PDT11/150

SYSTEM EXERCISER

CVKDAB0

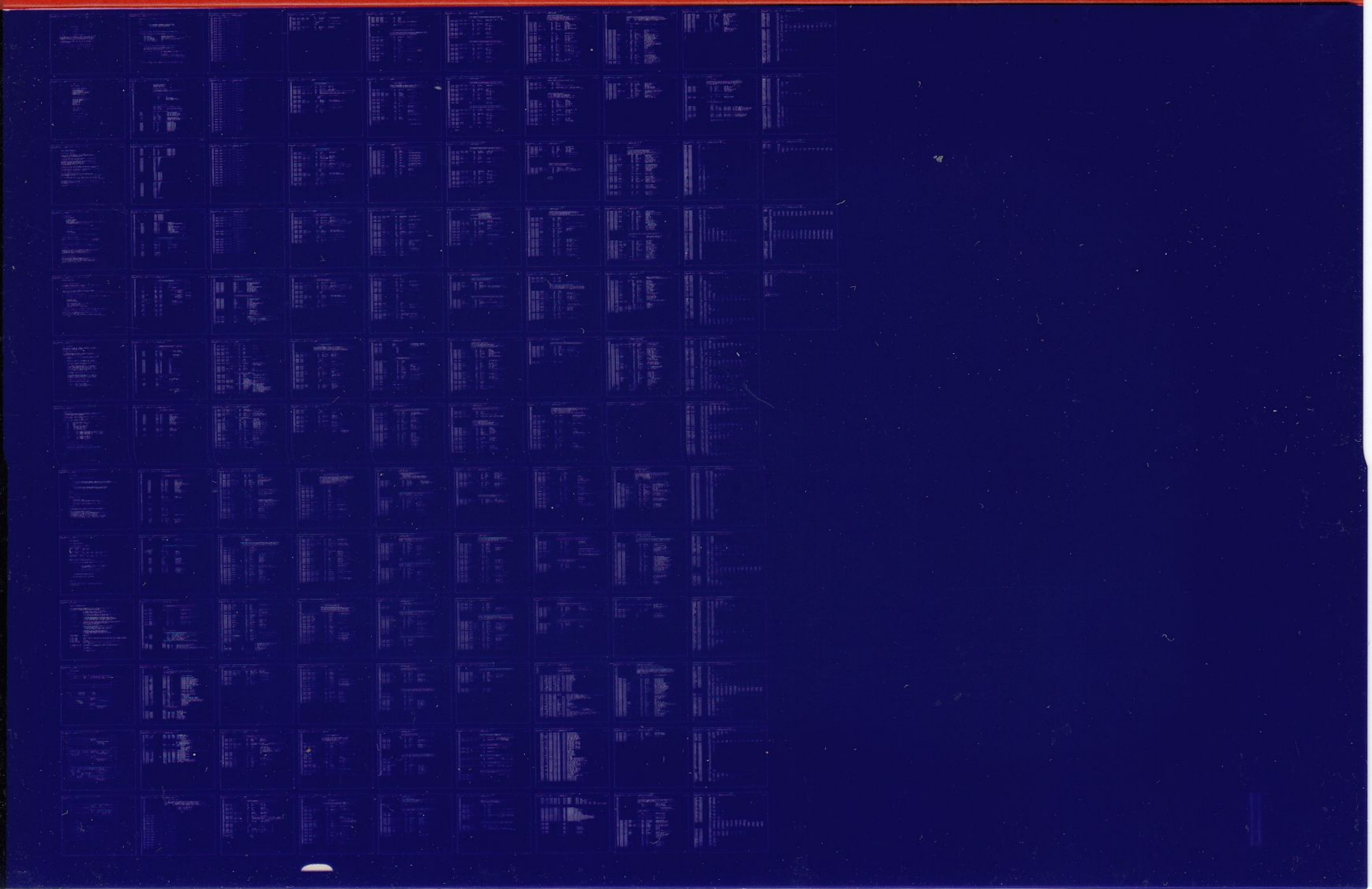
AH-F239B-MC

FICHE 1 OF 1

JAN 1979

digital

MADE IN USA



```
.REM %
```

IDENTIFICATION

PRODUCT CODE: AC-F238B-MC

PRODUCT NAME: CVKDAB0 PDT11/150 SYSTEM EXERCISER

PRODUCT DATE: NOV 1978

MAINTAINER: DIAGNOSTIC ENGINEERING

TABLE OF CONTENTS

1.0 GENERAL PROGRAM INFORMATION.

1.1 PROGRAM PURPOSE (ABSTRACT).

THE PDT-11/150 SYSTEM EXERCISER TESTS THE PROCESSORS ABILITY
TO OPERATE ALL ITS PERIPHERALS IN INTERRUPT MODE AT THE SAME TIME
WITH EMPHASIS ON THE DISK SUBSYSTEMS.

IT SHOULD BE NOTED THAT IT IS NOT A DIAGNOSTIC OF THE PERIPHERALS
BUT A SERIES OF SYSTEM INTERACTION TESTS.

THE PROGRAM IS DEFAULTED TO EXERCISE THE CLUSTER TERMINALS & THE
COMM PORT IN INTERNAL LOOPBACK (MAINT) MODE .
THE DEFAULT CAN BE CHANGED TO EXERCISE ANY OF THE CLUSTER TERMINALS
AND/OR THE COMM PORT IN EXTERNAL LOOPBACK.
SEE PROGRAM OPTIONS SEC. 2.4.

TESTING OF THE ACTUAL CLUSTER TERMINALS OR COMM DEVICE IS NOT PERFORMED.
THESE DEVICES CAN ONLY BE TESTED EITHER IN EXT. OR INT. LOOPBACK.

THE PRINTER LOGIC WILL BE EXERCISED IF SIZING DETERMINES IT TO BE PRESENT
OR IF DATA TERM RDY IS ASSERTED ON ITS CONNECTOR.

THE CONSOLE TERMINAL IS NOT TESTED.

AFTER THE MEMORY HAS BEEN TESTED & ALL THE PERIPHERALS ARE
BEING EXERCISED & INTERRUPTING AT RANDOM, THE DISK SUBSYSTEM TESTS WILL BEGIN.
SEE TEST SUMMARY SEC. 7.0.

NOTE: IF RUNNING UNDER APT, THE SYNC COMM LOGIC WILL BE TESTED ONLY ON
THE 1'ST PASS. THIS IS SO APT 'BREAKS' SO NOT CAUSE FALSE SYNC COMM ERRORS.

THE PROGRAM CONTAINS A UTILITY WHICH CAN COPY THE SYSTEM EXERCISER DISK
FROM DX0 ONTO A SCRATCH DISK IN DX1.
THIS ENABLES THE OPERATOR TO CREATE BACKUP COPIES OF THE EXERCISER DISK.
SEE SECTION 2.8.

THE PROGRAM ALSO CONTAINS A COMPATABILITY TESTING OPTION.
SEE SEC. 2.7.

1.2 SYSTEM REQUIREMENTS.

HARDWARE REQUIREMENTS:

PDT-11/150 SYSTEM
8K MEMORY - MINIMUM
CONSOLE TERMINAL
EXTERNAL LOOPBACK CONNECTORS (OPTIONAL) FOR COMM & CLUSTER TERMINAL PORTS.
(SEE PROGRAM OPTIONS SEC. 2.4.
V1-100 CONSOLE MUST BE SETUP FOR JUMP SCROLL.

SOFTWARE REQUIREMENTS:

THIS EXERCISER IS DESIGNED TO RUN IN ANY OF THE FOLLOWING WAYS:

STAND ALONE
WITH APT MONITOR
WITH RT-11 MONITOR

IT IS NOT DESIGNED TO RUN WITH THE DIAGNOSTIC SUPERVISOR.

1.3 RELATED DOCUMENTS AND STANDARDS.

DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS	175-003-009-02
APT	MD-11-DZZMA
SYSMAC	MD-11-DZQAC

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

THIS EXERCISER ASSUMES THAT THE POWER UP SELF TEST RUNS ERROR FREE.

1.5 ASSUMPTIONS

THIS EXERCISER ASSUMES THAT THE OPERATOR HAS MODIFIED THE DEVICE MAP
($DEVM) AT LOC. 1246 IF THE DEFAULTS DO NOT AGREE WITH THE ACTUAL
SYSTEM CONFIGURATION. SEE PROGRAM OPTIONS SEC. 2.4.
THE PROGRAM ALSO ASSUMES THE SOFTWARE SWITCH REGISTER IS
PROPERLY SET UP (SWREG) AT LOC. 176. SEE SEC. 2.3.

1.6 RUNNING SYSTEMS PROGRAMS

UNLESS THE SYSTEMS PROGRAMS ARE KNOWN TO INITIALIZE THE BAUD
RATES OF EACH DEVICE THRU THE PARAMETER REGISTER,
THE OPERATOR SHOULD POWER DOWN & UP AGAIN WHEN FINISHED RUNNING THIS EXERCISER.
THIS IS TO RESTORE THE PDT-11/150 TO ITS DEFAULT BAUD RATES.
OTHERWISE, THE DEVICES WILL ATTEMPT TO RUN AT WHATEVER BAUD RATES
WERE LAST USED BY THIS EXERCISER.

2.0 OPERATING INSTRUCTIONS.

2.1 LOADING AND STARTING PROCEDURES.

USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE BINARY FORMATTED PAPER TAPE.
THE PROGRAM WILL AUTO-START AFTER LOADING.

THE SYSTEM EXERCISER DISK WILL AUTO-START AFTER BOOTING.

EXAMPLE OF STARTUP ASSUMING PRINTER NOT PRESENT, 24K MEMORY,
& TERM #2 NOT TO BE TESTED, COMM PORT EXT. LOOPBACK, TERM 1 & 3 INT. LOOPBACK.
(20007 IN $DEVM, SEE SEC. 1.5):

```
        CVKDAB  PDT-11/150 SYSTEM EXERCISER

        SWR = 000000    NEW = 110000 <CR>       (HALT ON ERR, ENABLE PERFORMANCE REPORTS)

        DEVM = 000017   NEW = <CR>              (KEEP DEFAULTS)

        24K MEMORY PRESENT
        PRINTER NOT PRESENT
        TERM #2 TESTING DROPPED

        INSERT SCRATCH DISKS, TYPE 'P' FOR NORMAL TESTING
        '240G' FOR NORMAL RESTARTS
        '250G' TO COPY SYS EXERCISER DISK
        '260G' FOR COMPATABLITIY  PASS 1: WRITE
        '270G' FOR                PASS 2: READ

(PROGRAM HALTS & WAITS FOR 'P' & CONTINUES....ABOVE NOTE & HALT OMITTED UNDER APT)

THE PROGRAM WILL BEGIN ACTUAL TESTING AT THIS POINT. (SEE SUMMARY OF TESTS SEC. 7.0)

DURING THE TESTS, THE PROGRAM WILL PRINT PROGRESS REPORTS (SEF SEC. 4.1) IF
BIT 12 IS SET IN THE SWREG (SEE SEC 2.3).
END OF PASS & TOTAL ERROR COUNT IS PRINTED AT THE CONCLUSION OF TESTING.
THE PROGRAM JUMPS TO THE BEGINNING & THE ABOVE SEQUENCE IS REPEATED UNTIL HALTED.
```

2.2 SPECIAL ENVIRONMENTS.

THIS DIAGNOSTIC FOLLOWS THE STANDARD PROCEDURE FOR RUNNING UNDER APT, RT-11 MONITORS, AS DESCRIBED IN THEIR RESPECTIVE PROCEDURES MANUAL AND SYSMAC PACKAGE.

2.3 OPERATIONAL SWITCH SETTINGS

THIS PROGRAM SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH RECISTER (SWREG AT LOC. 176) FROM THE CONSOLE. THIS CAN BE ACCOMPLISHED BY DOING THE FOLLOWING:

1) TYPE CONTROL G <^G>; THIS WILL ALLOW THE CONSOLE TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.

2) THE MACHINE WILL THEN TYPE:' SWR=XXXXXX NEW=' (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)

3) AFTER THE 'NEW=' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE CONSOLE:

A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED). LEADING ZEROS NEED NOT BE TYPED, AND IF MORE THAN 6 DIGITS ARE TYPED THE LAST 6 WILL BE USED. IF A <CR> IS THE FIRS KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.

B) IF A CONTROL U <^U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 3.

C) IF THE INPUT CHARACTER IS NOT ONE OF THE CHARACTERS MENTIONED ABOVE THEN A QUESTION MARK (?) WILL BE TYPED FOLLOWED BY A CARRAGE RETURN AND A LINE FEED SEQUENCE THEN PROCEED FROM STEP 3 (ERASING ALL PREVIOUS INPUT).

4) THE DIAGNOSTIC WILL CONTINUE ON TYPING <CR>.

SOFTWARE SWITCH REGISTER OPTIONS (SWREG)

BIT 15 SET = 100000 = HALT ON ERROR
13 SET = 20000 = INHIBIT ERROR TYPEOUTS
12 SET 10000 = ENABLE PERFORMANCE REPORTS
10 SET 2000 = BELL ON ERROR

H 1

2.4 PROGRAM OPTIONS.

A DEVICE MAP ($DEVM) AT LOCATION 1246 (ENTERED BY A CONTROL-G & CONTROL-C)
IS EXAMINED BY THE PROGRAM TO DETERMINE THE METHOD OF TESTING
THE CLUSTER TERMINALS & THE COMM PORT.

