	DATA+1	/ANALYZE LENGTH	RLD01200
	LDB	DATA+2		RLD01210
	LCY	6	/LEFT CYCLE TO GET KEY IN A REGISTER	RLD01220
	ETR	MASK	/MASK LENGTH	RLD01230
	STA	LENGTH	/STORE IN LENGTH	RLD01240
	LDB	MINUS1	/TEST FOR A ZERO	RLD01250
	SKM	FOUR	/CHECK FOR LENGTH OF FOUR	RLD01260
	BRU	*+2		RLD01270
	BRU	LAST	/YES, TEST FOR LAST RECORD	RLD01280
***				RLD01290
***				RLD01300
	LDA	STLEN	/NO, READ NEXT INFORMATION	RLD01310
	STA	LENGTH		RLD01320
	BRU	*+2		RLD01330
TRANS1	PZE			RLD01340
	ECM	36300	/TEST BUFFER FULL	RLD01350
	SKS	30001		RLD01360
	BRU	*-2	/LOOP UNTIL FULL	RLD01370
	EOM	30301	/PUT ADDRESS INTO DATA	RLD01380
	PIN	DATA	/AND USE BY INPUT ROUTINE	RLD01390
SECCND	ECM	30302	/READ FIRST DATA WORD	RLD01400
	PIN	ADDR	/STORE TEMP	RLD01410
	LCA	ADDR	/LOAD VALUE	RLD01420
	SKA	MINUS1		RLD01430
	STA*	DATA	/NO, STORE IN DATA	RLD01440
	MIN	DATA	/YES, JUST INCREMENT DATA ADDRESS	RLD01450
	ECM	30303	/READ SECOND DATA WORD	RLD01460
	PIN	ADDR	/STORE TEMP	RLD01470
	LDA	ADDR	/LCAD VALUE	RLD01480

	SKA	MINUS1		RLD01490
	STA*	DATA	/NO, STORE IN DATA	RLD01500
	MIN	DATA	/YES, JUST INCREMENT DATA ADDRESS	RLD01510
	LDA	LENGTH	/DECREMENT LENGTH BY FOUR	RLD01520
	SLB	FOUR		RLD01530
	STA	LENGTH		RLD01540
	SKN	LENGTH	/TEST IF LENGTH IS NEGATIVE,	RLD01550
	BRU	*+2		RLD01560
	BRU	WAIT	/NO MORE DATA TO BE READ	RLD01570
	EOM	36300	/TEST BUFFER FULL	RLD01580
	SKS	30001		RLD01590
	BRU	*-2	/WAIT UNTIL BUFFER IS FULL	RLD01600
	ECM	30301	/READ FIRST DATA WORD	RLD01610
	PIN	ADDR	/STORE TEMP	RLD01620
	LDA	ADDR	/LOAD VALUE	RLD01630
	SKA	MINUS1		RLD01640
	STA*	DATA	/NO, STORE IN DATA	RLD01650
RDIT	MIN	DATA	/INCREMENT ADDRESS	RLD01660
	BRU	SECOND		RLD01670
***				RLD01680
***				RLD01690
WAIT	LDA	227	/TEST FOR SYSTEM IN CONTROL	RLD01700
	SKA	SYSMSK		RLD01710
	BRU*	*+2	/NC, WAIT	RLD01720
	BRR	TRANS1		RLD01730
	HLT	*+1	/CLEAR INTERRUPT LEVEL	RLD01740
	STA	DATA+1	/WAIT LOOP NEED DUMMY INSTRUCTION	RLD01750
	BRU	*-1		RLD01760
***				RLD01770
***				RLD01780
LAST	ECM	36300	/TEST BUSY	RLD01790
	SKS	30001		RLD01800
	BRU	*-2		RLD01810
	ECM	30301	/GET FIRST WORD	RLD01820
	PIN	DATA	/SET UP ADDRESS FIELD	RLD01830
	LDA	DATA	/LOAD AND CHECK FOR 54 PATTERN	RLD01840
	SKA	FIVFOR	/TEST FOR 54 PATTERN	RLD01850

BRU	+	2		RLD01860				
BRU	R	CIT	/GET NEXT WORD	RLDC1870				
ECM	3	0302	/YES, READ NEXT WORD AND ANALYZE FOR ADDRESS	RLCC1880				
PIN	D	A+1		RLDC1890				
EOM	3	0303		RLD01900				
PIN	D	A+2		RLDC1910				
LDA	D	A+1		RLDC1920				
LDB	M	INUS 1		RLD01930				
SKM	Z	ER0ES		RLDC1940				
BRU*	D	A+1		RLC01950				
ERU	W	AIT		RLD01960				
***				RLDC1970				
***				RLD01980				
CCUNT	O	C	77771500	RLDC1990				
ADDR	O	C	0	RLD02000				
ADDR2	O	C	6740	RLD02010				
ADDR3	O	C	6740	RLDC2020				
CCUNT2	O	C	77750350	RLD02030				
INT1	H	L	T	+	2	RLDC2040		
INT	H	L	T	+	1	RLD02050		
* INTERRUPT ASSIGNMENT FOR REACTOR SDS-910 SYSTEM								
PZE					RLDC2060			
BRU*	2				RLDC2070			
MIN	T	I	M	E	/SPECIAL 1.6 SECOND CLOCK	RLD02080		
BRM	I	D	L	B	00	RLDC2090		
MIN	N	T	H	E	T	A	/RTERROR INTERRUPT	RLD02100
MDE	N	T	H	E	T	A	/NEUTRON THETA2	RLD02110
MIN	N	O	M	E	G	A	/NEUTRON THETA2	RLDC2120
MDE	N	O	M	E	G	A	/NEUTRON OMEGA2	RLD02130
MIN	N	O	M	E	G	A	/NEUTRON OMEGA2	RLD02140
MDE	N	P	H	I		/NEUTRON PHI2	RLD02150	
MIN	N	P	H	I		/NEUTRON PHI2	RLD02160	
MDE	N	C	H	I		/NEUTRON CHI2	RLD02170	
MIN	N	C	H	I		/NEUTRON CHI2	RLD02180	
MDE	D	T	H	E	T	A	/DATEX THETA3	RLD02190
MIN	D	T	H	E	T	A	/DATEX THETA3	RLD02200
MDE	D	O	M	E	G	A	/DATEX OMEGA3	RLDC2210
MIN	D	O	M	E	G	A	/DATEX OMEGA3	RLD02220
MDE	D	O	M	E	G	A	/DATEX OMEGA3	RLD02220

	MIN	DPHI	/DATEX PHI3	RLD02230
	MDE	DPHI	/DATEX PHI3	RLD02240
	MIN	DCHI	/DATEX CHI3	RLD02250
	MDE	DCHI	/DATEX CHI3	RLD02260
	STA	AVAIL	/AVAILABLE INTERRUPT LOCATION	RLD02270
	STA	AVAIL	/AVAILABLE INTERRUPT LOCATION	RLD02280
	STA	AVAIL	/AVAILABLE INTERRUPT LOCATION	RLD02290
	STA	AVAIL	/AVAILABLE INTERRUPT LOCATION	RLD02300
	STA	AVAIL	/AVAILABLE INTERRUPT LOCATION	RLD02310
	BRM	TRANS	/15 TO 910 INTERRUPT	RLD02320
	BRM	IDL800	/LIMIT 2	RLD02330
	BRM	IDL800	/COUNT 2	RLD02340
	BRM	IDL800	/LIMIT 3	RLD02350
	BRM	IDL800	/COUNT 3	RLD02360
	BRM	IDL800	/ISOTOPE SEPARATOR	RLD02370
	BRM	IDL800	/ISOTOPE SEPARATOR	RLD02380
	BRM	IDL800	/THUMB CHARACTER	RLD02390
	BRM	IDL800	/THUMB BLOCK	RLD02400
	BRM	IDL800	/FINGER CHARACTER	RLD02410
	BRM	IDL800	/FINGER BLOCK	RLD02420
	BRM	IDL800	/TRIPLE AXIS	RLD02430
	BRM	IDL800	/PLCTTER	RLD02440
	BRM	IDL800		RLD02450
	BRM	IDL800		RLD02460
	BRM	IDL800	/FAST CLOCK INTERRUPT	RLD02470
	BRM	IDL800	/MEDIUM CLOCK INTERRUPT	RLD02480
	BRM	IDL800	/SLOW CLOCK INTERRUPT	RLD02490
LCC1	HLT	*+2		RLD02500
LOC	HLT	*+1		RLD02510
	OCT	2		RLD02520
	OCT	3		RLD02530
	OCT	200	/1.6 SEC REAL TIME CLOCK	RLD02540
	OCT	201	/ADDRESS OF INTERRUPTS TO BE LOADED IN INIT.	RLD02550
	OCT	202		RLD02560
	OCT	203		RLD02570
	OCT	204		RLD02580
	OCT	205		RLD02590