THE DEFAULT VALUE = 000017 (INTERNAL LOOPBACK FOR COMM PORT & CLUSTER TERMINALS)
IT CAN BE CHANGED TO DO THE FOLLOWING:

$DEVM (LOC 1246) ENTERED BY CONTROL-G & C

```
BIT 15 SET=     100000= DROP PRINTER TESTS
     4 SET=      40000= DROP CLUSTER TERM #3 TESTS
    13 SET=      20000=                  #2
    12 SET=      10000=                  #1
    11 SET-       4000= DROP ASYNC COMM TESTS
    10 SET=       2000= DROP  SYNC COMM TESTS
     9 SET=       1000= DROP DRIVE 1 TESTS
     8 SET=        400= DROP DRIVE 0 TESTS
     7 SET=        200= DROP CLOCK TESTS

BIT  3 SET=          10= INT. LOOPBACK (MAINT MODE) FOR COMM PORT (DEFAULT)
     2 SET=           4- INT. LOOPBACK (MAINT MODE) FOR TERM #1   (DEFAULT)
     1 SET=           2= INT. LOOPBACK (MAINT MODE) FOR      #2   (DEFAULT)
     0 SET            1- INT. LOOPBACK (MAINT MODE) FOR      #3   (DEFAULT)

BIT  3 CLR=               EXT. LOOPBACK FOR COMM PORT.
BIT  2 CLR=               EXT. LOOPBACK FOR TERM #1.
BIT  1 CLR                EXT. LOOPBACK FOR TERM #2.
BIT  0 CLR-               EXT. LOOPBACK FOR TERM #3.
```

NOTES:

1. THE CONSOLE IS NOT TESTED.

2. BITS 15-7 SETTINGS WILL OVERRIDE BIT 3-0 SETTINGS.

3. THE PRINTER LOGIC WILL BE TESTED IF BIT 15 = 0 & THE DATA TERM READY
   IS ASSERTED ON THE CONNECTOR. (A PHYSICAL PRINTER IS NOT REQ'D)

4. A VT-50,52 MAY BE USED INSTEAD OF A PRINTER FOR A VISUAL CHECK.

2.5 EXECUTION TIMES.

ASSUMING ALL DEVICES & 2 DISK SUBSYSTEMS PRESENT:

1'ST PASS:

25 - 30 SEC TO STARTUP CLOCK, PRINTER, COMM DEVICE & CLUSTER TERMINALS.
5 MIN TO WRITE, READ & DATA COMPARE ON TRACKS 0-4, 40-44, 72-76(10) ON BOTH DRIVES.
5 - 60 SEC TO DO 10 RANDOM SEEKS ON EACH DISK BETWEEN 1'ST 5 TRACKS.

SUBSEQUENT PASSES:

30 SEC TO STARTUP CLOCK, PRINTER, COMM DEVICE & CLUSTER TERMINALS.
20 MIN TO WRITE, READ & DATA COMPARE ALL TRACKS ON BOTH DISKS.
5 MIN TO DO 500 RANDOM SEEKS ON EACH DISK BETWEEN ALL TRACKS.

2.6 POWER FAIL

THERE IS NO POWER FAIL AUTO-RESTART CAPABILITY IN THE PDT-11.

2.7 COMPATABILTIY TESTING

PASS 1: ENTERED FROM A '260G'.
WILL WRITE ALL TRACKS & SECTORS ON SELECTED DRIVES
& HALT AFTER AN OPERATOR PROMPT. TYPING 'P' WILL BEGIN PASS 2.

PASS 2: ENTERED BY A 'P' AFTER THE ABOVE HALT, OR A '270G'.
WILL PERFORM SEQUENTIAL & RANDOM READS ONLY, ON THE SELECTED DRIVES.

2.8 COPY UTILITY

TO PROVIDE BACKUP DISKS, A UTILITY IS PROVIDED IN THE PROGRAM TO COPY
THE SYSTEM EXERCISER DISK FROM DX0 TO DX1 IN THE FOLLOWING WAY:

1. BOOT THE EXERCISER NORMALLY FROM DX0.
2. ALLOW THE PROGRAM TO HALT AFTER THE WARNING MESSAGE TO USE SCRATCH DISKS.
3. AT THIS POINT, DO A '250G' TO ENTER THE COPY UTILITY.
4. THE OPERATOR WILL BE PROMPTED TO USE A SCRATCH DISK IN DX1 & TYPE 'P' TO PROCEED.
5. WHEN COMPLETED, THERE WILL BE A VERIFYING MESSAGE.
   THE OPERATOR CAN THEN DO A 'P' TO COPY ANOTHER IN THE SAME MANNER,
   OR A '240G' TO BEGIN NORMAL TESTING. (OPERATOR WILL BE PROMPTED)
6. THE COPY UTILITY CAN BE ENTERED AT ANY TIME BY DOING A 'BREAK' & '250G'.

J 1

3.0 ERROR INFORMATION.

3.1 ERROR REPORTING PROCEDURE.

TYPICAL ERROR PRINTOUTS ARE SHOWN BELOW:

TERM #3 DATA COMP ERR

ERROR #	ERR PC	EXPECT	RECVD
000011	006516	000200	000100

DX1 SOFT ERR - DATA COMP

DXTST #	ERR PC	RXCS	RXES	RXSA	EXPECT	RECVD	# RETRIES
000003	010636	100337	000230	003405	100520	100120	4

DX0 HARD ERR - AFTER WRITE CMD

ERROR #	DXTST #	ERR PC	RXCS	RXES	TRACK	SECTOR	#RETRIES
000016	000001	011246	100337	000230	59	18	10

WHERE ALL VALUES TYPED ARE OCTAL EXCEPT :
#RETRIES, TRACK, & SECTOR WHICH ARE IN DECIMAL.

BITS 15,13, & 10 OF THE SWITCH REGISTER (SWREG) CONTROL THE SEQUENCE OF EVENTS AFTER AN ERROR IS CAUGHT.

BIT 15 SET: CAUSES THE PROGRAM TO HALT IN THE ERROR ROUTINE. IF THE PROGRAM IS CONTINUED, IT WILL PROCEED FROM WHERE IT HALTED.

BIT 13 SET: DISABLES THE PRINTING OF THE ERROR MESSAGE.

BIT 10 SET: CAUSES THE BELL TO RING ON ERROR.

THE ERROR ROUTINE SUPPORTS THE CONTROL G <^G> FUNCTION.
REFER TO SECTION 2.3 FOR DETAILS.

NOTE: SINCE THERE ARE NO SPECIFIC TEST NUMBERS, APT WILL REPORT ALL ERRORS AS TEST 0 ERRORS.

3.2 ERROR HALTS.

THE ONLY HALT IN THE EXERCISER IS IN THE ERROR ROUTINE, AND IS EXECUTED ONLY IF BIT 15 OF THE SWITCH REGISTER (SWREG) IS SET WHEN AN ERROR OCCURS.

K 1

4.0 PROGRESS & PERFORMANCE REPORTS

THE FOLLOWING REPORTS ARE ENABLED BIT 12 SET IN THE SWITCH REGISTER (SWREG LOC 176)
& WILL APPEAR FOLLOWING LOADING & STARTING AS DESCRIBED
IN SECTION 2.1 (ASSUMING ALL DEVICES ARE TO BE TESTED & NO ERRORS):

MEM TESTS DONE	(ALL MEMORY FROM LOCATION 0 TO THE BEGINNING OF THE MONITOR/ABS LOADER HAS BEEN TESTED.
CLOCK RUNNING	(100 INTERRUPTS HAVE BEEN DETECTED BEFORE EXERCISING THE NEXT DEVICE. THE CLOCK CONTINUES RUNNING IN INTERRUPT MODE.)
PRINTER RUNNING	(5 LINES HAVE BEEN PRINTED WITH ERROR BIT CHECKING ONLY. THE PRINTER RUNS CONTINUOUSLY IN INTERRUPT MODE AT 9600 BAUD. ITS ACTUAL PERFORMANCE IS A VISUAL CHECK IF PRINTER CONNECTED. NO OTHER CHECKING PERFORMED IF EXTERNAL LOOPBACK USED.
SYNC COMM DONE	(CHARACTERS 0 THRU 377 HAVE BEEN TRANSMITTED & RECEIVED WITH ODD PARITY ENABLED. INTERRUPTS ARE THEN DISABLED & THE SYNC COMM IS NO LONGER EXERCISED. THIS IS SO THAT THE ASYNC COMM CAN BE EXERCISED FOR THE DURATION OF THE PASS.)
ASYNC COMM RUNNING	(CHARS 0 THRU 377 HAVE BEEN TRANSMITTED & RECEIVED BEFORE EXERCISING THE NEXT DEVICE. IT CONTINUES RUNNING IN INTERRUPT MODE AT 9600 BAUD WITH ODD PARITY ENABLED & REPEATS THE 0 THRU 377 CYCLE)
TERM #1 RUNNING	(CHARACTERS 0 THRU 377 HAVE BEEN TRANSMITTED & RECEIVED BEFORE EXERXCISING THE NEXT DEVICE. THE TERMINAL KEEPS RUNNING CONTINUOUSLY AT 2400 BAUD IN INTERRUPT MODE & REPEATS THE 0 THRU 377 CYCLE)
TERM #2 RUNNING	(SAME AS #1)
TERM #3 RUNNING	(SAME AS #1)
DX0 TRK 0 DONE	(DRIVE 0, TRACK 0 IS WRITTEN WITH A DATA PATTERN, READ & DATA COMPARE PERFORMED)
DX1 TRK 0 DONE	(DRIVE 1, TRACK 0 IS WRITTEN WITH A DATA PATTERN, READ & DATA COMPARE PERFORMED)
DX0 TRK 1 DONE	(ETC)
DX1 TRK 1 DONE	(ETC UNTIL...)
DX0 DATA PATT DONE	(ALL TRACKS HAVE BEEN WRITTEN WITH A DATA PATTERN IN INTERRUPT MODE.)
DX1 DATA PATT DONE	(SAME AS DX0) (FOR 1'ST PASS TRACKS 0-4, 40-44, 72-76(10) DONE ONLY.)
DX0 RANDOM SEEKS DONE	(500 RANDOM SEEKS WITH READ & DATA COMPARE HAVE BEEN PERFORMED ON DRIVE 0)
DX1 RANDOM SEEKS DONE	(SAME AS DX0....10 RANDOM SEEKS ONLY FOR 1'ST PASS QUICK VERIFY)
END OF PASS #1	TOTAL ERRORS: 0 TOTAL SOFT ERRORS: 0 TOTAL ERRORS THIS PASS: 0 SOFT ERRORS THIS PASS: 0 (...& ENTIRE PROCESS REPEATS)

5.0 DEVICE REGISTERS.

PARAMETER REGISTER:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
.	.	PORT SELECT			BAUD RATE			USR LT2	USR LT1	PAR TYP	PAR EN	CHAR LENGTH		MAIN		177420 (WR ONLY)

LINE CLOCK: VECTOR: 100

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
									CLK IE							177546

CONSOLE: ADDR: 177560-177566 VECTOR: 60/64
CLUSTER #1: 176500-176506 300/304
#2: 176510-176516 310/314
#3: 176520-176526 320/324

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TKS									RCVR DONE	RCVR IE						
TKB	KB DATA BUFFER															
TPS									XMIT RDY	XMIT IE						
TPB	PRINTER DATA BUFFER (WRITE ONLY)															

PRINTER: VECTOR: 200

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
PRS	ERR								PRIN RDY	PRIN IE							177514
PRB	PRINTER DATA BUFFER (WRITE ONLY)																177516

ASYNC MODEM: VECTOR: 330/334

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RCSR	DSI	RING	CTS	CAR DET		SEC REC	DSET RDY		RCVR DONE	RCVR IE	DSET IE		SEC XMIT	RTS	DTR		176610
RBUF	ERR	OR ERR	FR ERR	PAR ERR					RECEIVER DATA BUFFER								176612
XCSR									XMIT RDY	XMIT IE						BRK	176614
XBUF	TRANSMITTER DATA BUFFER (WRITE ONLY)																176616

SYNC MODEM: VECTOR: 340/344

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RCSR	DSC	RING	CTS	CAR DET	REC ACT	SEC REC	DSET RDY	STRP SYNC	RCVR DONE	RCVR IE	DSET IE	SRCH SYNC	SEC XMIT	RTS	DTR		176620
RBUF	ERR	OR ERR		PAR ERR					RECEIVER DATA BUFFER								176622 (RD ONLY)
PCSR		SYNC CHAR			WORD LENGTH		PAR IE	EVEN PAR	SYNC REG								176622 (WR ONLY)
XCSR	DNA			MA MODE	MA CLK			MAST RST	XMIT RDY	XMIT IE	DNA IE	SEND	H/F DUP				176624
XBUF	TRANSMITTER DATA BUFFER (WRITE ONLY)																176626

DISK: VECTOR: 264

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RXCS ERR								CTL DONE	DRV IE		UNIT SEL	FUNCTION			GO	177170
RXDB	DATA BUFFER															177172
RXES								DRV RDY		DEL DATA	RNF	CRC	LOST DATA	INV ADDR	ID	177172
RX A	TRACK 0 - 114(8)											SECTOR 1 - 32(8)				177174

6.0 DEVICE CONNECTOR LAYOUT

CLUSTER TERMINAL #3 PORT	COMM PORT
CLUSTER TERMINAL #2 PORT	PRINTER PORT
CLUSTER TERMINAL #1 PORT	CONSOLE PORT

B 2

7.0 SUMMARY OF TESTS (SEE PROGRAM LISTING FOR ADDT'L DETAILS ON SPECIFIC TESTS)

PHASE 1

MEMORY TESTS
A. LOC 0 THRU END OF PROGRAM IS TESTED FOR TIMEOUT.
B. END OF PROGRAM TO BOTTOM OF MONITOR/ABS LOADER IS TESTED WITH A 1010 & 0101 PATTERN.
C. IF 28K PRESENT, MEMORY FROM 28K TO 30K IS SIMILARLY TESTED.

PHASE 2

ALL AVAILABLE PERIPHERALS ARE EXERCISED CONTINUOUSLY IN INTERRUPT MODE.

SEE SECTION 2.4 FOR DETAILS OF SETTING UP THE DEVICE MAP ($DEVM) FOR DEVICE TO BE TESTED.
SEE SECTION 4.1 FOR DETAILS ON PROGRESS REPORTS.

START LINE CLOCK	INTERRUPTS ARE DETECTED.
START PRINTER	CONTINUOUS LINES ARE PRINTED.
START SYNC COMM PORT	CHARS 27 THRU 377 " " " 1 PASS ONLY
START ASYNC COMM PORT	CHARS 0 THRU 377 TESTED CONTINUOUSLY.
START CLUSTER TERMINAL #1	CHARS 0 THRU 377 ARE XMITTED, REC'D & TESTED.
START CLUSTER TERMINAL #2	SAME
START CLUSTER TERMINAL #3	SAME

PHASE 3

WHILE THE ABOVE DEVICES ARE INTERRUPTING RANDOMLY, DISK SUBSYSTEM TESTS BEGIN:

FILL & EMPTY BUFFER TESTS ARE PERFORMED TO VERIFY THAT NO CABLE CROSS-TALK PROBLEMS EXIST BEFORE BEGINNING ACTUAL DATA TRANSFERS TO THE DISK SURFACE.

DX0 & DX1 DATA PATTERN: WRITE, READ & DATA COMPARE 1'ST 10 TRACKS ON 1'ST PASS.
ALL TRACKS ON SUBSEQUENT PASSES.

DX0 & DX1 RANDOM SEEKS: READ & DATA COMPARE.
20 SEEKS ON 1'ST 10 TRACKS ON 1'ST PASS.
500 SEEKS ON ALL TRACKS ON SUBSEQUENT PASSES.

DX0 INITIALIZE, RESTORE, WRITE DELETED DATA, READ STATUS, & INVALID ADDRESS TESTS.

NOTE:

1. RECOVERY IS PERFORMED (RESTORE COMMAND) AFTER ANY RNF ERROR ON WRITE OR READ.
2. DATA IS TESTED AFTER A HARD CRC ERROR ON A READ TO TEST WHETHER DATA CRC ERROR OR CRC ERROR.

%

```
                                .TITLE  PDT-11 EXERCISER
                                ;*COPYRIGHT (C) 1978
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY GARY PAPAZIAN
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*

                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;
                                ;       SWITCH          USE
                                ;       ------          ---
                                ;         15            HALT ON ERROR
                                ;         13            INHIBIT ERROR TYPEOUTS
                                ;         12            ENABLE PERFORMANCE REPORTS
                                ;         10            BELL ON ERROR
                                ;



                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3=     %3              ;;GENERAL REGISTER
000004                          R4=     %4              ;;GENERAL REGISTER
000005                          R5=     %5              ;;GENERAL REGISTER
000006                          R6=     %6              ;;GENERAL REGISTER
000007                          R7=     %7              ;;GENERAL REGISTER
000006                          SP=     %6              ;;STACK POINTER
000007                          PC=     %7              ;;PROGRAM COUNTER

                                ;*PRIORITY LEVEL DEFINITIONS
```

```
                    
000000              PR0=    0           ;;PRIORITY LEVEL 0
000040              PR1=    40          ;;PRIORITY LEVEL 1
000100              PR2=    100         ;;PRIORITY LEVEL 2
000140              PR3=    140         ;;PRIORITY LEVEL 3
000200              PR4=    200         ;;PRIORITY LEVEL 4
000240              PR5=    240         ;;PRIORITY LEVEL 5
000300              PR6=    300         ;;PRIORITY LEVEL 6
000340              PR7=    340         ;;PRIORITY LEVEL 7

                    ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
100000              SW15=   100000
040000              SW14=   40000
020000              SW13=   20000
010000              SW12=   10000
004000              SW11=   4000
002000              SW10=   2000
001000              SW09=   1000
000400              SW08=   400
000200              SW07=   200
000100              SW06=   100
000040              SW05=   40
000020              SW04=   20
000010              SW03=   10
000004              SW02=   4
000002              SW01=   2
000001              SW00=   1
                    .EQUIV  SW09,SW9
                    .EQUIV  SW08,SW8
                    .EQUIV  SW07,SW7
                    .EQUIV  SW06,SW6
                    .EQUIV  SW05,SW5
                    .EQUIV  SW04,SW4
                    .EQUIV  SW03,SW3
                    .EQUIV  SW02,SW2
                    .EQUIV  SW01,SW1
                    .EQUIV  SW00,SW0

                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000              BIT15=  100000
040000              BIT14=  40000
020000              BIT13=  20000
010000              BIT12=  10000
004000              BIT11=  4000
002000              BIT10=  2000
001000              BIT09=  1000
000400              BIT08=  400
000200              BIT07=  200
000100              BIT06=  100
000040              BIT05=  40
000020              BIT04=  20
000010              BIT03=  10
000004              BIT02=  4
000002              BIT01=  2
000001              BIT00=  1
                    .EQUIV  BIT09,BIT9
                    .EQUIV  BIT08,BIT8
```

```
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;''T'' BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;''TRAP'' TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR


                        ;;*********************************************************************
                        ;*      PDT-11/150 DEVICE VECTORS
                        ;;*********************************************************************

000200                  PRVEC=  200             ;PRINTER

000100                  CLKVEC= 100             ;LINE CLOCK

000330                  AMRVEC= 330             ;ASYNC MODEM RECVR
000334                  AMXVEC= 334             ;             XMITR

000340                  SMRVEC= 340             ;SYNC MODEM RECVR
000344                  SMXVEC= 344             ;             XMITR

000264                  RXVEC=  264             ;DISK

000300                  TK1VEC= 300             ;CLUSTER TERM #1 VECTORS
000304                  TP1VEC= 304
000310                  TK2VEC= 310             ;               #2
000314                  TP2VEC= 314
000320                  TK3VEC= 320             ;               #3
000324                  TP3VEC  324
```

```
                        ;;*****************************************************************
                        ;       PDT-11/150 DEVICE REGISTERS
                        ;;*****************************************************************

177420                  PARAM=  177420          ;PARAMETER REGISTER             (WRITE ONLY)

177546                  CLK=    177546          ;LINE CLOCK

177514                  PRS=    177514          ;PRINTER STATUS REG
177516                  PRB=    177516          ;       BUFFER                  (WRITE ONLY)

176610                  AMRC=   176610          ;ASYNC MODEM RECVR STATUS REG
176612                  AMRB=   176612          ;                   BUFFER
176614                  AMXC=   176614          ;           XMITR STATUS REG
176616                  AMXB=   176616          ;                   BUFFER      (WRITE ONLY)

176620                  SMRC=   176620          ;SYNC MODEM RECVR STATUS REG
176622                  SMRB=   176622          ;                   BUFFER      (READ ONLY)
176622                  SMPAR=  176622          ;             PARAMETER REG     (WRITE ONLY)
176624                  SMXC=   176624          ;           XMITR STATUS REG
176626                  SMXB=   176626          ;                   BUFFER      (WRITE ONLY)

177170                  RXCS=   177170          ;DISK CSR
177172                  RXDB=   177172          ;       DATA BUFF
177172                  RXES=   177172          ;       ERROR & STATUS REG
177174                  RXSA=   177174          ;       TRK & SECTOR ADDR REG

176500                  TK1S=   176500          ;CLUSTER TERMINAL #1
176502                  TK1B=   176502
176504                  TP1S=   176504
176506                  TP1B=   176506

176510                  TK2S=   176510          ;               #2
176512                  TK2B=   176512
176514                  TP2S=   176514
176516                  TP2B=   176516

176520                  TK3S=   176520          ;               #3
176522                  TK3B=   176522
176524                  TP3S=   176524
176526                  TP3B=   176526
```

```
                        ;;*****************************************************************
                        ;*      PARAMETER REGISTER EQUATES      WRITE ONLY
                        ;;*****************************************************************

                        ;BIT 14-12  PORT SELECT
                        ;
010000                  CONSOL= 10000           ;SELECT CONSOLE TERMINAL
000000                  CT1=    0               ;       CLUSTER TERMINAL #1
050000                  CT2=    50000           ;                       #2
060000                  CT3=    60000           ;                       #3
020000                  PRT=    20000           ;       PRINTER PORT
030000                  AMOD=   30000           ;       ASYNC COMM PORT
040000                  ULITE=  40000           ;       USER LIGHTS

                        ;BIT 11-8  BAUD RATE    (DELTA = 400)
                        ;
000000                  B50=    0               ;50 BAUD
000400                  B75=    400             ;75
001000                  B110=   1000            ;110
001400                  B134=   1400            ;134.5
002000                  B150=   2000            ;150
002400                  B300=   2400            ;300
003000                  B600=   3000            ;600
003400                  B1200=  3400            ;1200
004000                  B1800=  4000            ;1800
004400                  B2000=  4400            ;2000
005000                  B2400=  5000            ;2400 (CLUSTER DEFAULT)
005400                  B3600=  5400            ;3600
006000                  B4800=  6000            ;4800
006400                  B7200=  6400            ;7200
007000                  B9600=  7000            ;9600 (PRINTER & ASYNC DEFAULT)
007400                  B19200= 7400            ;19200
                        ;
                        ;IF 110 BAUD SELECTED, 2 STOP BITS ARE ASSUMED.

                        ;BITS 4-5 PARITY CONTROL
                        ;
000060                  EPAR=   60              ;EVEN PARITY ENABLE
000020                  OPAR=   20              ;ODD PARITY ENABLE (DEFAULT)

                        ;BITS 3-2  CHARACTER LENGTH (DELTA - 4)
                        ;
000000                  CHAR5=  0               ;5 BITS/CHAR
000004                  CHAR6=  4               ;6
000010                  CHAR7=  10              ;7
000014                  CHAR8=  14              ;8 (PRINTER,TERMINAL,MODEM DEFAULT)

                        ;BIT 1  MAINTENANCE BIT
                        ;
000002                  MAINT=  BIT1            ;ENABLE MAINT MODE
                                                ;PRINTER & DISK DO NOT HAVE MAINT MODE


                        ;;*****************************************************************
```

```
                        ;*      ASYNC MODEM BIT DEFINITIONS
                        ;;*****************************************************************
                        ;RCSR
                        ;
100000                  DSI=    BIT15           ;DATA SET INTERRUPT
040000                  RING=   BIT14           ;RING
020000                  CTS=    BIT13           ;CLEAR TO SEND
010000                  CDET=   BIT12           ;CARRIER DETECTED
002000                  SREC=   BIT10           ;SECONDARY RECD/SUPERVISORY RECD
001000                  DSRDY=  BIT9            ;DATA SET RDY
000200                  DONE=   BIT7            ;RECVR DONE
000100                  IE=     BIT6            ;RECVR IE
000040                  DSIE=   BIT5            ;DATA SET IE
000010                  SXMIT=  BIT3            ;SECONDARY XMIT/SUPERV XMIT
000004                  RTS=    BIT2            ;REQ TO SEND
000002                  DTR=    BIT1            ;DATA TERMINAL RDY

                        ;RBUF
                        ;
100000                  ERR=    BIT15           ;ERROR
040000                  ORERR=  BIT14           ;OVERRUN ERROR
020000                  FRERR=  BIT13           ;FRAMING ERROR
010000                  PARERR= BIT12           ;PARITY ERROR

                        ;XCSR
                        ;
000200                  RDY-    BIT7            ;XMIT RDY
000100                  IE-     BIT6            ;XMIT IE
000001                  BREAK-  BIT0            ;BREAK