	OCT	206		RLD02600
	OCT	207		RLDC2610
	OCT	210		RLD02620
	OCT	211		RLD02630
	OCT	212		RLD02640
	OCT	213		RLD02650
	OCT	214		RLDC2660
	OCT	215		RLD02670
	OCT	216		RLD02680
	OCT	217		RLDC2690
	OCT	220		RLD02700
	OCT	221		RLDC2710
	OCT	222		RLD02720
	OCT	223		RLD02730
	OCT	224		RLD02740
	OCT	225		RLD02750
	OCT	226		RLDC2760
	OCT	227		RLDC2770
	OCT	230		RLD02780
	OCT	231		RLDC2790
	OCT	232		RLD02800
	OCT	233		RLDC2810
	OCT	234		RLDC2820
	OCT	235		RLC02830
	OCT	236		RLDC2840
	OCT	237		RLDC2850
	OCT	240		RLD02860
	OCT	241		RLDC2870
	OCT	242		RLD02880
	OCT	243		RLDC2890
	OCT	244		RLD02900
	OCT	245		RLD02910
	OCT	246		RLDC2920
	OCT	247		RLD02930
	OCT	250		RLD02940
CNT 1	OCT	77777725		RLD02950
DATA	OCT	0	/TEMPORARY DATA AREA.	RLD02960

	OCT	0	/TEMPORARY DATA AREA	RLDC2970
	OCT	0	/TEMPORARY DATA AREA	RLDC02980
MASK	OCT	00777777	/MASK OFF LENGTH	RLDC2990
FIVFOR	OCT	54545454		RLD03000
ZEROES	OCT	0		RLD03010
MASK7	OCT	77770000		RLDC3020
SYMSK	BRM	TRANS		RLD03030
LENGTH	OCT	0		RLD03040
C1	OCT	1		RLD03050
C40	OCT	40		RLD03060
STATUS	OCT	14051000	/EXPECTED STATUS WORD	RLD03070
SYS	OCT	1400000		RLD03080
STAT	OCT	777777		RLDC3090
KEYMS4	OCT	77000000		RLD03100
SENDT1	OCT	0		RLD03110
SYSMAS	OCT	20051000		RLD03120
BIT9	OCT	40000		RLD03130
FCUR	OCT	4		RLDC3140
STLEN	OCT	53		RLDC03150
MINUS1	OCT	77777777		RLD03160
	END			RLDC3170

1				JLD00010
	ACRG	331		JLD00020
NTHETA	OCT	0		JLD00030
NCEGA	OCT	0		JLD00040
NPHI	OCT	0		JLD00050
NCHI	OCT	0		JLD00060
AVAIL	OCT	0		JLD00070
	AORG	14000		JLD00080
ZERO	DIR		/DISABLE INTERRUPTS	JLD00090
	LDA	INT1		JLD00100
	STA	INT		JLD00110
	LDA	LCC1		JLD00120
	STA	LOC		JLD00130
	CLR		/CLEAR REGISTERS	JLD00140
	STA	ADDR		JLD00150
	LDX	COUNT	/AMOUNT OF CORE TO ZERO	JLD00160
	STA*	ADDR	/ZERO CORE	JLD00170
	MIN	ADDR	/INCREMENT ADDRESS	JLD00180
	BRX	*-2	/BRANCH UNTIL ALL CORE IS ZEROED	JLD00190
***				JLD00200
***				JLD00210
	LDA	LOC		JLD00220
	MRG	BIT9		JLD00230
	STA	LOC		JLD00240
	LDX	CNT	/LOAD X WITH NUMBER OF LOADS	JLD00250
INIT	LDA*	INT	/INITIALIZE INTERRUPT VECTORS	JLD00260
	STA*	LOC	/FOR CLOCKS AND RTERROR	JLD00270
	MIN	INT	/INCREMENT INT	JLD00280
	MIN	LOC	/INCREMENT LOC	JLD00290
	BRX	*-4	/BRANCH UNTIL ALL INITIALIZATION IS DONE	JLD00300
	EIR		/ENABLE INTERRUPTS	JLD00310
	BRU	WAIT		JLD00320
***				JLD00330
***				JLD00340
TRANS	PZE		/TRANSFER INFORMATION FROM 15	JLD00350
	EOM	32300	/RESET OUTBOARD REGISTER TO WAIT FOR	JLD00360
	POT	ZEROES	/NEXT INTERRUPT	JLD00370

ECM	30301	/READ WORD 1 FROM INTERFACE	JLDC0380
PIN	DATA	/	JLD00390
ECM	30302	/READ WORD 2 FROM INTERFACE	JLDC0400
PIN	DATA+1		JLDC0410
ECM	30303	/READ WORD 3 FROM INTERFACE	JLD00420
PIN	DATA+2		JLDC0430
LDA	DATA	/CHECK FIRST WORD FOR	JLD00440
LCB	MINUS1	SYSTEM SEND?	JLD00450
SKM	SYSMAS		JLD00460
BRU	NONSND	/NO, GO TO NON-SEND	JLD00470
SYSSND ECM	32304	/YES, /SYSTEM SEND, RETURN INFORMATION	JLDC0480
POT	C1	/SET REQUEST	JLD00490
ECM	36307	/TEST 15 LOCK-UP	JLDC0500
SKS	30004		JLD00510
BRU	*-2	/WAIT UNTIL LOCK-UP NOT SET	JLD00520
ECM	32305	/SET BUSY	JLD00530
POT	C1		JLD00540
ECM	32304		JLD00550
POT	ZEROES		JLD00560
ECM	32307	/SET 910 INITIATE	JLDC0570
POT	C40		JLD00580
LCB	DATA+1		JLDC0590
LCY	6		JLDC0600
XAB			JLD00610
ETR	KEYMS4		JLDC0620
STA	SENDT1		JLDC0630
LCB	DATA+1		JLD00640
LDA	DATA+2		JLDC0650
RCY	12		JLDC0660
STB	DATA+1		JLDC0670
LDA	DATA		JLDC0680
RCY	6		JLD00690
ETR	STAT		JLDC0700
STA	DATA		JLDC0710
LDB	DATA+2		JLDC0720
LCY	6		JLD00730
LDA	DATA		JLD00740

LCY	6		JLDC0750
STA	DATA		JLDC0760
LCA	SENDT1		JLD00770
STA	DATA+2		JLD00780
EOM	32301	/SEND INFORMATION BACK TO 15	JLD00790
POT	DATA	/FIRST WORD	JLD00800
ECM	32302		JLD00810
POT	DATA+1	/SECOND WORD	JLD00820
EOM	32303		JLD00830
POT	DATA+2	/THIRD WORD	JLC00840
BRU	WAIT	/WAIT UNTIL 15 GIVES MORE INFORMATION	JLD00850
***			JLD00860
***			JLD00870
NONSD	SKN	DATA	JLD00880
	BRU	*+2	JLD00890
	BRU	WAIT	JLD00900
	CLR		JLD00910
LCB	DATA+1	/TEST FOR OPEN OF THE 15 SYSTEM	JLD00920
LCY	12		JLD00930
ADD	MINUS1		JLD00940
STA	SENDT1		JLD00950
SKN	SENDT1		JLD00960
BRU	*+2		JLD00970
BRU	WAIT		JLD00980
LDA	DATA	/TEST FOR PROPER SYSTEM STATUS WORD	JLD00990
SKA	STATUS		JLD01000
BRU	*+2		JLD01010
BRU	WAIT		JLD01020
LDA	DATA+1	/ANALYZE LENGTH	JLD01030
LCB	DATA+2		JLD01040
LCY	6	/LEFT CYCLE TO GET KEY IN A REGISTER	JLD01050
ETR	MASK	/MASK LENGTH	JLD01060
STA	LENGTH	/STORE IN LENGTH	JLD01070
LDB	MINUS1	/TEST FOR A ZERO	JLD01080
SKM	FOUR	/CHECK FOR LENGTH OF FOUR	JLD01090
BRU	*+2		JLD01100
BRU	LAST	/YES, TEST FOR LAST RECORD	JLD01110