```

```
                        ;;*****************************************************************
                        ;*      SYNC MODEM BIT DEFINITIONS
                        ;;*****************************************************************

                        ;RCSR
                        ;
100000                  DSC=    BIT15           ;DATA SET CHANGE
040000                  RING=   BIT14           ;RING
020000                  CTS=    BIT13           ;CLEAR TO SEND
010000                  CDET=   BIT12           ;CARRIER DETECTED
004000                  RACT=   BIT11           ;RECVR ACTIVE  (SYNC DETECT)
002000                  SREC=   BIT10           ;SECONDARY RECD/SUPERVISORY RECD
001000                  DSRDY=  BIT9            ;DATA SET RDY
000400                  STSYN=  BIT8            ;STRIP SYNC
000200                  DONE=   BIT7            ;RECVR DONE
000100                  IE=     BIT6            ;RECVR IE
000040                  DSIE=   BIT5            ;DATA SET IE
000020                  SRSYN=  BIT4            ;SEARCH SYNC
000010                  SXMIT=  BIT3            ;SECONDARY XMIT/SUPERV XMIT
000004                  RTS=    BIT2            ;REQ TO SEND
000002                  DTR=    BIT1            ;DATA TERMINAL RDY

                        ;RBUF
                        ;
100000                  ERR=    BIT15           ;ERROR
040000                  ORERR=  BIT14           ;OVERRUN ERROR
010000                  PARERR= BIT12           ;PARITY ERROR

                        ;PARCSR  PARAMETER CONT REG
                        ;
                        ;BIT 14   SYNC CHAR
                        ;
000000                  SYNC2=  0               ;2 SYNC CHARS (DEFAULT)
040000                  SYNC1=  BIT14           ;1

                        ;BITS 10-11     WORD LENGTH
                        ;
000000                  CHR5=   0               ;5 BITS/CHAR
002000                  CHR6=   2000            ;6
004000                  CHR7=   4000            ;7
006000                  CHR8=   6000            ;8 (DEFAULT)

                        ;BITS 8-9 PARITY CONTROL
                        ;
001400                  ENVPAR= 1400            ;ENABLE EVEN PARITY
001000                  ODDPAR= 1000            ;ENABLE ODD PARITY (DEFAULT)

                        ;XCSR
                        ;
100000                  DNA     BIT15           ;DATA NOT AVAIL
010000                  MM=     BIT12           ;MAINT MODE
004000                  MCLK=   BIT11           ;MAINT CLOCK
000400                  MR=     BIT8            ;MASTER RESET
000200                  RDY     BIT7            ;XMIT RDY
000100                  IE      BIT6            ;XMIT IE
```

```
1014            000040                  DNAIE=  BIT5            ;DATA NOT AVAIL IE
1015            000020                  SEND=   BIT4            ;SEND
1016            000010                  HFDUP=  BIT3            ;1=HALF DUPLEX, 0=FULL DUPLEX (DEFAULT = 0 FOR TESTS)
1017
1018
1019
1020
1021                                    ;;*****************************************************************
1022                                    ;DISK BIT DEFINITIONS
1023                                    ;;*****************************************************************
1024
1025                                    ;RXCS
1026                                    ;
1027            100000                  ERR=    BIT15           ;ERROR
1028            000200                  DONE=   BIT7            ;CONTR RDY
1029            000100                  IE=     BIT6            ;DRIVE IE
1030            000020                  DX1=    BIT4            ;UNIT SELECT: 0=DX0, 1=DX1
1031            000020                  USEL=   BIT4            ;UNIT SELECT
1032
1033                                    ;FUNCTIONS (INCLUDES BIT0 = GO)
1034
1035            000001                  FBUF=   1               ;FILL BUFFER
1036            000003                  EBUF=   3               ;EMPTY BUFFER
1037            000005                  WSEC=   5               ;WRITE SECTOR
1038            000007                  RSEC=   7               ;READ SECTOR
1039            000011                  INITAL= 11              ;INITIALIZE
1040            000013                  RSTAT=  13              ;READ STATUS
1041            000015                  WDDSEC= 15              ;WRITE DELETED DATA SECTOR
1042            000017                  RESTOR= 17              ;RESTORE TO TRACK 0
1043
1044                                    ;RXES: ERROR & STATUS
1045                                    ;
1046            000200                  RDY=    BIT7            ;DRIVE RDY
1047            000040                  DD=     BIT5            ;DELETED DATA DET.
1048            000020                  RNF=    BIT4            ;RECORD NOT FOUND
1049            000010                  CRC=    BIT3            ;CRC ERROR
1050            000004                  LDATA=  BIT2            ;LOST DATA
1051            000002                  INVADR= BIT1            ;INVALID ADDR
1052            000001                  ID=     BIT0            ;INITIALIZE DONE
1053
1054
```

```
                                ;;*****************************************************************
                                ;*      SWITCH REGISTER
                                ;;*****************************************************************

            000174                      .=174
000174  000000                  DISPREG:        .WORD   0       ;SOFTWARE DISPLAY REG
000176  000000                  SWREG:  .WORD 0                 ;SOFTWARE SWITCH REG

            000100                      .=100
000100  000102                          102                     ;SETUP CLOCK VECTOR AREA TO DO RTI
000102  000002                          2                       ;IF ENABLED

            000200                      .=200
000200  000137  003174                  JMP     START           ;USE ONLY ONCE. WILL BE OVERLAID BY
                                                                ;PRINTER VECTOR ADDR.
            000240                      .=240
000240  000137  003174                  JMP     START           ;RESTART ADDR

            000250                      .=250
000250  000137  003156                  JMP     START3          ;ENTER HERE FOR COPY UTILITY
            000260                      .=260
000260  000137  003136                  JMP     START1          ;COMPATABILTIY PASS 1
            000270                      .=270
000270  000137  003144                  JMP     START2          ;               PASS 2


            001000              .=1000
                                .SBTTL  APT PARAMETER BLOCK

                                ;;*****************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;*****************************************************************
            001000                      .$X=.   ;;SAVE CURRENT LOCATION
            000024                      .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                          200     ;;FOR APT START UP
            000044                      .-44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
            001000                      .=.$X   ;;RESET LOCATION COUNTER
                                ;;*****************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

001000                          $APTHD:
001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001170                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000024                  $TSTM:  .WORD   20.     ;;RUN TIM OF LONGEST TEST
001006  000226                  $PASTM: .WORD   150.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000000                  $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000030                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

                                ;;*****************************************************************
                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;*USED IN THE PROGRAM.

                                        .=1100
001100                          $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                          .WORD   0
001102  000                     $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000                     $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000                     $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001                     $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134  000                     $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000                     $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                          .WORD   0
001140  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560                  $TKS:   177560                  ;;TTY KBD STATUS
001146  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000                     $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002                     $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012                     $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
001157  000                     $TPFLG: .BYTE   0               ;;'TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
001160  177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001164  077                     $QUES:  .ASCII  /?/             ;;QUESTION MARK
001165  015                     $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001166  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;*****************************************************************
                                .EVEN
001170                          $MAIL:                          ;;APT MAILBOX
001170  000000                  $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001172  000000                  $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001174  000000                  $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001176  000000                  $PASS:  .WORD   APASS           ;;PASS COUNT
001200  000000                  $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
001202  000000                  $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001204  000000                  $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
001206  000000                  $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
001210                          $ETABLE:                        ;;APT ENVIRONMENT TABLE
```

```
1162  001210  000          $ENV:    .BYTE   AENV    ;;ENVIRONMENT BYTE
1163  001211  000          $ENVM:   .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
1164  001212  000000       $SWREG:  .WORD   ASWREG  ;;APT SWITCH REGISTER
1165  001214  000000       $USWR:   .WORD   AUSWR   ;;USER SWITCHES
1166  001216  000000       $CPUOP:  .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
1167                        ;*                       BITS 15-11=CPU TYPE
1168                        ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45-05
1169                        ;*                              11/70=06,PDQ=07,Q=10
1170                        ;*                       BIT 10=REAL TIME CLOCK
1171                        ;*                       BIT  9=FLOATING POINT PROCESSOR
1172                        ;*                       BIT  8=MEMORY MANAGEMENT
1173  001220  000          $MAMS1:  .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
1174  001221  000          $MTYP1:  .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
1175                        ;*                       MEM.TYPE BYTE   --  (HIGH BYTE)
1176                        ;*                              900 NSEC CORE=001
1177                        ;*                              300 NSEC BIPOLAR=002
1178                        ;*                              500 NSEC MOS=003
1179  001222  000000       $MADR1:  .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
1180                        ;*                         MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
1181  001224  000          $MAMS2:  .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
1182  001225  000          $MTYP2:  .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
1183  001226  000000       $MADR2:  .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
1184  001230  000          $MAMS3:  .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
1185  001231  000          $MTYP3:  .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
1186  001232  000000       $MADR3:  .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
1187  001234  000          $MAMS4:  .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
1188  001235  000          $MTYP4:  .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
1189  001236  000000       $MADR4:  .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
1190  001240  000300       $VECT1:  .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1191  001242  000000       $VECT2:  .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1192  001244  176500       $BASE:   .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1193  001246  000017       $DEVM:   .WORD   ADEVM   ;;DEVICE MAP
1194  001250               $ETEND:
1195                        .MEXIT
```

```
1196                                    .SBTTL  ERROR POINTER TABLE
1197
1198                                    ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
1199                                    ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1200                                    ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
1201                                    ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
1202                                    ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1203
1204                                    ;*      EM              ;;POINTS TO THE ERROR MESSAGE
1205                                    ;*      DH              ;;POINTS TO THE DATA HEADER
1206                                    ;*      DT              ;;POINTS TO THE DATA
1207                                    ;*      DF              ;;POINTS TO THE DATA FORMAT
1208
1209
1210  001250                            $ERRTB:
1211                                    ;;              GLOBAL DATA
1212                                    ;ERR 1
1213  001250  020456  023074  023502            EM1,DH2,DT2,DF
1214  001256  023776
1215                                    ;ERR 2
1216  001260  020456  023074  023502            EM1,DH2,DT2,DF
1217  001266  023776
1218                                    ;ERR 3
1219  001270  020475  023120  023512            EM2,DH3,DT3,DF
1220  001276  023776
1221                                    ;ERR 4
1222  001300  020456  023074  023502            EM1,DH2,DT2,DF
1223  001306  023776
1224                                    ;ERR 5
1225  001310  020475  023120  023512            EM2.DH3,DT3,DF
1226  001316  023776
1227                                    ;ERR 6
1228  001320  020517  023055  023474            EM3,DH1,DT1,DF
1229  001326  023776
1230                                    ;ERR 7
1231  001330  020563  023162  023526            EM4,DH4,DT4,DF
1232  001336  023776
1233                                    ;ERR 10