***				JLD01120
***				JLDC1130
	LDA	STLEN	/NO. READ NEXT INFORMATION	JLD01140
	STA	LENGTH		JLDC1150
	BRU	*+2		JLDC1160
TRANS1	PZE			JLD01170
	ECM	36300	/TEST BUFFER FULL	JLD01180
	SKS	30001		JLD01190
	BRU	*-2	/LOOP UNTIL FULL	JLDC1200
	ECM	30301	/PUT ADDRESS INTO DATA	JLD01210
	PIN	DATA	/AND USE BY INPUT ROUTINE	JLD01220
SECOND	ECM	30302	/READ FIRST DATA WORD	JLDC1230
	PIN	ADDR	/STORE TEMP	JLD01240
	LDA	ADDR	/LOAD VALUE	JLDC1250
	SKA	MINUS1		JLDC1260
	STA*	DATA	/NO. STORE IN DATA	JLD01270
	MIN	DATA	/YES. JUST INCREMENT DATA ADDRESS	JLD01280
	ECM	30303	/READ SECOND DATA WORD	JLD01290
	PIN	ADDR	/STORE TEMP	JLDC1300
	LDA	ADDR	/LOAD VALUE	JLD01310
	SKA	MINUS1		JLD01320
	STA*	DATA	/NC. STORE IN DATA	JLDC1330
	MIN	DATA	/YES. JUST INCREMENT DATA ADDRESS	JLD01340
	LCA	LENGTH	/DECREMENT LENGTH BY FOUR	JLDC1350
	SUB	FOUR		JLDC1360
	STA	LENGTH		JLDC1370
	SKN	LENGTH	/TEST IF LENGTH IS NEGATIVE,	JLDC1380
	BRU	*+2		JLD01390
	BRU	WAIT	/NC MORE DATA TO BE READ	JLDC1400
	ECM	36300	/TEST BUFFER FULL	JLD01410
	SKS	30001		JLD01420
	BRU	*-2	/WAIT UNTIL BUFFER IS FULL	JLDC1430
	ECM	30301	/READ FIRST DATA WORD	JLD01440
	PIN	ADDR	/STORE TEMP	JLDC1450
	LDA	ADDR	/LOAD VALUE	JLD01460
	SKA	MINUS1		JLD01470
	STA*	DATA	/NC. STORE IN DATA	JLDC1480

RDIT	MIN	DATA	/INCREMENT ADDRESS	JLD01490
	BRU	SECOND		JLD01500
***				JLD01510
***				JLD01520
WAIT	LDA	217	/TEST FOR SYSTEM IN CONTROL	JLD01530
	SKA	SYSMSK		JLD01540
	BRU*	*+2	/NC, WAIT	JLD01550
	BRR	TRANS1		JLD01560
	HLT	*+1	/CLEAR INTERRUPT LEVEL	JLD01570
	STA	DATA+1	/WAIT LOOP NEED DUMMY INSTRUCTION	JLD01580
	BRU	*-1		JLD01590
***				JLD01600
***				JLD01610
LAST	ECM	36300	/TEST BUSY	JLD01620
	SKS	30001		JLD01630
	BRU	*-2		JLD01640
	ECM	30301	/GET FIRST WORD	JLD01650
	PIN	DATA	/SET UP ADDRESS FIELD	JLD01660
	LDA	DATA	/LOAD AND CHECK FOR 54 PATTERN	JLD01670
	SKA	FIVFOR	/TEST FOR 54 PATTERN	JLD01680
	BRU	*+2		JLD01690
	BRU	RDIT	/GET NEXT WORD	JLD01700
	ECM	30302	/YES, READ NEXT WORD AND ANALYZE FOR ADDRESS	JLD01710
	PIN	DATA+1		JLD01720
	ECM	30303		JLD01730
	PIN	DATA+2		JLD01740
	LDA	DATA+1		JLD01750
	LDB	MINUS1		JLD01760
	SKM	ZEROES		JLD01770
	BRU*	DATA+1		JLD01780
	BRU	WAIT		JLD01790
***				JLD01800
***				JLD01810
IDL800	PZE		/RESET INTERRUPTS	JLD01820
	BRU*	*-1	/GO BACK TO ROUTINE	JLD01830
***				JLD01840
***				JLD01850

COUNT	OCT	77764000		JLD01860
ADDR	OCT	0		JLD01870
INT1	HLT	*+2		JLD01880
INT	HLT	*+1		JLD01890
	MIN	TIME	/SPECIAL 1.6 SECOND CLOCK	JLD01900
	BRM	IDL800	/R TERROR INTERRUPT	JLD01910
	MIN	NTHETA	/NEUTRON THETA2	JLD01920
	MDE	NTHETA	/NEUTRON THETA2	JLD01930
	MIN	NOMEGA	/NEUTRON OMEGA2	JLD01940
	MDE	NOMEGA	/NEUTRON OMEGA2	JLD01950
	MIN	NPHI	/NEUTRON PHI2	JLD01960
	MDE	NPHI	/NEUTRON PHI2	JLD01970
	MIN	NCHI	/NEUTRON CHI2	JLD01980
	MDE	NCHI	/NEUTRON CHI2	JLD01990
	BRM	TRANS	/15 TO 910 INTERRUPT	JLD02000
	BRM	IDL800		JLD02010
	BRM	IDL800		JLD02020
	BRM	IDL800	/FAST CLOCK INTERRUPT	JLD02030
	BRM	IDL800	/MEDIUM CLOCK INTERRUPT	JLD02040
	BRM	IDL800	/SLOW CLOCK INTERRUPT	JLD02050
LOC1	HLT	*+2		JLD02060
LCC	HLT	*+1		JLD02070
	OCT	200	/1.6 SEC REAL TIME CLOCK	JLD02080
	OCT	201	/ADDRESS OF INTERRUPTS TO BE LOADED IN INIT.	JLD02090
	OCT	202	/2THETA +	JLD02100
	OCT	203	/2THETA -	JLD02110
	OCT	204	/OMEGA +	JLD02120
	OCT	205	/OMEGA -	JLD02130
	OCT	206	/CHI +	JLD02140
	OCT	207	/CHI -	JLD02150
	OCT	210	/PHI +	JLD02160
	OCT	211	/PHI -	JLD02170
	OCT	217	/R TERROR, 15-910	JLD02180
	OCT	220	/COUNT COMPLETE	JLD02190
	OCT	221	/LIMIT SWITCH	JLD02200
	OCT	224	/FAST CLOCK	JLD02210
	OCT	225	/MEDIUM CLOCK	JLD02220

	OCT	226	/SLOW CLOCK	JLD02230
CNT	OCT	77777760	/NUMBER OF ITEMS TO BE LOADED	JLD02240
DATA	OCT	0	/TEMPORARY DATA AREA	JLD02250
	OCT	0	/TEMPORARY DATA AREA	JLD02260
	OCT	0	/TEMPORARY DATA AREA	JLD02270
MASK	OCT	00777777	/MASK OFF LENGTH	JLD02280
FIVFOR	OCT	54545454		JLD02290
ZEROES	OCT	0		JLD02300
MASK7	OCT	77770000		JLD02310
SYSMSK	BRM	TRANS		JLD02320
LENGTH	OCT	0		JLD02330
C1	OCT	1		JLD02340
C40	OCT	40		JLD02350
STATUS	OCT	14051000	/EXPECTED STATUS WORD	JLD02360
SYS	OCT	01400000	/910-15 SYSTEM SEND XOR'ED	JLD02370
TIME	OCT	0		JLD02380
FCUR	OCT	4		JLD02390
BITS	OCT	40000		JLD02400
STLEN	OCT	53		JLD02410
STAT	OCT	777777		JLD02420
KEYMS4	OCT	77000000		JLD02430
SENDT1	OCT	0		JLD02440
SYSMAS	OCT	20051000		JLD02450
MINUS1	OCT	77777777		JLD02460
	END			JLD02470

RAID--THE SDS-910 DEBUG PACKAGE:

This routine is a stand alone teletype debugging aid¹. On the Reactor SDS-910 system it is located at 36370₈. On Jacobson's system it is located at 16370₈. RAID provides the following functions upon command from the SDS-910 console:

1. Octal dump of memory to the console
2. Address stop
3. Enter octal information into memory from the console
4. Punch memory on to paper tape
5. Octal dump of memory to magnetic tape on PDP-15
6. Branch to location in memory
7. Continue processing from point of address stop.