1234  001340  020606  023213  023536            EM5,DH5,DT5,DF
1235  001346  023776
1236                                    ;ERR 11
1237  001350  020634  023213  023550            EM6,DH5,DT6,DF
1238  001356  023776
1239                                    ;ERR 12
1240  001360  020662  023213  023562            EM7,DH5,DT7,DF
1241  001366  023776
1242                                    ;ERR 13
1243  001370  020710  023213  023574            EM8,DH5,DT8,DF
1244  001376  023776
1245                                    ;ERR 14
1246  001400  020741  023213  023574            EM9,DH5,DT8,DF
1247  001406  023776
1248                                    ;ERR 15
1249  001410  020771  023250  023606            EM10,DH6,DT9,DF
1250  001416  023776
1251                                    ;ERR 16
```

```
1252  001420  021016  023315  023624          EM11,DH7,DT10,DF1
1253  001426  024004
1254                                  ;ERR 17
1255  001430  021016  023315  023624          EM11,DH7,DT10,DF1
1256  001436  024004
1257                                  ;ERR 20
1258  001440  021034  023315  023624          EM13,DH7,DT10,DF1
1259  001446  024004
1260                                  ;ERR 21
1261  001450  021065  023315  023624          EM14,DH7,DT10,DF1
1262  001456  024004
1263                                  ;ERR 22
1264  001460  021116  023404  023646          EM15,DH8,DT11,DF2
1265  001466  024014
1266                                  ;ERR 23
1267  001470  021147  023404  023646          EM16,DH8,DT11,DF2
1268  001476  024014
1269                                  ;ERR 24
1270  001500  021200  023315  023670          EM17,DH7,DT12,DF1
1271  001506  024004
1272                                  ;ERR 25
1273  001510  021200  023315  023670          EM17,DH7,DT12,DF1
1274  001516  024004
1275                                  ;ERR 26
1276  001520  021216  023315  023670          EM19,DH7,DT12,DF1
1277  001526  024004
1278                                  ;ERR 27
1279  001530  021247  023315  023670          EM20,DH7,DT12,DF1
1280  001536  024004
1281                                  ;ERR 30
1282  001540  021300  023404  023712          EM21,DH8,DT13,DF2
1283  001546  024014
1284                                  ;ERR 31
1285  001550  021331  023404  023712          EM22,DH8,DT13,DF2
1286  001556  024014
1287                                  ;ERR 32
1288  001560  021362  023055  023474          EM23,DH1,DT1,DF
1289  001566  023776
1290                                  ;ERR 33
1291  001570  021373  023162  023526          EM24,DH4,DT4,DF
1292  001576  023776
1293                                  ;ERR 34
1294  001600  021410  023055  023474          EM25,DH1,DT1,DF
1295  001606  023776
1296                                  ;ERR 35
1297  001610  021425  023055  023474          EM26,DH1,DT1,DF
1298  001616  023776
1299                                  ;ERR 36
1300  001620  021442  023055  023474          EM27,DH1,DT1,DF
1301  001626  023776
```

```
1302                                        ;ERR 37
1303  001630  021457  023055  023474              EM28,DH1,DT1,DF
1304  001636  023776
1305                                        ;ERR 40
1306  001640  021476  023055  023474              EM29,DH1,DT1,DF
1307  001646  023776
1308                                        ;ERR 41
1309  001650  021516  023250  023734              EM30,DH6,DT14,DF
1310  001656  023776
1311                                        ;ERR 42
1312  001660  021527  023250  023734              EM31,DH6,DT14,DF
1313  001666  023776
1314                                        ;ERR 43
1315  001670  021540  023055  023474              EM32,DH1,DT1,DF
1316  001676  023776
1317                                        ;ERR 44
1318  001700  021603  023055  023474              EM33,DH1,DT1,DF
1319  001706  023776
1320                                        ;ERR 45
1321  001710  021644  023055  023474              EM34,DH1,DT1,DF
1322  001716  023776
1323                                        ;ERR 46
1324  001720  021516  023250  023734              EM30,DH6,DT14,DF
1325  001726  023776
1326                                        ;ERR 47
1327  001730  021516  023250  023734              EM30,DH6,DT14,DF
1328  001736  023776
1329                                        ;ERR 50
1330  001740  021516  023250  023734              EM30,DH6,DT14,DF
1331  001746  023776
1332                                        ;ERR 51
1333  001750  021516  023250  023734              EM30,DH6,DT14,DF
1334  001756  023776
1335                                        ;ERR 52
1336  001760  021703  023250  023606              EM35,DH6,DT9,DF
1337  001766  023776
1338                                        ;ERR 53
1339  001770  021750  023404  023646              EM36,DH8,DT11,DF2
1340  001776  024014
1341                                        ;ERR 54
1342  002000  022017  023250  023606              EM37,DH6,DT9,DF
1343  002006  023776
1344                                        ;ERR 55
1345  002010  022043  023250  023606              EM38,DH6,DT9,DF
1346  002016  023776
1347                                        ;ERR 56
1348  002020  021516  023250  023734              EM30,DH6,DT14,DF
1349  002026  023776
1350                                        ;ERR 57
1351  002030  022111  023250  023606              EM39,DH6,DT9,DF
1352  002036  023776
1353                                        ;ERR 60
1354  002040  021516  023250  023734              EM30,DH6,DT14,DF
1355  002046  023776
1356                                        ,ERR 61
1357  002050  022141  023250  023606              EM40,DH6,DT9,DF
```

```
1358  002056  023776
1359                                  ;ERR 62
1360  002060  022204  023250  023606          EM41,DH6,DT9,DF
1361  002066  023776
1362                                  ;ERR 63
1363  002070  022240  023250  023606          EM42,DH6,DT9,DF
1364  002076  023776
1365                                  ;ERR 64
1366  002100  022261  023315  023624          EM43,DH7,DT10,DF1
1367  002106  024004
1368                                  ;ERR 65
1369  002110  022261  023315  023624          EM43,DH7,DT10,DF1
1370  002116  024004
1371                                  ;ERR 66
1372  002120  022313  023315  023624          EM45,DH7,DT10,DF1
1373  002126  024004
1374                                  ;ERR 67
1375  002130  022313  023315  023624          EM45,DH7,DT10,DF1
1376  002136  024004
1377                                  ;ERR 70
1378  002140  022344  023315  023670          EM47,DH7,DT12,DF1
1379  002146  024004
1380                                  ;ERR 71
1381  002150  022344  023315  023670          EM47,DH7,DT12,DF1
1382  002156  024004
1383                                  ;ERR 72
1384  002160  022376  023315  023670          EM49,DH7,DT12,DF1
1385  002166  024004
1386                                  ;ERR 73
1387  002170  022376  023315  023670          EM49,DH7,DT12,DF1
1388  002176  024004
1389                                  ;ERR 74
1390  002200  022427  023250  023606          EM51,DH6,DT9,DF
1391  002206  023776
1392                                  ;ERR 75
1393  002210  022453  023250  023734          EM52,DH6,DT14,DF
1394  002216  023776
1395                                  ;ERR 76
1396  002220  022477  023213  023752          EM53,DH5,DT15,DF
1397  002226  023776
1398                                  ;ERR 77
1399  002230  022477  023213  023764          EM53,DH5,DT16,DF
1400  002236  023776
```

```
1401
1402                                    ; ERRORS FOR COPY UTILITY
1403
1404                                    ;ERR 100
1405  002240  021034  023250  023734            EM13,DH6,DT14,DF
1406  002246  023776
1407                                    ;ERR 101
1408  002250  021200  023250  023734            EM17,DH6,DT14,DF
1409  002256  023776
1410                                    ;ERR 102
1411  002260  022546  023250  023734            EM54,DH6,DT14,DF
1412  002266  023776
1413                                    ;ERR 103
1414  002270  021216  023250  023734            EM19,DH6,DT14,DF
1415  002276  023776
1416
1417                                    ;BAD SECTOR/TRACK TABLE FULL ERRORS
1418
1419                                    ;ERR 104
1420  002300  022563  000000  000000            EM55,0,0,0
1421  002306  000000
1422                                    ;ERR 105
1423  002310  022620  000000  000000            EM56,0,0,0
1424  002316  000000
1425
1426                                    ;CRC ERRORS
1427
1428                                    ;ERR 106
1429  002320  022655  023315  023624            EM57,DH7,DT10,DF1
1430  002326  024004
1431                                    ;ERR 107
1432  002330  022676  023404  023646            EM58,DH8,DT11,DF2
1433  002336  024014
1434                                    ;ERR 110
1435  002340  022717  000000  000000            EM59,0,0,0
1436  002346  000000
1437                                    ;ERR 111
1438  002350  022755  023315  023670            EM60,DH7,DT12,DF1
1439  002356  024004
1440                                    ;ERR 112
1441  002360  022776  023404  023712            EM61,DH8,DT13,DF2
1442  002366  024014
1443                                    ;ERR 113
1444  002370  023017  000000  000000            EM62,0,0,0
1445  002376  000000
```

```
1446                                    ;;*****************************************************************
1447                                    ;REGISTERS & BUFFERS FOR DX0 TESTS
1448                                    ;;*****************************************************************
1449
1450  002400  000000                    D0PAT:  0               ;DX0 PATT TEST DONE FLAG
1451  002402  000000                    D0CNT:  0               ;DX0 RANDOM SEEK COUNTER
1452  002404  000000                    D0ERR:  0               ;DX0 RCSR ERR COUNTER
1453  002406  000000                    D0TRK:  0               ;DX0 TRACK
1454  002410  000000                    D0SEC:  0               ;DX0 SECTOR
1455  002412  000000                    D0CMER: 0               ;DX0 DATA COMPARE ERROR FLAG
1456  002414  000000                    D0CERR: 0               ;DX0 TOTAL DATA COMP ERROR CT
1457  002416  000000                    D0EXP:  0               ;DX0 EXPECTED DATA
1458  002420  000000                    D0REC:  0               ;DX0 RECEIVED DATA
1459  002422  000000                    D0CRC:  0               ;DX0 CRC ERROR FLAG
1460  002424  000100                    D0BUF:  .BLKW 100       ;DX0 DATA BUFFER
1461  002624  000012                    D0BAD:  .BLKW 10.       ;DX0 BAD TRK/SEC TABLE
1462
1463
1464                                    ;;*****************************************************************
1465                                    ;REGISTERS & BUFFERS FOR DX1 TESTS
1466                                    ;;*****************************************************************
1467
1468  002650  000000                    D1PAT:  0               ;DX1 PATT TEST DONE FLAG
1469  002652  000000                    D1CNT:  0               ;DX1 RANDOM SEEK COUNTER
1470  002654  000000                    D1ERR:  0               ;DX1 RCSR ERR COUNTER
1471  002656  000000                    D1TRK:  0               ;DX1 TRACK
1472  002660  000000                    D1SEC:  0               ;DX1 SECTOR
1473  002662  000000                    D1CMER: 0               ;DX1 DATA COMPARE ERROR FLAG
1474  002664  000000                    D1CERR: 0               ;DX1 TOTAL DATA COMPARE ERROR CT
1475  002666  000000                    D1EXP:  0               ;DX1 EXPECTED DATA
1476  002670  000000                    D1REC:  0               ;DX1 RECEIVED DATA
1477  002672  000000                    D1CRC:  0               ;DX1 CRC ERROR FLAG
1478  002674  000100                    D1BUF:  .BLKW 100       ;DX1 DATA BUFFER
1479  003074  000012                    D1BAD:  .BLKW 10.       ;DX1 BAD TRK/SEC TABLE
1480
1481  003120  000000                    DXTST:  0               ;1 = DATA PATT TEST
1482                                                            ;2 = RANDOM SEEK TEST
1483                                                            ;3 = INITIALIZE TEST
1484                                                            ;4 = RESTORE TEST
1485                                                            ;5 - WRITE DELETED DATA TEST
1486                                                            ;6 - INVALID ADDRESS TEST
1487
1488  003122  000000                    FIN:    0               ;COMMON FLG FOR DX0 & DX1 FOR TRK/SEC FINISHED
1489  003124  000000                    MAXTRK: 0               ;MAX TRACK #
1490                                                            ;0 FOR 1'ST PASS, 114 FOR SUBSEQUENT PASSES. SET AT START.
1491  003126  000000                    MAXSK:  0               ;# OF RANDOM SEEKS TO BE PERFORMED.
1492                                                            ;10 FOR 1'ST PASS, 500 FOR SUBSEQUENT PASSES. SET AT EOP.
1493  003130  000000                    COPFLG: 0               ; 0 - NORM TEST   1 = COPY UTILITY ACTIVE
1494  003132  000000                    COMP1:  0               ;1   COMPAT TESTS, PASS 1
1495  003134  000000                    COMP2:  0               ;1                 PASS 2
1496
1497
```

```
1498                                        .SBTTL  PROGRAM
1499
1500  003136  005237  003132        START1: INC     COMP1           ;COMPAT PASS 1
1501  003142  000424                        BR      S1
1502
1503  003144  005237  003134        START2: INC     COMP2           ;COMPAT PASS 2
1504  003150  005037  003132                CLR     COMP1
1505  003154  000421                        BR      S2
1506
1507  003156  005237  003130        START3: INC     COPFLG          ;COPY UTILITY
1508  003162  005037  003132                CLR     COMP1
1509  003166  005037  003134                CLR     COMP2
1510  003172  000414                        BR      S3
1511
1512  003174  012737  000102  000100 START:  MOV     #CLKVEC+2,CLKVEC        ;DO RTI IF CLOCK INTERR
1513  003202  012737  000002  000102         MOV     #RTI,CLKVEC+2
1514
1515  003210  005037  003132                CLR     COMP1           ;NORMAL TESTING
1516  003214  005037  003134        S1:     CLR     COMP2
1517  003220  005037  003130        S2:     CLR     COPFLG
1518
1519  003224  012706  001100        S3:     MOV     #STACK,SP       ;SETUP STACK POINTER
1520  003230  042777  000100  175706        BIC     #100,@$TKS      ;DISABLE ANY TTY INTERRUPTS
1521                                                                ;TO RT-11 MONITOR
1522                                        ;MTPS   #PR4            ;DISABLE INTERRUPTS
1523  003236  106427                        .WORD   106427
1524  003240  000200                        PR4
1525  003242  013737  000042  004464        MOV     42,HLD42        ;SAVE
1526  003250  123727  001210  000001        CMPB    $ENV,#1         ;APT?
1527  003256  001402                        BEQ     2$              ;BR IF YES
1528  003260  005037  000042                CLR     42              ;ELSE CLR SO WE CAN USE SOFTSWR
1529  003264                        2$:
1530                                        .SBTTL  INITIALIZE THE COMMON TAGS
1531                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1532  003264  012706  001100                MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1533  003270  005026                        CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1534  003272  022706  001140                CMP     #SWR,R6 ;;DONE?
1535  003276  001374                        BNE     .-6             ;;LOOP BACK IF NO
1536  003300  012706  001100                MOV     #STACK,SP       ;;SETUP THE STACK POINTER
1537                                        ;;INITIALIZE A FEW VECTORS
1538  003304  012737  025416  000030        MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1539  003312  012737  000340  000032        MOV     #340,@#EMTVEC+2 ;;LEVEL 7
1540  003320  012737  026450  000034        MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1541  003326  012737  000340  000036        MOV     #340,@#TRAPVEC+2;LEVEL 7
1542                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1543                                        ;;EQUAL TO A ''-1'', SETUP FOR A SOFTWARE SWITCH REGISTER.
1544  003334  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1545  003340  012737  003374  000004        MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1546  003346  012737  177570  001140        MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1547  003354  012737  177570  001142        MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1548  003362  022777  177777  175550        CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1549  003370  001012                        BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1550                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
1551  003372  000403                        BR      65$             ;;BRANCH IF NO TIMEOUT
1552  003374  012716  003402        64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1553  003400  000002                        RTI
```

```
1554  003402  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1555  003410  012737  000174  001142          MOV     #DISPREG,DISPLAY
1556  003416  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1557
1558  003422  005037  001176                  CLR     $PASS           ;;CLEAR PASS COUNT
1559  003426  132737  000200  001211          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
1560  003434  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
1561  003436  012737  001212  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
1562  003444                          67$:
1563                                  .SBTTL  TYPE PROGRAM NAME
1564                                  ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1565  003444  005227  177777                  INC     #-1             ;;FIRST TIME?
1566  003450  001046                          BNE     68$             ;;BRANCH IF NO
1567  003452  104401  003520                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1568                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1569  003456  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1570  003462  001012                          BNE     70$             ;;BRANCH IF YES
1571  003464  123727  001210  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
1572  003472  001406                          BEQ     70$             ;;BRANCH IF YES
1573  003474  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1574  003502  001005                          BNE     71$             ;;BRANCH IF NO
1575  003504  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
1576  003506  000403                          BR      71$
1577  003510  112737  000001  001134  70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1578  003516                          71$:
1579  003516  000423                          BR      68$             ;;GET OVER THE ASCIZ
1580                                  ;;69$:  .ASCIZ  <CRLF>*CVKDAB  PDT-11/150 SYSTEM EXERCISER*<CRLF>
1581  003566                          68$:
1582  003566  005737  003130                  TST     COPFLG          ;COPY UTILITY?
1583  003572  001057                          BNE     LOOP            ;BR IF YES
1584
1585  003574  005737  003132                  TST     COMP1           ;COMPAT PASS 1?
1586  003600  001054                          BNE     LOOP            ;BR IF YES
1587  003602  005737  003134                  TST     COMP2           ;COMPAT PASS 2?
1588  003606  001051                          BNE     LOOP            ;BR IF YES
1589
1590  003610  012737  000004  003124          MOV     #4,MAXTRK       ;TRK 0-4, 40-44, 72-76(10) FOR 1'ST PASS
1591  003616  012737  000012  003126          MOV     #10.,MAXSK      ;10 RANDOM SEEKS FOR 1'ST PASS
1592
1593  003624  005037  025766                  CLR     TERR            ;TOTAL ERR CT
1594  003630  005037  025770                  CLR     TSERR           ;TOTAL SOFT ERR CT FOR EOP TYPEOUT
1595
1596  003634  052737  000006  176610          BIS     #<RTS!DTR>,AMRC
1597  003642  042737  000006  176610          BIC     #<RTS.DTR>,AMRC ;DTR MUST BE TOGGLED.
```

```
1598
1599  003650  005227  177777          INC     #-1             ;1'ST TIME?
1600  003654  001005                  BNE     4$              ;BR IF NO
1601  003656  123727  001210  000001  CMPB    $ENV,#1         ;APT?
1602  003664  001401                  BEQ     4$              ;BR IF YES
1603  003666  104412                  GTDEVM                  ;SHOW CURRENT DEVM & GET NEW
1604
1605  003670  004737  010342  4$:     JSR     PC,SIZE         ;SEE WHO IS ON THE SYSTEM
1606  003674  123727  001210  000001  CMPB    $ENV,#1         ;ARE WE RUNNING UNDER APT?
1607  003702  001413                  BEQ     LOOP            ;BR IF YES & DONT HALT
1608  003704  032737  001000  001246  BIT     #1000,$DEVM     ;DX0 THERE?
1609  003712  001404                  BEQ     1$              ;BR IF YES
1610  003714  032737  000400  001246  BIT     #400,$DEVM      ;DX1 THERE?
1611  003722  001003                  BNE     LOOP            ;BR IF NO
1612  003724  104401  017351  ^$:     TYPE    ,WARNING        ;INSERT SCRATCH DISKS
1613  003730  000000                  HALT
1614
1615
1616  003732  012706  001100  LOOP:   MOV     #1100,SP        ;LOOP HERE AFTER EOP & RESET STACK
1617
1618  003736  004737  011566          JSR     PC,TIMER1       ;TIMER TO ALLOW 'P' TO PRINT BEFORE RESET
1619
1620  003742  000005                  RESET                   ;CLEAR ALL INTERRUPT ENABLES
1621  003744  042737  000100  177546  BIC     #IE,CLK         ;RESET SETS CLK IE. DONT ALLOW YET
1622
1623                                  ;MTPS   #PR0            ;ALLOW INTERRUPTS
1624  003752  106427                  .WORD   106427
1625  003754  000000                  PR0
1626
1627  003756  052737  000006  176610  BIS     #<RTS!DTR>,AMRC ;TO CONDITION ALL TERMINALS &
1628                                                          ;COMM PORT TO OPERATE IN EXT LOOPBACK MODE.
1629                                                          ;NO HARM IF LOOPBACK NOT USED.
1630  003764  004737  011566          JSR     PC,TIMER1       ;TIMER TO ALLOW PREV XFERS TO FINISH
1631
1632  003770  005037  025772          CLR     PERR            ;PASS ERR COUNT FOR EOP TYPEOUT
1633  003774  005037  025774          CLR     PSERR           ;PASS SOFT ERR COUNT FOR EOP TYPEOUT
1634
1635  004000  004737  011664          JSR     PC,CLRBAD       ;CLEAR BAD SECTOR/TRACK TABLES
1636
1637  004004  005737  003132          TST     COMP1           ;COMPAT PASS 1?
1638  004010  001004                  BNE     1$              ;BR IF YES
1639  004012  005737  003134          TST     COMP2           ;COMPAT PASS 2?
1640  004016  001001                  BNE     1$              ;BR IF YES
1641  004020  000405                  BR      2$              ;ELSE TEST FOR COPY
1642
1643  004022  012737  000114  003124  1$:     MOV     #114,MAXTRK
1644  004030  000137  006256          JMP     DXTST1          ;COMPATABILITY TESTING
1645
1646  004034  005737  003130  2$:     TST     COPFLG          ;COPY UTILITY?
1647  004040  001402                  BEQ     MEMTST          ;BR IF NO...NORMAL TESTING
1648  004042  000137  016354          JMP     COPY            ;ELSE GO COPY
1649
```

```
1650                                    .SBTTL  PROGRAM
1651                                    ;;*****************************************************************
1652                                    ;       MEMORY TESTS
1653                                    ;
1654                                    ;MEMORY FROM 0 TO THE LAST LOC OF PROGRAM (LSTAD) IS TESTED FOR TIMEOUT.
1655                                    ;MEMORY FROM 'LSTAD' TO THE BOTTOM OF THE RT11 MONITOR/ABS LOADER
1656                                    ;IS TESTED BY A PASS OF A 1010 PATT FOLLOWED BY A PASS OF 0101 PATTERN.
1657                                    ;IF THE SYSTEM HAS A 28K MEMORY, 28K TO 30K WILL BE TESTED SIMILARLY.
1658                                    ;;*****************************************************************
1659
1660  004046  004737  011102           MEMTST: JSR     PC,EXAMKB
1661
1662  004052  012737  012142  000004           MOV     #INTSRV,ERRVEC  ;SETUP TIMEOUT TRAP ADDR
1663  004060  012737  000200  000006           MOV     #200,ERRVEC+2
1664
1665  004066  013703  010766                   MOV     MAXMEM,R3
1666  004072  162703  000300                   SUB     #300,R3         ;TEST ONLY AS FAR AS ABS LOADER IF NOT RT-11
1667
1668  004076  005004                           CLR     R4              ;ADDR POINTER
1669  004100  005037  012150           1$:     CLR     INTFLG
1670  004104  005714                           TST     (R4)            ;TEST LOCS 0 THRU END OF PGM.
1671  004106  005737  012150                   TST     INTFLG          ;TIMEOUT?
1672  004112  001403                           BEQ     2$              ;BR IF NO
1673  004114  010437  004462                   MOV     R4,ADDR         ;FOR ERR TYP
1674  004120  104001                           ERROR   1               ;TIMEOUT ON READ
1675  004122  062704  000002           2$:     ADD     #2,R4           ;BUMP ADDR
1676  004126  020427  026532                   CMP     R4,#LSTAD       ;AT END?
1677  004132  001362                           BNE     1$              ;BR IF NO
1678
1679  004134  005037  004454                   CLR     MEMCT
1680  004140  012737  125252  004460           MOV     #125252,PAT     ;DATA PATT FOR 1'ST PASS
1681  004146  012704  026532           8$:     MOV     #LSTAD,R4       ;ADDR POINTER. R/W LOCS LSTAD THRU
1682                                                                   ;BOT OF RT11 OR BOT OF ABS LOADER
1683  004152  005037  012150           3$:     CLR     INTFLG
1684  004156  013714  004460                   MOV     PAT,(R4)        ;WRITE
1685  004162  005737  012150                   TST     INTFLG          ;TIMEOUT?
1686  004166  001403                           BEQ     4$              ;BR IF NO
1687  004170  010437  004462                   MOV     R4,ADDR         ;FOR ERR TYP
1688  004174  104002                           ERROR   2               ;TIMEOUT ON WRITE
1689  004176  011437  004456           4$:     MOV     (R4),MEMHLD     ;READ
1690  004202  023737  004456  004460           CMP     MEMHLD,PAT      ;COMPARE
1691  004210  001403                           BEQ     6$
1692  004212  010437  004462                   MOV     R4,ADDR         ;FOR ERR TYP
1693  004216  104003                           ERROR   3               ;COMPARE ERROR
1694
1695  004220  062704  000002           6$:     ADD     #2,R4           ;BUMP POINTER
1696  004224  023727  004464  001000           CMP     HLD42,#1000     ;RT11 IF LOC 42 = 1000
1697  004232  001403                           BEQ     7$              ;BR IF RT11
1698
1699  004234  020403                           CMP     R4,R3           ;ELSE AT BOT OF ABS LOADER?
1700  004236  001345                           BNE     3$              ;BR IN NO
1701  004240  000403                           BR      5$              ;ELSE EXIT
```

```
1702
1703  004242  020437  000054          7$:   CMP   R4,54            ;LOC 54 CONTAINS LOW ADDR OF RT11 MONITOR
1704  004246  001341                        BNE   3$              ;BR IN NOT THERE
1705  004250  005737  004454          5$:   TST   MEMCT
1706  004254  001006                        BNE   9$
1707  004256  005237  004454                INC   MEMCT
1708  004262  012737  052525  004460        MOV   #052525,PAT     ;DATA PATT FOR 2'ND PASS
1709  004270  000726                        BR    8$              ;REPEAT
1710
1711  004272  023727  010766  157776  9$:   CMP   MAXMEM,#157776  ;DO WE HAVE 28K?
1712  004300  001047                        BNE   15$             ;BR IF NO & EXIT
1713
1714  004302  012737  125252  004460        MOV   #125252,PAT     ;DATA PATT FOR 1'ST PASS OF HI MEM
1715  004310  012704  160000          10$:  MOV   #160000,R4      ;ADDR PTR TO TOP OF 28K
1716  004314  005037  012150          11$:  CLR   INTFLG
1717  004320  013714  004460              MOV   PAT,(R4)          ;WRITE
1718  004324  005737  012150              TST   INTFLG
1719  004330  001403                        BEQ   12$
1720  004332  010437  004462              MOV   R4,ADDR           ;FOR ERR TYP
1721  004336  104004                        ERROR 4               ;TIMEOUT
1722  004340  011437  004456          12$:  MOV   (R4),MEMHLD     ;READ
1723  004344  023737  004456  004460        CMP   MEMHLD,PAT      ;COMPARE
1724  004352  001403                        BEQ   14$
1725  004354  010437  004462              MOV   R4,ADDR           ;FOR ERR TYP
1726  004360  104005                        ERROR 5               ;COMPARE ERROR
1727  004362  062704  000002          14$:  ADD   #2,R4
1728  004366  020427  167776              CMP   R4,#167776        ;AT END OF 30K?
1729  004372  001350                        BNE   11$             ;BR IF NO
1730
1731  004374  005237  004454              INC   MEMCT
1732  004400  023727  004454  000003        CMP   MEMCT,#3
1733  004406  001404                        BEQ   15$
1734  004410  012737  052525  004460        MOV   #052525,PAT     ;DATA PATT FOR 2'ND HALF OF HI MEM
1735  004416  000734                        BR    10$             ;REPEAT
1736
1737  004420  012737  000006  000004  15$:  MOV   #ERRVEC+2,ERRVEC       ;RESTORE TIMEOUT VECTOR
1738  004426  005037  000006              CLR   ERRVEC+2
1739
1740  004432  032777  010000  174500        BIT   #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
1741  004440  001412                        BEQ   CLKTST
1742  004442  104401  017743              TYPE  ,MEMORY
1743  004446  104401  017756              TYPE  ,DUN
1744  004452  000405                        BR    CLKTST
1745
1746  004454  000000          MEMCT:  0               ;0 = MEM FROM LSTAD TO MONITOR/ABS LOADER W/ 125252 PATT
1747                                                  ;1 - SAME WITH 052525 PATT
1748                                                  ;1 = MEM FROM 28K TO 30K WITH 125252 PAT
1749                                                  ;2 = SAME WITH 052525 PATT
1750                                                  ;3 = EXIT
1751  004456  000000          MEMHLD: 0               ;PUT MEMORY READ HERE
1752  004460  000000          PAT:    0               ;PATT TO BE WRITTEN & READ
1753  004462  000000          ADDR:   0               ;ADDR UNDER TEST
1754  004464  000000          HLD42:  0               ;SAVE LOC 42
1755