RAID is on PDP-15 disk and is loaded into the SDS-910 with the loading routines explained in this manual. The symbol C_R means a carriage return character. All number values will be octal numbers in the following description.

ENTRANCE TO RAID:

There are three methods of entering RAID:

1. Branch to 36371 (Reactor system).
Branch to 16371 (Jacobson's system).
2. BRU* 23 (when OS910 Version 2 is already in core).
3. Address stop.

When RAID is entered it saves the A, B, and X registers and location 0. If RAID is entered via mode 3 (address stop) it also saves the return location. Upon exit (using a Continue command) from RAID the A, B, and X registers and location 0 are restored. RAID executes with all interrupts disabled.

¹Original source of the code is unknown.

RAID COMMANDS

RAID is ready for input whenever the light on the teletype is on. Whenever RAID reads a C_R from the teletype it types a line feed character. If the command being input to RAID requires one or more numbers, the following rules apply:

1. All numbers are octal (0,1,2,3,4,5,6,7).
All other characters are ignored.
2. Leading C_R are ignored.
3. If two numbers are expected they must be separated with a comma.
4. Numbers are right justified if fewer than eight characters and truncated on the left if greater than eight characters.
5. A "null" number (,, or, , or, C_R) is replaced by zeroes.

ENTER information into memory from the teletype

EADR,WORD0. C_R

WORD1. C_R

.

.

.

WORDN. C_R

C_R

ADR is the address at which the information is to start loading. WORD0 is stored at ADR, WORD1 is stored at ADR+1, ..., WORDN is stored at ADR+N. Two carriage returns are needed to terminate enter.

LIST memory to the teletype

DADR1[,ADR2]. C_R

This causes an octal dump of memory. The memory dumped is from ADR1 to ADR2 inclusive. If the expression within the brackets is missing then only the contents of ADR1 is dumped.

A five digit address is printed first on every line of the output.

ADDRESS STOP

AADR.C_R

This sets-up an address stop at location ADR. This is a branch to RAID in which location 0 and registers A, B, and X are saved. The return address is the location ADR. Upon entering RAID with an address stop the instruction at ADR is restored. At present only one address stop at a time can be used.

CONTINUE

C

This command is used to continue processing from the point of an address stop. It restores the registers and location 0 to the value saved in ADDRESS STOP.

JUMP

JADR.C_R

JUMP will cause RAID to branch to location ADR.

DUMP MEMORY to PDF-15 magnetic tape

*DADR1,ADR2.C_R

This command is used in conjunction with the PDP-15 utility CORE TO TAPE. It dumps 1400_8 SDS-910 words or 2000_8 PDP-15 words at a time. The dump is accomplished by starting at location ADR1 and going in 1400_8 SDS-910 word blocks until ADR2 has been satisfied.

The PDP-15 CORE TO TAPE program first dumps the PDP-15 core to magnetic tape and then waits for a dump from the SDS-910.

PAPER TAPE COMMANDS:

The paper tape commands are for two purposes. First is to allow patches to be made while debugging a program. Second is to make self-loading paper tapes of entire programs. At present, only Jacobson's SDS-910 computer has a paper tape punch available.

PUNCH BOOTSTRAP LOADER

$H.C_R$

This command punches a bootstrap absolute loader on the front of the paper tape. This command must be used if you want a self-loading paper tape.

PUNCH

$TADR1[,ADR2].C_R$

This command will punch memory on paper tape from location ADR1 to location ADR2 inclusive. If only ADR1 is given, only the memory value at location ADR1 is punched on paper tape.

PUNCH TRANSFER RECORD

SADR.C_R

This command punches a transfer (BRU) to the address (ADR) specified. If ADR is zero or unspecified the loader halts at the end of the load.

	AORG	36370
START	PZE	**
	DIR	
	BRM	SETPOP
	BRM	TYPRTN
	BRM	RESETP
	EIR	
	BRR	START
SETPOP	PZE	**
	LDA	PAD150
	LDB	PAD160
	STA	SPOP1
	STB	SPOP2
	LDA	PCPB1
	LDB	POP82
	STA	PAD150
	STB	PAD160
	BRR	SETPOP
PAD150	BOOL	176
PAD160	BOOL	177
SPOF1	PZE	**
SPOP2	PZE	**
FOPE1	BRU	POPSKE
POP82	BRU	PCPSKR
RESETP	PZE	**
	LCA	SPOP1
	LDB	SPOP2
	STA	PAD150
	STB	PAD160
	BRR	RESETP
* ADDRESS STOP ROUTINES *		
*		
ADSTOP	PZE	
	ECM	20004
	STA	ACSA
	STB	ACSB

STOP REACHED

RAIC0040
 RAIC0050
 RAID0060
 RAIC0070
 RAIC0080
 RAIC0090
 RAID0100
 RAIC0110
 RAIC0120
 RAID0130
 RAIC0140
 RAIC0150
 RAID0160
 RAIC0170
 RAID0180
 RAIC0190
 RAID0200
 RAIC0210
 RAIC0220
 RAID0230
 RAIC0240
 RAIC0250
 RAIC0260
 RAIC0270
 RAID0280
 RAIC0290
 RAID0300
 RAIC0310
 RAIC0320
 RAID0330
 RAID0340
 RAID0350
 RAIC0360
 RAIC0370
 RAID0380

STX	ADSX	RAIC0390
LCA	0	RAIC0400
STA	ADSO	RAID0410
BRM	SETPOP	RAIC0420
LDA	MASTPI	RAIC0430
STA*	MASTPA	RAIC0440
LCA	ADS	RAID0450
BRM	OUT4PW	RAIC0460
LDA	ADS+1	RAIC0470
BRM	OUT4PW	RAIC0480
LDA	MASTPA	RAIC0490
BRM	TADDR	RAIC0500
LCA	MASTPS	RAIC0510
SKE	ZERO	RAIC0520
BRU	*+3	RAIC0530
BRM	OUTCRT	RAIC0540
BRU	*+4	RAIC0550
BRM	OUTSP4	RAIC0560
LDA	ZNAME	RAIC0570
BRM	OUT4PW	RAIC0580
BRM	OUTCRT	RAIC0590
CLR		RAID0600
STA	MASTPA	RAIC0610
STA	MASTPS	RAIC0620
BRM	TYPRTN	RAID0630
BRM	RESETP	RAIC0640
LDA	ADSTOP	RAIC0650
ETR	MASK6	RAID0660
MRG	BRUCON	RAIC0670
STA	ACSXIT	RAIC0680
LDA	ADSO	RAIC0690
STA	0	RAIC0700
LCA	ADSA	RAID0710
LDB	ADSB	RAIC0720
LCX	ADSX	RAID0730
EIR		RAIC0740
ADSXIT PZE	**	RAIC0750

```

MASK6 OCT 37777
SVXREG PZE **
BRUCGN BRU 0
*
ASTOPS PZE
      LCA* MASTPA
      STA MASTPI
      LDA MASTPB
      STA* MASTPA
      BRR ASTOPS
*
MASTPA PZE
MASTPS PZE
MASTPI PZE
MASTPB BRM ADSTOP
ACSA PZE
ADSB PZE
ADX PZE
ADSO PZE
ACS BCD 7,ADR STOP
*
* END OF ADR STOP *
      PAGE
* REM TYPEWRITER ROUTINE
*
* CCLD START ENTRY
*
TYPBEG LCA **3
      STA TYPRTN
      BRU TYPRTN+1
      HLT *
      BRM OUTCRT
      BRU TYPRTN+1
*
* SYSTEM ENTRY
*
TYPRTN PZE **

```

SET STOP

STOP ADDRESS
 STOP STATE
 INST REPLACED
 TRAP INSTR

```

RAIC0760
RAIC0770
RAIC0780
RAID0790
RAIC0800
RAIC0810
RAIC0820
RAID0830
RAIC0840
RAIC0850
RAID0860
RAIC0870
RAIC0880
RAID0890
RAIC0900
RAIC0910
RAIC0920
RAIC0930
RAID0940
RAIC0950
RAIC0960
RAIC0970
RAIC0980
RAID0990
RAIC1000
RAIC1010
RAIC1020
RAID1030
RAID1040
RAIC1050
RAID1060
RAIC1070
RAIC1080
RAID1090
RAIC1100
RAIC1110
RAIC1120

```


* STX TYPXSV
 * FIRST CHARACTER
 *
 TYP C1 BRM TYP IN
 BRU *+1.2
 BRU TYP C1 C
 BRU TYP C1 O
 BRU TYP C1 I
 BRU TYP C1 O
 BRU TYP C1 O
 TYP C1 C BRM OUT CRT
 BRU TYP C1
 TYP C1 O BRM TYP XX
 BRU TYP C1 C
 TYP C1 I STA TYP FLG
 SKE STAR
 BRU *+2
 BRU SPCODE
 SKE TYPT23
 BRU *+2
 BRU TYP CON
 SKE TYPT27
 BRU *+2
 BRU FILGAM
 SKE T43
 BRU *+2
 BRU TYPL OG
 SKE TC30
 BRU *+2
 BRU PHEADR
 SKE TC63
 BRU *+2
 BRU PUNCH
 SKE TC62
 BRU *+2
 BRU TRMRCD

CARRIAGE CONTROL CHARACTER
 OCTAL DIGIT
 INSTRUCTION
 PERIOD OR COMMA
 OTHER

RAIC1130
 RAID1140
 RAIC1150
 RAIC1160
 RAID1170
 RAIC1180
 RAID1190
 RAIC1200
 RAIC1210
 RAID1220
 RAIC1230
 RAID1240
 RAIC1250
 RAIC1260
 RAID1270
 RAIC1280
 RAIC1290
 RAIC1300
 RAID1310
 RAID1320
 RAIC1330
 RAID1340
 RAIC1350
 RAID1360
 RAID1370
 RAIC1380
 RAID1390
 RAIC1400
 RAIC1410
 RAID1420
 RAIC1430
 RAID1440
 RAIC1450
 RAID1460
 RAIC1470
 RAIC1480
 RAID1490

	SKE	T51		RAIC1500
	BRU	TYPC1J		RAID1510
TYPC1J	CLR			RAID1520
	STA	TYPW		RAIC1530
	STA	TYPW+1		RAID1540
	STA	TYPWX		RAIC1550
	*			RAID1560
	* SECOND, ETC. CHARACTERS			RAIC1570
	*			RAIC1580
TYPC2	BRM	TYPIN		RAID1590
	BRU	*+1.2		RAIC1600
	BRU	TYPC1C		RAID1610
	BRU	TYPC2D		RAID1620
	BRU	TYPC2O		RAIC1630
	BRU	TYPC2P		RAID1640
	BRU	TYPC2O		RAIC1650
TYPC2D	LDX	TYPWX		RAID1660
	LDA	TYPW.2		RAIC1670
	LSH	3		RAIC1680
	STA	TYPW.2		RAID1690
	BRU	TYPC2		RAIC1700
TYPC2C	LDA	TYPFLG		RAID1710
	SKE	T24		RAIC1720
	BRU	TYPC2X		RAIC1730
	LDA	TYPCHR		RAID1740
	LDX	ZERO		RAIC1750
	SKG	T21	A	RAIC1760
	BRU	TYPC2T		RAIC1770
	SKG	T31	I	RAIC1780
	BRU	TYPC2T+1		RAID1790
	SKG	T43	L	RAIC1800
	BRU	TYPC2T+2		RAID1810
	SKG	T62	S	RAIC1820
	BRU	TYPC2T+3		RAIC1830
	SKG	T63	T	RAID1840
	BRU	TYPC2T+4		RAIC1850
TYPC2X	BRM	TYPXX		RAID1860

	BRU	TYP C2
TYP C2 T	EAX	1,2
	EAX	1,2
	EAX	1,2
	EAX	1,2
	LDA	TYP C2 A,2
	ADD*	TYP C2 B,2
	STA	TYP W
	LDA	TYP C2 A,2
	ADD*	TYP C2 C,2
	STA	TYP W+1
	BRU	TYP DMP
TYP C2 A	NOP	
TYP C2 C	NOP	
TYP C2 P	SKG	D27
	BRU	TYP P C2
	LDA	D1
	STA	TYP W X
	BRU	TYP C2
TYP P C2	BRM	TYP I N
	BRU	*+1,2
	BRU	TYP D0
	BRU	TYP P C2
	BRU	TYP P C2
	BRU	TYP P C2
	BRU	TYP P C2
*		
* DO IT		
*		
TYP D0	BRM	OUT CRT
	LDA	TYP FLG
	SKG	T21
	BRU	TYP ADS
	SKG	T24
	BRU	TYP DMP
	SKG	T25
	BRU	TYP ENT

RAID1870
 RAIC1880
 RAIC1890
 RAIC1900
 RAIC1910
 RAIC1920
 RAIC1930
 RAID1940
 RAIC1950
 RAID1960
 RAID1970
 RAIC1980
 RAID1990
 RAIC2000
 RAID2010
 RAIC2020
 RAIC2030
 RAID2040
 RAIC2050
 RAID2060
 RAIC2070
 RAIC2080
 RAID2090
 RAIC2100
 RAIC2110
 RAIC2120
 RAIC2130
 RAID2140
 RAIC2150
 RAID2160
 RAIC2170
 RAID2180
 RAIC2190
 RAIC2200
 RAID2210
 RAIC2220
 RAIC2230

```

      SKG      T26
      SKG      T41
      BRU      TYPJMP
*
* ADDRESS STOP *
*
TYPADS LDA      ZERO
      SKE      MASTPA
      BRU      *+2
      BRU      TYPSTA
      SKE      MASTPS
      BRU      TYPSTA
      LDA      MASTPI
      STA*     MASTPA
TYPSTA LDA      TYPW
      STA      MASTPA
      LDA      TYPW+1
      STA      MASTPS
      SKE      ZERO
      BRU      TYPC1
      BRM      ASTOPS
      BRU      TYPC1
*
* DUMP
*
TYPDMP LCA      TYPW
      MRG      TYPNOP
      STA      *+5
      LDA      TYPW+1
      MRG      TYPNOP
      STA      *+3
      BRM      JPHDMP
      HLT
      HLT
      BRU      TYPC1
*
* ENTER A WORD

```

```

RAID2240
RAIC2250
RAID2260
RAID2270
RAIC2280
RAIC2290
RAIC2300
RAIC2310
RAID2320
RAIC2330
RAID2340
RAIC2350
RAIC2360
RAID2370
RAIC2380
RAID2390
RAIC2400
RAID2410
RAID2420
RAIC2430
RAID2440
RAIC2450
RAID2460
RAID2470
RAIC2480
RAIC2490
RAIC2500
RAID2510
RAID2520
RAIC2530
RAID2540
RAIC2550
RAID2560
RAID2570
RAIC2580
RAID2590
RAIC2600