1756                          ;;*************** **************************************************
1757                          ;       START LINE CLOCK
```

```
1758                                    ;;*************************************************************
1759
1760  004466  032737  000200  001246  CLKTST: BIT     #BIT7,$DEVM     ;DROP TEST?
1761  004474  001031                          BNE     PRTST           ;BR IF YES
1762  004476  012737  012142  000100          MOV     #INTSRV,CLKVEC  ;SETUP VECTOR AREA
1763  004504  012737  000200  000102          MOV     #200,CLKVEC+2   ;LOCKOUT OTHER INTER.
1764  004512  005037  012150                  CLR     INTFLG
1765  004516  052737  000100  177546          BIS     #IE,CLK         ;SET CLOCK LOOSE!
1766
1767  004524  004537  011316                  JSR     R5,TIMER        ;SEE IF INTFLG GOES TO 100
1768  004530  012150                            INTFLG
1769  004532  000100                            100
1770  004534  104032                          ERROR   32              ;CLK NOT RESPONDING
1771  004536  000410                          BR      PRTST
1772
1773  004540  032777  010000  174372          BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
1774  004546  001404                          BEQ     PRTST
1775  004550  104401  017244                  TYPE    ,CLOCK
1776  004554  104401  017670                  TYPE    ,RUN
```

```
1777
1778                                    ;;*****************************************************************
1779                                    ;       START PRINTER
1780                                    ;;*****************************************************************
1781
1782  004560  032737  100000  001246  PRTST:  BIT     #BIT15,$DEVM    ;DROP TEST?
1783  004566  001056                   BNE     SMTST           ;BR IF YES
1784  004570  005737  010744           TST     PRPRES          ;PRINTER THERE?
1785  004574  001453                   BEQ     SMTST           ;BR IF NO
1786
1787  004576  012737  027014  177420  2$:     MOV     #<PRT!B9600!CHAR8>,PARAM        ;SETUP BAUD RATE & CHAR LENGTH
1788  004604  012737  012152  000200          MOV     #PRSRV,PRVEC    ;ELSE SETUP PRINTER VECTOR
1789  004612  012737  000200  000202          MOV     #200,PRVEC+2    ;LOCKOUT INTERR
1790
1791  004620  005037  012300                   CLR     LINCT           ;CLEAR LINE CTR
1792  004624  012737  000015  012276          MOV     #CR,PRCHR       ;CHAR TO BE PRINTED
1793                                                                  ;'DEL' (177) CAN BE USED TO CLR PRINTER BUFF
1794  004632  052737  000100  177514          BIS     #IE,PRS         ;SET PRINTER LOOSE!
1795                                                                  ;WILL PRINT CONTINUOUS 132 COLUMN LINES
1796                                                                  ;EA BEGINNING WITH ASCII 40 THRU 176
1797  004640  004537  011316                   JSR     R5,TIMER        ;SEE IF LINCT GOES TO 5
1798  004644  012300                             LINCT
1799  004646  000005                             5
1800  004650  104033                           ERROR   33              ;PRINTER NOT RESPONDING
1801  004652  000424                           BR      SMTST
1802
1803  004654  005737  010746                   TST     PRPORT          ;EXT LOOPBACK MODE?
1804  004660  001403                           BEQ     3$              ;BR IF NO
1805  004662  042737  000100  177514          BIC     #IE,PRS         ;ELSE SHUT DOWN PRINTER
1806
1807  004670  032777  010000  174242  3$:     BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
1808  004676  001412                           BEQ     SMTST
1809  004700  104401  017204                   TYPE    ,PRINT
1810  004704  005737  010746                   TST     PRPORT          ;EXT LOOPBACK MODE?
1811  004710  001403                           BEQ     4$              ;BR IF NO
1812  004712  104401  017215                   TYPE    ,PORT
1813  004716  000402                           BR      SMTST
1814  004720  104401  017670          4$:     TYPE    ,RUN
```

```
1815
1816                                    ;;*****************************************************************
1817                                    ;       START SYNC COMM PORT
1818                                    ;;*****************************************************************
1819
1820  004724  032737  002000  001246  SMTST:  BIT     #BIT10,$DEVM    ;DROP TEST?
1821  004732  001402                          BEQ     8$              ;BR IF NO
1822  004734  000137  005200                  JMP     AMTST           ;ELSE JUMP TEST
1823
1824  004740  123727  001210  000001  8$:     CMPB    $ENV #1         ;UNDER APT?
1825  004746  001003                          BNE     9$              ;BR IF NO
1826  004750  005737  001176                  TST     $PASS           ;ELSE 1'ST PASS?
1827  004754  001111                          BNE     AMTST           ;EXIT TEST IF NO
1828
1829  004756  005037  012772          9$:     CLR     LSTCHR          ;LAST CHAR FLAG FOR XMIT
1830  004762  052737  000400  176624          BIS     #MR,SMXC        ;MASTER RESET
1831  004770  052737  004000  176624          BIS     #MCLK,SMXC      ;MAINT CLOCK
1832
1833  004776  032737  000010  001246          BIT     #BIT3,$DEVM     ;EXT LOOPBACK?
1834  005004  001403                          BEQ     1$              ;BR IF YES
1835  005006  052737  014000  176624          BIS     #<MM!MCLK>,SMXC ;ELSE SET MAINT MODE FOR INT. LOOPBACK
1836  005014  012737  007026  176622  1$:     MOV     #<CHR8!ODDPAR!026>,SMPAR        ;SETUP SYNC PARAMETER REGISTER
1837  005022  052737  000026  176620          BIS     #<RTS!DTR!SRSYN>,SMRC   ;SET REQ TO SEND, DATA TERM RDY
1838                                                                          ;& SEARCH SYNC
1839  005030  052737  000020  176624          BIS     #SEND,SMXC      ;ENABLE XMIT SEND
1840  005036  012737  000026  176626          MOV     #026,SMXB       ;XMIT SYNC CHAR
1841
1842  005044  004537  011600                  JSR     R5,TIMER2       ;WAIT FOR DONE
1843  005050  176620                              SMRC
1844  005052  000200                              DONE
1845  005054  104045                          ERROR   45              ;NO DONE
1846  005056  000435                          BR      5$              ;EXIT
1847
1848  005060  013737  176622  012702          MOV     SMRB,COMHLD     ;READ WORD
1849  005066  023727  012702  000026          CMP     COMHLD,#026     ;IS IT SYNC CHAR?
1850  005074  001402                          BEQ     4$              ;BR IF YES
1851  005076  104006                          ERROR   6               ;DID NOT REC SYNC CHAR
1852  005100  000424                          BR      5$              ;EXIT
1853
1854  005102  004537  011272          4$:     JSR     R5,SETVEC       ;SETUP VECTOR AREA
1855  005106  000340                              SMRVEC
1856  005110  012774                              SMRSRV
1857  005112  012704                              SMXSRV
```

```
 1858
 1859  005114  012737  000027  012700          MOV     #027,COMCHR     ;1'ST CHAR TO BE XMITTED
 1860  005122  052737  000100  176620          BIS     #IE,SMRC        ;SET SYNC MODEM LOOSE!
 1861  005130  052737  000100  176624          BIS     #IE,SMXC
 1862
 1863  005136  004537  011316                  JSR     R5,TIMER        ;SEE IF COMCHR GOES TO ALL 1'S (DONE)
 1864  005142  012700                            COMCHR
 1865  005144  177777                            177777
 1866  005146  104037                          ERROR   37              ;SYNC COMM NOT RESPONDING
 1867  005150  000240                          NOP
 1868
 1869  005152  012737  000400  176624  5$:     MOV     #MR,SMXC        ;MASTER RESET
 1870  005160  032777  010000  173752          BIT     #BIT12,@SWR     ;DO TYPEOUT IF SET
 1871  005166  001404                          BEQ     AMTST           ;EXIT IF NOT
 1872  005170  104401  017155                  TYPE    ,SCOM
 1873  005174  104401  017756                  TYPE    ,DUN
```

```
1874
1875                                    ;;*****************************************************************
1876                                    ;       START ASYNC COMM PORT
1877                                    ;;*****************************************************************
1878
1879  005200  032737  004000  001246  AMTST:  BIT     #BIT11,$DEVM    ;DROP TEST?
1880  005206  001054                   BNE     CL1TST          ;BR IF YES
1881
1882  005210  032737  000010  001246          BIT     #BIT3,$DEVM     ;EXT LOOPBACK?
1883  005216  001404                   BEQ     1$              ;BR IF YES
1884  005220  012737  037036  177420          MOV     #<AMOD!B9600!OPAR!CHAR8!MAINT>,PARAM    ;ELSE SET FOR INT LOOPBACK
1885  005226  000403                   BR      3$
1886  005230  012737  037034  177420  1$:     MOV     #<AMOD!B9600!OPAR!CHAR8>,PARAM  ;WAKE UP AMOD IF JUST DID SMOD
1887  005236  052737  000006  176610  3$:     BIS     #<RTS!DTR>,AMRC ;SET REQ TO SEND & DATA TERM RDY
1888
1889  005244  004537  011272                  JSR     R5,SETVEC       ;SETUP VECTOR AREA
1890  005250  000330                           AMRVEC
1891  005252  012622                           AMRSRV
1892  005254  012604                           AMXSRV
1893
1894  005256  005037  012700                  CLR     COMCHR          ;CHAR TO BE XMITTED
1895  005262  013737  176612  012702          MOV     AMRB,COMHLD     ;CLR OUT ANY PREVIOUS STORED WORD
1896  005270  052737  000100  176610          BIS     #IE,AMRC        ;SET ASYNC MODEM LOOSE!
1897  005276  052737  000100  176614          BIS     #IE,AMXC
1898
1899  005304  004537  011316                  JSR     R5,TIMER        ;SEE IF COMCHR GOES TO 377
1900  005310  012700                            COMCHR
1901  005312  000377                            377
1902  005314  104040                   ERROR   40              ;ASYNC COMM NOT RESPONDING
1903  005316  000410                   BR      CL1TST
1904
1905  005320  032777  010000  173612          BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
1906  005326  001404                   BEQ     CL1TST
1907  005330  104401  017170                  TYPE    ,ACOM
1908  005334  104401  017670                  TYPE    ,RUN
```

```
1909
1910                            ;;*****************************************************************
1911                            ;       START CLUSTER TERMINAL #1
1912                            ;;*****************************************************************
1913
1914  005340  005737  010742   CL1TST: TST     CLPRES          ;OPTION PRESENT?
1915  005344  001002                   BNE     1$              ;BR IF YES
1916  005346  000137  006014           JMP     DXTSTO
1917
1918  005352  032737  004000  001246 1$:     BIT     #4000,$DEVM     ;DROP ASYNC?
1919  005360  001406                   BEQ     4$              ;BR IF NO
1920  005362  042737  000006  176610         BIC     #<RTS!DTR>,AMRC ;MUST BE TOGGLED
1921  005370  052737  000006  176610         BIS     #<RTS!DTR>,AMRC ;PRIME IT
1922
1923  005376  032737  010000  001246 4$:     BIT     #BIT12,$DEVM    ;DROP TEST?
1924  005404  001051                   BNE     CL2TST          ;BR IF YES
1925
1926  005406  032737  000001  001246         BIT     #BIT0,$DEVM     ;TERM #1 SET FOR EXT LOOPBACK?
1927  005414  001404                   BEQ     2$              ;BR IF YES
1928  005416  012737  005016  177420         MOV     #<CT1!B2400!CHAR8!MAINT>,PARAM  ;ELSE SET FOR INT LOOPBACK
1929  005424  000403                   BR      3$
1930  005426  012737  005014  177420 2$:     MOV     #<CT1!B2400!CHAR8>,PARAM        ;SETUP FOR EXT LOOPBACK
1931
1932  005434  004537  011272         3$:     JSR     R5,SETVEC       ;SETUP INTERR VECTORS
1933  005440  000300                           TK1VEC
1934  005442  012322                           TK1SRV
1935  005444  012304                           TP1SRV
1936
1937  005446  005037  012400                 CLR     T1CHR           ;CHAR TO BE XMITTED
1938  005452  013737  176502  012402         MOV     TK1B,T1HLD      ;READ CHAR (CLEAR STALE DATA)
1939  005460  052737  000100  176500         BIS     #IE,TK1S        ;SET CLUSTER TERM #1 LOOSE!
1940  005466  052737  000100  176504         BIS     #IE,TP1S
1941
1942  005474  004537  011316                 JSR     R5,TIMER        ;SEE IF T1CHR GOES TO 377
1943  005500  012400                           T1CHR
1944  005502  000377                           377
1945  005504  104034                 ERROR   34              ;CLUSTER TERM 1 NOT RESPONDING
1946  005506  000410                 BR      CL2TST
1947
1948  005510  032777  010000  173422         BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
1949  005516  001404                   BEQ     CL2TST
1950  005520  104401  017122                 TYPE    ,CLT1
1951  005524  104401  017670                 TYPE    ,RUN
```

```
                                ;;*****************************************************************
                                ;       START CLUSTER TERMINAL #2
                                ;;*****************************************************************

005530  032737  020000  001246  CL2TST: BIT     #BIT13,$DEVM    ;DROP TEST?
005536  001051                          BNE     CL3TST          ;BR IF YES

005540  032737  000002  001246          BIT     #BIT1,$DEVM     ;TERM #2 SET FOR EXT LOOPBACK?
005546  001404                          BEQ     1$              ;BR IF YES
005550  012737  055016  177420          MOV     #<CT2!B2400!CHAR8!MAINT>,PARAM  ;ELSE SET FOR INT LOOPBACK
005556  000403                          BR      2$
005560  012737  055014  177420  1$:     MOV     #<CT2!B2400!CHAR8>,PARAM        ;SETUP FOR EXT LOOPBACK

005566  004537  011272          2$:     JSR     R5,SETVEC       ;SETUP INTERR VECTORS
005572  000310                              TK2VEC
005574  012422                              TK2SRV
005576  012404                              TP2SRV

005600  005037  012500                      CLR     T2CHR           ;CHAR TO BE XMITTED
005604  013737  176512  012502          MOV     TK2B,T2HLD      ;READ CHAR (CLEAR STALE DATA)
005612  052737  000100  176510          BIS     #IE,TK2S        ;SET CLUSTER TERM #2 LOOSE.
005620  052737  000100  176514          BIS     #IE,TP2S

005626  004537  011316                  JSR     R5,TIMER        ;SEE IF T2CHR GOES TO 377
005632  012500                            T2CHR
005634  000377                            377
005636  104035                          ERROR   35              ;CLUSTER TERM 2 NOT RESPONDING
005640  000410                          BR      CL3TST

005642  032777  010000  173270          BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
005650  001404                          BEQ     CL3TST
005652  104401  017133                  TYPE    ,CLT2
005656  104401  017670                  TYPE    ,RUN
```

```
                                ;;*****************************************************************
                                ;       START CLUSTER TERMINAL #3
                                ;;*****************************************************************

005662  032737  040000  001246  CL3TST: BIT     #BIT14,$DEVM    ;DROP TEST?