```

*			RAIC2610
TYPENT	LDX	TYPW	RAID2620
	LDA	TYPW+1	RAIC2630
	STA	0.2	RAIC2640
	MIN	TYPW	RAID2650
	CLR		RAIC2660
	STA	TYPW+1	RAIC2670
	BRU	TYPC2	RAID2680
*			RAID2690
* JUMP			RAID2700
*			RAIC2710
TYPJMP	LDX	TYPW	RAIC2720
	BRU	0.2	RAIC2730
*			RAID2740
* GET A CHARACTER			RAID2750
*			RAIC2760
TYPIN	PZE	**	RAID2770
	DIR		RAIC2780
	RKBW	1.1	RAID2790
	WIM	TYPCHR	RAID2800
	LDA	TYPCHR	RAIC2810
	ETR	T77	RAID2820
	STA	TYPCHR	RAIC2830
	LDX	TYPCHR	RAID2840
	LDX	TYPTAB,2	RAIC2850
	CLR		RAIC2860
	LDA	TYPCHR	RAID2870
	RSH	3	RAIC2880
	LDA	TYPCHR	RAID2890
	BRR	TYPIN	RAIC2900
*			RAIC2910
* FUNCTIONS			RAID2920
*			RAIC2930
TYPCON	LDX	TYPXSV	RAID2940
	BRR	TYPRTN	RAID2950
FILGAM	BRM	GMAFIL	RAIC2960
	BRM	OUTCRT	RAID2970

```

        BRM      OUTCRT
        BRU      TYP C1
TYPLCG BRM      OUTSP4
        BRM      TYP IN
        SKE      TYPT52
        BRU      *-2
        BRU      TYP C1
TYPT52 OCT      52
*
* DELETE CHARACTER
*
TYPXX  PZE      **
        ECM      20004
        EOM      03241
        MIW      TYPQX
        EOM      14000
        SKS      21000
        BRU      *-1
        BRR      TYPXX
*
* CONSTANTS AND STORAGE
*
TYPQX  OCT      32131212
TYPT23 OCT      23
TYPT27 OCT      27
TYPNOP NCP      0
TYPCHR BSS      1
TYPFLG BSS      1
TYPXSV BSS      1
TYPWX  BSS      1
TYPW   BSS      2
*
* CHARACTER KEY TABLE
*
TYPTAB OCT      1
        OCT      1
        OCT      1

```

```

00 - ZERO
01 - 1
02 - 2

```

```

RAIC2980
RAIC2990
RAID3000
RAIC3010
RAIC3020
RAIC3030
RAIC3040
RAIC3050
RAIC3060
RAID3070
RAIC3080
RAIC3090
RAID3100
RAIC3110
RAIC3120
RAIC3130
RAIC3140
RAID3150
RAIC3160
RAID3170
RAIC3180
RAIC3190
RAID3200
RAIC3210
RAID3220
RAIC3230
RAIC3240
RAID3250
RAID3260
RAID3270
RAIC3280
RAIC3290
RAID3300
RAIC3310
RAIC3320
RAID3330
RAIC3340

```

OCT	1	03 - 3	RAIC3350
OCT	1	04 - 4	RAIC3360
OCT	1	05 - 5	RAIC3370
OCT	1	06 - 6	RAID3380
OCT	1	07 - 7	RAIC3390
OCT	4	10	RAID3400
OCT	4	11	RAIC3410
OCT	0	12 - SPACE	RAIC3420
OCT	4	13	RAIC3430
OCT	4	14	RAIC3440
OCT	4	15	RAID3450
OCT	4	16	RAIC3460
OCT	4	17	RAIC3470
OCT	4	20	RAID3480
OCT	2	21 - A %ADR STOP<	RAIC3490
OCT	4	22	RAID3500
OCT	2	23 - C %CONTINUE<	RAIC3510
OCT	2	24 - D %DUMP<	RAIC3520
OCT	2	25 - E %ENTER<	RAID3530
OCT	2	26 - F %FETCH<	RAIC3540
OCT	2	27 - G %GAMMA FILL<	RAID3550
OCT	2	30	RAIC3560
OCT	4	31	RAIC3570
OCT	0	32 - BACKSPACE	RAID3580
OCT	3	33 - PERIOD	RAIC3590
OCT	4	34	RAIC3600
OCT	4	35	RAIC3610
OCT	4	36	RAID3620
OCT	4	37	RAID3630
OCT	4	40	RAIC3640
OCT	2	41 - J %JUMP<	RAID3650
OCT	4	42	RAIC3660
OCT	2	43 - L %LOG<	RAID3670
OCT	4	44	RAID3680
OCT	4	45	RAIC3690
OCT	4	46	RAID3700
OCT	2	47 - P %PROCEED<	RAIC3710

OCT	4	50	RAIC3720
OCT	2	51 - R %RESTART<	RAID3730
OCT	0	52 - CARRIAGE RETURN	RAIC3740
OCT	4	53	RAID3750
OCT	2	54 - SPECIAL CODES	RAID3760
OCT	4	55	RAIC3770
OCT	4	56	RAID3780
OCT	4	57	RAIC3790
OCT	4	60	RAIC3800
OCT	4	61	RAID3810
OCT	2	62	RAIC3820
OCT	2	63	RAID3830
OCT	4	64	RAIC3840
OCT	4	65	RAIC3850
OCT	4	66	RAID3860
OCT	4	67	RAIC3870
OCT	4	70	RAID3880
OCT	4	71	RAIC3890
OCT	0	72 - TAB	RAID3900
OCT	3	73 - COMMA	RAID3910
OCT	4	74	RAIC3920
OCT	4	75	RAID3930
OCT	4	76	RAIC3940
OCT	4	77	RAIC3950
*			RAIC3960
* END OF REM TYPE			RAIC3970
TC30	OCT	30	RAID3980
TC62	OCT	62	RAIC3990
TC63	OCT	63	RAIC4000
STAR	OCT	54	RAIC4010
PAGE			RAIC4020
* PRINT ROUTINES			RAID4030
*			RAIC4040
* OUTPUT 1 CHARACTER PER WORD			RAID4050
OUT1PW	PZE	**	RAIC4060
STA	TEMP		RAIC4070
TYPW	1.1		RAID4080

MIW	TEMP		RAIC4090
TOPW			RAIC4100
BRTW			RAID4110
BRU	*-1		RAIC4120
BRR	OUT1PW		RAIC4130
* OUTPUT 4 CHARACTERS PER WORD			RAIC4140
CUT4PW	PZE	**	RAID4150
STA	TEMP		RAID4160
TYPW	1,4		RAIC4170
MIW	TEMP		RAIC4180
TOPW			RAIC4190
BRTW			RAID4200
BRU	*-1		RAIC4210
BRR	OUT4PW		RAIC4220
* OUTPUT A CARRIAGE RETURN			RAID4230
OUTCRT	PZE	**	RAIC4240
LCA	T52M		RAIC4250
BRM	OUT1PW		RAIC4260
BRR	OUTCRT		RAIC4270
* OUTPUT 4 SPACES			RAID4280
OUTSP4	PZE		RAIC4290
LDA	A4SP		RAID4300
BRM	OUT4PW		RAIC4310
BRR	OUTSP4		RAIC4320
* OUTPUT OCTAL WORD			RAID4330
TWORD	PZE	**	RAIC4340
XAB			RAIC4350
LDA	D7		RAIC4360
STA	TWORDX	COUNTER	RAIC4370
TWORD1	LCA	ZERO	RAID4380
LSH	3		RAIC4390
STB	TWORDX+1		RAIC4400
LSH	18		RAIC4410
BRM	OUT1PW		RAIC4420
LDB	TWORDX+1		RAID4430
SKR	TWORDX	TEST	RAIC4440
BRU	TWORD1	CONTINUE	RAIC4450

BRR	TWORD	RETURN	RAID4460
TWORDX BSS	2		RAIC4470
* OUTPUT OCTAL ADDRESS			RAIC4480
TADDR PZE	**		RAIC4490
LSH	9		RAID4500
XAB			RAID4510
LCA	D4		RAIC4520
STA	TADDRX	COUNTER	RAID4530
TADDR1 LDA	ZERO		RAIC4540
LSH	3		RAID4550
STB	TADDRX+1		RAID4560
LSH	18		RAIC4570
BRM	OUT1PW		RAID4580
LDB	TADDRX+1		RAIC4590
SKR	TADDRX		RAID4600
BRU	TADDR1	CONTINUE	RAIC4610
BRR	TADDR	RETURN	RAIC4620
TADDRX BSS	2		RAID4630
* DUMP CORE BLOCK			RAID4640
JPHDMP PZE	**		RAID4650
MIN	JPHDMP		RAIC4660
LDA*	JPHDMP		RAIC4670
STA	JPHDPX	START	RAID4680
MIN	JPHDMP		RAIC4690
LDA*	JPHDMP		RAID4700
STA	JPHDPX+1	END	RAIC4710
BRM	OUTCRT	CARRIAGE RETURN	RAIC4720
JPHDP1 LDA	JPHDPX		RAID4730
BRM	TADDR	LOCATION	RAIC4740
LDA	A4SP		RAID4750
BRM	OUT4PW	4 SPACES	RAIC4760
JPHCP2 LCA*	JPHDPX		RAID4770
BRM	TWORD	CONTENTS	RAID4780
LDA	A4SP		RAIC4790
BRM	OUT4PW	4 SPACES	RAID4800
MIN	JPHDPX	ADVANCE	RAIC4810
LCA	JPHDPX		RAID4820

	SKG	JPHDPX+1	TEST END	RAID4830
	BRU	*+3	CONTINUE	RAIC4840
	BRM	OUTCRT	DONE, CR	RAIC4850
	BRR	JPHDMP	RETURN	RAIC4860
	ETR	D7		RAIC4870
	SKE	ZERO		RAIC4880
	BRU	*+2		RAIC4890
	BRU	JPHDP1-1	NEW LINE	RAIC4900
	SKE	D4		RAIC4910
	BRU	JPHDP2	NO NEW LINE	RAIC4920
	BRU	JPHDP1-1	NEW LINE	RAID4930
JPHCPX	BSS	2		RAIC4940
A4SP	BCD	4,		RAID4950
	PAGE			RAIC4960
* SKE				RAID4970
POPSKE	STB	POPB		RAID4980
	LDB	POP7S		RAIC4990
	SKM*	0		RAID5000
	BRU	*+2		RAIC5010
	MIN	0		RAIC5020
	LDB	POPB		RAID5030
	BRR	0		RAIC5040
* SKR				RAIC5050
POPSKR	OCT	06040000	MDC* 0	RAIC5060
	SKN*	0		RAIC5070
	BRR	0		RAID5080
	MIN	0		RAIC5090
	BRR	0		RAIC5100
POPB	PZE	**		RAIC5110
POP7S	OCT	77777777		RAIC5120
ZERC	OCT	0		RAID5130
ZNAME	PZE			RAIC5140
SKE	PCPD	17600000		RAIC5150
SKR	POPD	17700000		RAIC5160
T43	OCT	43		RAIC5170
T51	OCT	51		RAID5180
T24	OCT	24		RAIC5190

T21	OCT	21	RAIC5200
T31	OCT	31	RAID5210
T62	OCT	62	RAIC5220
T63	OCT	63	RAIC5230
D27	DEC	27	RAIC5240
D1	OCT	1	RAID5250
T25	OCT	25	RAID5260
T26	OCT	26	RAIC5270
T77	OCT	77	RAID5280
TEMP	PZE		RAIC5290
T52M	OCT	52000000	RAID5300
D7	OCT	7	RAID5310
D4	OCT	4	RAIC5320
GMAFIL	NOP		RAID5330
TYPC2B	NOP		RAIC5340
T41	OCT	41	RAID5350
	PAGE		RAIC5360
*			RAID5370
* PAPER TAPE PUNCHING ROUTINES			RAID5380
*			RAIC5390
PUNCH	CLR		RAID5400
	STA	SW	RAID5410
	STA	CNT	RAIC5420
	BRM	WORD	RAID5430
	BRU	NOCR	RAIC5440
*			RAID5450
* CARRIAGE RETURN ENCOUNTERED - ONLY ONE LOCATION TO PUNCH			RAID5460
*			RAIC5470
	LCA	W	RAID5480
	STA	W1	RAIC5490
	MIN	CNT	RAID5500
	BRM	PNCH	RAID5510
	BRM	OUTCRT	RAIC5520
	BRU	TYPC1	RAID5530
NOCR	LDA	W	RAIC5540
	STA	W1	RAID5550
	BRM	WORD	RAID5560

	BRU	*+2
	MDE	SW
	LDA	W1
	LDB	W
	SKG	W
	BRU	*+2
	XAB	
	STA	W1
	STB	W2
	XAB	
	SUB	W1
	STA	CNT
	MIN	CNT
	BRM	PNCH
	SKN	SW
	BRU	PUNCH
	BRU	TYPC1
*		
PNCH	PZE	
	BRTW	
	BRU	*-1
	PPT	1.4
	MIW	W1
	MIW	CNT
LPPNCH	MDE	CNT
	LDA	W1
	SKN	CNT
	BRU	*+2
	BRU	L FNCHP
	MIW	0.2
	MIN	W1
	BRU	LPPNCH
L FNCHP	TOP	0
	BRTW	
	BRU	*-1
	BRR	PNCH
*		

RAIC5570
RAIC5580
RAIC5590
RAIC5600
RAIC5610
RAIC5620
RAIC5630
RAIC5640
RAIC5650
RAIC5660
RAIC5670
RAIC5680
RAIC5690
RAIC5700
RAIC5710
RAIC5720
RAIC5730
RAIC5740
RAIC5750
RAIC5760
RAIC5770
RAIC5780
RAIC5790
RAIC5800
RAIC5810
RAIC5820
RAIC5830
RAIC5840
RAIC5850
RAIC5860
RAIC5870
RAIC5880
RAIC5890
RAIC5900
RAIC5910
RAIC5920
RAIC5930

W1	HLT			RAID5940
W2	HLT			RAIC5950
CNT	HLT			RAID5960
SW	HLT			RAIC5970
*				RAIC5980
*				RAID5990
*				RAIC6000
WORD	PZE			RAID6010
	CLR			RAID6020
	STA	W		RAIC6030
	BRM	TYPIN		RAID6040
	BRU	*+1.2		RAIC6050
	BRU	CARRT1		RAIC6060
	BRU	DIGIT8		RAID6070
	BRU	WORD+3		RAIC6080
	BRU	L1PORC	/ PERIOD OR COMMA	RAID6090
	BRU	WORD+3		RAIC6100
CARRT1	MIN	WORD		RAID6110
	BRM	OUTCRT		RAIC6120
	BRR	WORD		RAIC6130
L1FCRC	SKE	TYPT33		RAID6140
	BRR	WORD		RAIC6150
	BRM	TYPIN	/ PERIOD	RAID6160
	BRU	*+1.2		RAIC6170
	BRU	CARRT1		RAIC6180
	BRU	*-3		RAID6190
	BRU	*-4		RAIC6200
	BRU	*-5		RAID6210
	BRU	*-6		RAIC6220
DIGIT8	LDA	W		RAIC6230
	LCY	3		RAID6240
	STA	W		RAIC6250
	BRU	WORD+3		RAID6260
W	HLT	0		RAIC6270
TYPT33	OCT	33		RAIC6280
*				RAID6290
*				RAIC6300

*			RAID6310
	OCT	0	RAID6320
PHEADR	LDX	*-1	RAID6330
	EAX	DATA	RAID6340
	STX	DATA-1	RAID6350
	LDX	DATA-2	RAID6360
	BRM	OUTCRT	RAID6370
	BRTW		RAID6380
	BRU	*-1	RAID6390
	EOM	C0644	RAID6400
	M[W*	DATA-1	RAID6410
	MIN	DATA-1	RAID6420
	BRX	*-2	RAID6430
	TOP	0	RAID6440
	BRTW		RAID6450
	BRU	*-1	RAID6460
	BRU	TYPC1	RAID6470
	OCT	77777751	RAID6480
	HLT		RAID6490
DATA	OCT	23200012	RAID6500
	OCT	04100002	RAID6510
	OCT	07100011	RAID6520
	OCT	03217757	RAID6530
	OCT	06100005	RAID6540
	OCT	04100005	RAID6550
	OCT	00117757	RAID6560
	OCT	77777761	RAID6570
	OCT	04021000	RAID6580
	OCT	00117762	RAID6590
	OCT	0	RAID6600
	OCT	03217776	RAID6610
	OCT	03217777	RAID6620
	OCT	06017777	RAID6630
	OCT	05317777	RAID6640
	OCT	00117770	RAID6650
	OCT	00117757	RAID6660
	OCT	04021000	RAID6670

OCT	00117773		RAIC6680
OCT	0		RAID6690
OCT	03257776		RAIC6700
OCT	06117776		RAID6710
OCT	00117764		RAIC6720
*			RAIC6730
*			RAID6740
*			RAIC6750
TRMRCD CLR			RAID6760
STA	DMYBRU-1		RAID6770
BRM	TYP IN		RAIC6780
BRU	*+1,2		RAIC6790
BRU	PNCHTR	/ PUNCH THE TERMINATOR RECORD	RAIC6800
BRU	*+4		RAID6810
BRU	TRMRCD+2		RAID6820
BRU	TRMRCD+2		RAIC6830
BRU	TRMRCD+2		RAID6840
LDA	DMYBRU-1		RAIC6850
LCY	3		RAID6860
STA	DMYBRU-1		RAID6870
BRU	TRMRCD+2		RAIC6880
FNCHTR BRM	OUT CRT		RAID6890
LDA	DMYBRU-1		RAIC6900
MRG	DMYBRU		RAIC6910
STA	DMYBRU-1		RAIC6920
BRTW			RAIC6930
BRU	*-1		RAID6940
EOM	02644		RAIC6950
MIW	RECRDS		RAID6960
MIW	RECRDS+1		RAIC6970
MIW	RECRDS+2		RAIC6980
MIW	RECRDS+3		RAID6990
MIW	RECRDS+4		RAIC7000
MIW	RECRDS+5		RAIC7010
TOP	0		RAIC7020
BRTW			RAIC7030
BRU	*-1		RAID7040


```

      BRU      TYP C1
RECRDS OCT     17757
      OCT     4
      OCT     00200000 / DSC      0
      OCT     04021000
      OCT     00117760
      OCT     0      / FOR THE BRU
DMYERU BRU     0
SPCODE BRM     TYP IN
      BRU     *+1,2
      BRU     SPCODE
      BRU     SPCODE
      BRU     SPCDL1
      BRU     SPCODE
      BRU     SPCODE
SPCDL1 SKE     SPCD24
      BRU     *+3
      BRM     DUMP
      BRU     TYP C1
      BRM     OUTCRT
      BRU     TYP C1
SPCD24 OCT     24
DUMP   PZE

```

*
*
*
*
*

THIS ROUTINE WILL DUMP PORTIONS OF STORAGE TO THE PDP 15.
IT DOES SO BY SENDING GROUPS OF 2000(OCTAL) WORDS OF STORAGE
FROM THE SDS 910.

```

      BRM     WORD
      BRU     *+5      NC CARRIAGE RETURN
      LDA     DUMPEC
      BRM     OUT4PW
      BRM     OUTCRT
      BRU     DUMP
      LDA     W
      ETR     DUMP37
      STA     W1

```

```

RAID7050
RAID7060
RAID7070
RAID7080
RAID7090
RAID7100
RAID7110
RAID7120
RAID7130
RAID7140
RAID7150
RAID7160
RAID7170
RAID7180
RAID7190
RAID7200
RAID7210
RAID7220
RAID7230
RAID7240
RAID7250
RAID7260
RAID7270
RAID7280
RAID7290
RAID7300
RAID7310
RAID7320
RAID7330
RAID7340
RAID7350
RAID7360
RAID7370
RAID7380
RAID7390
RAID7400
RAID7410

```

STA	W2		RAIC7420
BRM	WORD		RAIC7430
NCP	0		RAID7440
LCA	W		RAIC7450
ETR	DUMP37		RAID7460
SKG	W1		RAIC7470
BRU	*+3		RAIC7480
STA	W2		RAID7490
BRU	*+2		RAIC7500
STA	W1		RAID7510
LDA	W2		RAIC7520
SUB	W1		RAID7530
STA	CNT		RAIC7540
MIN	CNT		RAIC7550
LDB	W2		RAIC7560
LCY	6		RAIC7570
LCA	W1		RAIC7580
LCY	6		RAID7590
STA	DUMPBF		RAIC7600
STB	DUMPBF+1		RAIC7610
EOM	32307	SET THE 910 INITIATE AND	RAIC7620
POT	DUMP50	DMA BITS.	RAID7630
ECM	32301		RAID7640
POT	DUMPBF		RAIC7650
ECM	32302		RAID7660
POT	DUMPBF+1		RAIC7670
ECM	32303		RAIC7680
PCT	CNT		RAID7690
CUMPLA LDA	DUMPC2		RAIC7700
SKG	CNT	CNT<768?	RAID7710
BRU	DUMPLB	YES	RAIC7720
*			RAIC7730
* OUTPUT A COMPLETE BLOCK			RAID7740
*			RAIC7750
LCA	DUMPC1		RAIC7760
STA	DUMPI		RAID7770
CUMFLC LDX	W1		RAIC7780

```

BRM      DUMPIT
LDA      W1
ADD      DUMP3
STA      W1
MDE      DUMPI
SKN      DUMPI
BRU      DUMPLC
LDA      CNT
SUB      DUMPC2
STA      CNT
BRU      DUMPLA
*
*      TRY AGAIN
*
*      DUMPIT WOLL OUTPUT THREE WORDS
*
DUMPIT >ZE
    ECM      36300
    SKS      30001
    BRU      *+2
    BRU      *-3
    ECM      32301
    POT      0.2
    ECM      32302
    POT      1.2
    ECM      32303
    POT      2.2
    BRR      DUMPIT
*
*
*      ONLY A PIESE LEFT
*
DUMPLB LDA      DUMP1
    SKG      CNT
    BRU      *+2
    BRR      DUMP
    LDA      DUMPC1

```

```

RAID7790
RAID7800
RAIC7810
RAID7820
RAIC7830
RAIC7840
RAIC7850
RAIC7860
RAID7870
RAID7880
RAIC7890
RAIC7900
RAIC7910
RAID7920
RAIC7930
RAID7940
RAIC7950
RAIC7960
RAID7970
RAIC7980
RAID7990
RAIC8000
RAIC8010
RAID8020
RAIC8030
RAID8040
RAID8050
RAIC8060
RAID8070
RAIC8080
RAID8090
RAID8100
RAIC8110
RAIC8120
RAIC8130
RAIC8140
RAID8150

```

	STA	DUMPI		RAIC8160
DUMPLD	LDA	CNT		RAID8170
	SKG	DUMP2		RAID8180
	BRU	DUMPLE		RAIC8190
	MDE	DUMPI		RAID8200
	LDX	W1		RAIC8210
	BRM	DUMPIT		RAIC8220
	LDA	W1		RAID8230
	ADD	DUMP3		RAIC8240
	STA	W1		RAID8250
	LDA	CNT		RAIC8260
	SUB	DUMP3		RAID8270
	BRU	DUMPLD		RAID8280
DUMPLE	CLR		PUT OUT LAST PIECES	RAIC8290
	STA	DUMPBFB+1		RAID8300
	STA	DUMPBFB+2		RAIC8310
	LDA	DUMP1		RAID8320
	SKG	CNT		RAIC8330
	BRU	DUMPLF		RAIC8340
	LDA	CNT		RAID8350
	SKG	DUMPBFB+1		RAIC8360
	BRU	DUMPLG		RAID8370
	LDA	0.2		RAIC8380
	STA	DUMPBFB		RAIC8390
	LDA	1.2		RAID8400
	STA	DUMPBFB+1		RAIC8410
	BRU	DUMPSN		RAID8420
DUMPLF	LDA	0.2		RAIC8430
	STA	DUMPBFB		RAID8440
DUMPSN	EAX	DUMPBFB		RAID8450
	BRM	DUMPIT		RAIC8460
	CLR			RAID8470
	STA	DUMPBFB		RAIC8480
	STA	DUMPBFB+1		RAIC8490
*				RAID8500
*	PUT OUT ZEROES			RAIC8510
*				RAID8520

```

DUMPLH MDE      DUMPI
          SKN      DUMPI
          BRU      *+2
          BRR      DUMP          GC HOME
DUMPLG BRM      DUMPI
          BRU      DUMPLH

```

*

* CCNSTANTS AND TEMPORARY STORAGE ...

*

```

DUMPEC OCT      21522 733
DUMP37 OCT      37777
DUMPW1 HLT
DUMPI  HLT
DUMPC1 OCT      377
DUMPC2 OCT      1400
DUMP1  OCT      1
DUMP2  OCT      2
DUMF3  OCT      3
DUMP50 OCT      50
DUMPBF BSS      3
          END

```

```

RAID8530
RAIC8540
RAID8550
RAIC8560
RAIC8570
RAID8580
RAIC8590
RAID8600
RAIC8610
RAIC8620
RAID8630
RAIC8640
RAID8650
RAIC8660
RAIC8670
RAID8680
RAIC8690
RAID8700
RAIC8710
RAIC8720
RAID8730

```