005670  001051                          BNE     DXTST0          ;BR IF YES

005672  032737  000004  001246          BIT     #BIT2,$DEVM     ;TERM #3 SET FOR EXT LOOPBACK?
005700  001404                          BEQ     1$              ;BR IF YES
005702  012737  065016  177420          MOV     #<CT3!B2400!CHAR8!MAINT>,PARAM  ;ELSE SET FOR INT LOOPBACK
005710  000403                          BR      2$
005712  012737  065014  177420  1$:     MOV     #<CT3!B2400.CHAR8>,PARAM        ;SETUP FOR EXT LOOPBACK

005720  004537  011272          2$:     JSR     R5,SETVEC       ;SETUP INTERR VECTORS
005724  000320                              TK3VEC
005726  012522                              TK3SRV
005730  012504                              TP3SRV

005732  005037  012600                      CLR     T3CHR           ;CHAR TO BE XMITTED
005736  013737  176522  012602          MOV     TK3B,T3HLD      ;READ CHAR (CLEAR STALE DATA)
005744  052737  000100  176520          BIS     #IE,TK3S        ;SET CLUSTER TERM #3 LOOSE.
005752  052737  000100  176524          BIS     #IE,TP3S

005760  004537  011316                  JSR     R5,TIMER        ;SEE IF T3CHR GOES TO 377
005764  012600                            T3CHR
005766  000377                            377
005770  104036                          ERROR   36              ;CLUSTER TERM 3 NOT RESPONDING
005772  000410                          BR      DXTST0

005774  032777  010000  173136          BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
006002  001404                          BEQ     DXTST0
006004  104401  017144                  TYPE    ,CLT3
006010  104401  017670                  TYPE    ,RUN
```

```
2020
2021                                    ;;*****************************************************************
2022                                    ;       FILL & EMPTY BUFFER TEST
2023                                    ;
2024                                    ;A FILL BUFFER COMMAND IS ISSUED WITH ALTERNATING WORDS OF ALL 0'S & ALL 1'S.
2025                                    ;AN EMPTY BUFFER COMMAND IS ISSUED TO FILL 'DOBUF'.
2026                                    ;UPON COMPLETION, DOBUF IS CHECKED FOR ALTERNATING WORDS OF ALL 0'S & ALL 1'S.
2027                                    ;ERRORS IN THIS TEST MAY INDICATE CABLE CROSS-TALK INTERFERENCE.
2028                                    ;;*****************************************************************
2029
2030  006014  005037  003120           DXTST0: CLR     DXTST           ;0 = DXTST0
2031
2032  006020  032737  000400  001246           BIT     #BIT8,$DEVM     ;DROP DX0 TESTS?
2033  006026  001404                           BEQ     7$              ;BR IF NO
2034  006030  032737  001000  001246           BIT     #BIT9,$DEVM     ;DROP DX1 TESTS?
2035  006036  001107                           BNE     DXTST1          ;BR IF YES
2036
2037  006040  004537  011600           7$:     JSR     R5,TIMER2       ;WAIT FOR DISK DONE
2038  006044  177170                               RXCS
2039  006046  000200                               DONE
2040  006050  104075                           ERROR   75              ;NO DONE
2041  006052  000474                           BR      8$              ;EXIT
2042
2043  006054  012737  000011  177170           MOV     #INITAL,RXCS    ;DO AN INITIALIZE COMMAND
2044
2045  006062  004537  011600                   JSR     R5,TIMER2       ;WAIT FOR DONE
2046  006066  177170                               RXCS
2047  006070  000200                               DONE
2048  006072  104075                           ERROR   75              ;NO DONE
2049  006074  000463                           BR      8$
2050                                                                   ;FILL BUFFER
2051  006076  012700  000100                   MOV     #100,R0         ;WORD CT
2052  006102  005001                           CLR     R1              ;WORD
2053  006104  012737  000001  177170           MOV     #FBUF,RXCS      ;ISSUE FILL BUFF COMMAND
2054  006112  010137  177172           1$:     MOV     R1,RXDB
2055  006116  005101                           COM     R1              ;COMPLEMENT WORD
2056  006120  005300                           DEC     R0              ;DONE ALL 100 WORDS?
2057  006122  001373                           BNE     1$              ;BR IF NO
2058
2059  006124  004537  011600                   JSR     R5,TIMER2       ;WAIT FOR DISK DONE
2060  006130  177170                               RXCS
2061  006132  000200                               DONE
2062  006134  104075                           ERROR   75              ;NO DONE
2063  006136  000442                           BR      8$              ;EXIT
2064
2065                                                                   ;EMPTY BUFFER
2066  006140  012700  000100                   MOV     #100,R0         ;WORD CT
2067  006144  012701  002424                   MOV     #DOBUF,R1       ;BUFFER ADDR
2068  006150  012737  000003  177170           MOV     #EBUF,RXCS      ;ISSUE EMPTY BUFF COMMAND
2069  006156  013721  177172           2$:     MOV     RXDB,(R1)+      ;STORE WORD
2070  006162  005300                           DEC     R0              ;DONE?
2071  006164  001374                           BNE     2$              ;BR IF NO
```

```
2072
2073  006166  004537  011600              JSR     R5,TIMER2       ;WAIT FOR DISK DONE
2074  006172  177170                        RXCS
2075  006174  000200                        DONE
2076  006176  104075                      ERROR   75              ;NO DONE
2077  006200  000421                      BR      8$              ;EXIT
2078
2079                                                              ;CHECK IT
2080  006202  012701  002424              MOV     #DOBUF,R1       ;BUFFER ADDR
2081  006206  012137  006250      3$:     MOV     (R1)+,EBHLD     ;STORE IT
2082  006212  001401                      BEQ     4$              ;BR IF ZERO
2083  006214  104076                      ERROR   76              ;WORD NOT ALL ZEROS
2084
2085  006216  012137  006250      4$:     MOV     (R1)+,EBHLD
2086  006222  023727  006250  177777      CMP     EBHLD,#-1       ;ALL 1'S?
2087  006230  001401                      BEQ     5$              ;BR IF YES
2088  006232  104077                      ERROR   77              ;WORD NOT ALL ONES
2089
2090  006234  020127  002624      5$:     CMP     R1,#DOBUF+200   ;END OF DOBUFF?
2091  006240  001362                      BNE     3$              ;BR IF NO
2092  006242  000405                      BR      DXTST1          ;GO TO NXT TST
2093
2094  006244  000137  010200      8$:     JMP     EOP             ;EXIT ALL DISK TESTS
2095
2096  006250  000000          EBHLD:  0                       ;STORAGE        FOR ERROR REPORT
2097  006252  000000          EXP0:   0                       ;EXPECT 0'S     FOR ERROR REPORT
2098  006254  177777          EXP1:   -1                      ;EXPECT 1'S     FOR ERROR REPORT
```

```
2099
2100                                    ;;*****************************************************************
2101                                    ;       START DX0 & DX1 DATA PATTERN
2102                                    ;
2103                                    ;EACH SECTOR IS FILLED WITH ITS TRACK & SECTOR ADDRESS AS DATA.
2104                                    ;DX1 WILL ALWAYS HAVE ITS DATA BIT 15 SET TO DISTINGUISH IT FROM DX0.
2105                                    ;EACH TRACK IS READ & CHECKED BEFORE GOING TO THE NEXT TRACK.
2106                                    ;DATA IS WRITTEN ON ALTERANTE SECTORS TO AVOID LATENCY TIMES.
2107                                    ;IE: SECTORS 1,3,5...31 FOLLOWED BY 2,4,6...32.
2108                                    ;HARD ERRORS ARE THOSE ERRORS AFTER 10 MORE RETRIES.
2109                                    ;1'ST PASS PERFORMED ON TRACKS 0-4, 40-44, 72-76(10)
2110                                    ;SUBSEQUENT PASSES ON ALL TRACKS
2111                                    ;;*****************************************************************
2112
2113  006256  012737  000001  003120  DXTST1: MOV     #1,DXTST        ;1 - DATA PATT TESTS
2114
2115  006264  005037  002406            CLR     D0TRK           ;STARTING TRACK
2116  006270  005037  002656            CLR     D1TRK
2117
2118  006274  005037  002400    12$:    CLR     D0PAT           ;INIT DX0 FLAGS
2119  006300  005037  002404            CLR     D0ERR
2120  006304  005037  002414            CLR     D0CERR
2121  006310  012737  000001  002410    MOV     #1,D0SEC
2122
2123  006316  005037  002650            CLR     D1PAT           ;INIT DX1 FLAGS
2124  006322  005037  002654            CLR     D1ERR
2125  006326  005037  002664            CLR     D1CERR
2126  006332  012737  000001  002660    MOV     #1,D1SEC
2127
2128  006340  032737  000400  001246    BIT     #BIT8,$DEVM     ;DROP DX0 TESTS?
2129  006346  001031                    BNE     2$              ;BR IF YES
2130
2131  006350  012737  013074  000264    MOV     #RXSRV,RXVEC    ;SETUP VECTOR AREA
2132  006356  012737  000200  000266    MOV     #200,RXVEC+2
2133
2134  006364  005737  003134    1$:     TST     COMP2           ;DOING COMPAT PASS 2?
2135  006370  001404                    BEQ     10$             ;BR IF NO
2136  006372  012737  013566  013174    MOV     #DORSEC,DODISP  ;ELSE DO READS ONLY
2137  006400  000403                    BR      7$
2138
2139  006402  012737  013212  013174  10$:     MOV     #DOFBUF,DODISP  ;DO NORMAL TESTING
2140  006410  005037  003122    7$:     CLR     FIN             ;DO DX1 WHEN SET
2141  006414  052737  000100  177170    BIS     #IE,RXCS        ;LET INTERRUPTS LOOSE!
2142
2143  006422  004737  011402            JSR     PC,TIMDO
2144  006426  104041                    ERROR   41              ;DX0 NOT GETTING TRACK DONE FLAG
2145  006430  000240                    NOP
2146
2147  006432  032737  001000  001246  2$:      BIT     #BIT9,$DEVM     ;DROP DX1 TESTS?
2148  006440  001031                    BNE     4$              ;BR IF YES
2149
2150  006442  012737  013074  000264            MOV     #RXSRV,RXVEC    ;SETUP VECTOR AREA
2151  006450  012737  000200  000266            MOV     #200,RXVEC+2
```

```
2152
2153  006456  005737  003134          3$:     TST     COMP2           ;DOING COMPAT PASS 2?
2154  006462  001404                          BEQ     11$             ;BR IF NO
2155  006464  012737  015352  013202          MOV     #D1RSEC,D1DISP  ;ELSE DO READ ONLY
2156  006472  000403                          BR      8$
2157
2158  006474  012737  014776  013202  11$:     MOV     #D1FBUF,D1DISP  ;DO NORMAL TESTING
2159  006502  005037  003122          8$:     CLR     FIN             ;GO BACK TO DX0 WHEN SET
2160  006506  052737  000120  177170          BIS     #<IE.DX1>,RXCS  ;LET INTERRUPTS LOOSE.
2161
2162  006514  004737  011476                  JSR     PC,TIMD1
2163  006520  104042                          ERROR   42              ;DX1 NOT GETTING TRACK DONE FLAG
2164  006522  000240                          NOP
2165
2166  006524  032737  000400  001246  4$:     BIT     #BIT8,$DEVM     ;DROP DX0?
2167  006532  001007                          BNE     5$              ;BR IF YES
2168  006534  005737  002400                  TST     D0PAT           ;ELSE IS DX0 ALL DONE?
2169  006540  001004                          BNE     5$              ;BR IF YES
2170  006542  005737  011474                  TST     D0TMO           ;TIMEOUT?
2171  006546  001706                          BEQ     1$              ;BR IF NO
2172  006550  000717                          BR      7$              ;ELSE CONT LAST COMMAND
2173
2174  006552  032737  001000  001246  5$:     BIT     #BIT9,$DEVM     ;DROP DX1?
2175  006560  001007                          BNE     6$              ;BR IF YES
2176  006562  005737  002650                  TST     D1PAT           ;ELSE IS DX1 ALL DONE?
2177  006566  001004                          BNE     6$              ;BR IF YES
2178  006570  005737  011564                  TST     D1TMO           ;TIMEOUT?
2179  006574  001730                          BEQ     3$              ;BR IF NO
2180  006576  000741                          BR      8$              ;ELSE CONT LAST COMMAND
2181
2182  006600  005737  003132          6$:     TST     COMP1           ;DOING COMPAT PASS 1?
2183  006604  001402                          BEQ     13$             ;BR IF NO
2184  006606  000137  010200                  JMP     EOP             ;ELSE ALL DONE
2185
2186  006612  005737  001176          13$:    TST     $PASS           ;1'ST PASS?
2187  006616  001035                          BNE     DXTST2          ;BR IF NO
2188
2189  006620  023727  003124  000004          CMP     MAXTRK,#4       ;JUST DID 5 OUTER TRACKS?
2190  006626  001012                          BNE     14$             ;BR IF NO
2191  006630  012737  000050  002406          MOV     #40.,D0TRK      ;ELSE  SETUP FOR 5 MIDDLE TRKS
2192  006636  012737  000050  002656          MOV     #40.,D1TRK
2193  006644  012737  000054  003124          MOV     #44.,MAXTRK
2194  006652  000610                          BR      12$             ;GO DO IT.
2195
2196  006654  023727  003124  000054  14$:    CMP     MAXTRK,#44.     ;JUST DID 5 MIDDLE TRACKS?
2197  006662  001013                          BNE     DXTST2          ;BR IF NO, MUST BE ALL DONE
2198  006664  012737  000110  002406          MOV     #72.,D0TRK      ;ELSE SETUP FOR 5 INNER TRKS
2199  006672  012737  000110  002656          MOV     #72.,D1TRK
2200  006700  012737  000114  003124          MOV     #76.,MAXTRK
2201  006706  000137  006274                  JMP     12$             ;DO IT.
2202
```

```
2203
2204                                    ;;*****************************************************************
2205                                    ;       START DX0 & DX1 RANDOM SEEKS
2206                                    ;
2207                                    ;RANDOM SEEKS ARE PERFORMED ON EACH DISK.
2208                                    ;DATA IS READ & VERIFIED TO BE CORRECT FROM THE PREVIOUS TEST.
2209                                    ;HARD ERRORS ARE THOSE ERRORS AFTER 10 MORE RETRIES.
2210                                    ;1'ST PASS PERFORMS 20 RANDOM SEEKS ON 1'ST 5 TRACKS.
2211                                    ;SUBSEQUENT PASSES PERFORMS 500 RANDOM SEEKS BETWEEN ALL TRACKS.
2212                                    ;;*****************************************************************
2213
2214  006712  012737  000002  003120  DXTST2: MOV     #2,DXTST        ;2 = RANDOM SEEK TESTS
2215  006720  012737  000004  003124          MOV     #4,MAXTRK       ;INIT TO GO FROM TRKS 0-4
2216
2217  006726  005037  002404                  CLR     D0ERR           ;INIT DX0 FLAGS
2218  006732  005037  002414                  CLR     D0CERR
2219  006736  005037  002402                  CLR     D0CNT
2220
2221  006742  005037  002654                  CLR     D1ERR           ;INIT DX1 FLAGS
2222  006746  005037  002664                  CLR     D1CERR
2223  006752  005037  002652                  CLR     D1CNT
2224
2225  006756  032737  000400  001246          BIT     #BIT8,$DEVM     ;DROP DX0 TESTS?
2226  006764  001041                          BNE     2$              ;BR IF YES
2227
2228  006766  005037  011072          1$:     CLR     LOLIM
2229  006772  013737  003124  011074          MOV     MAXTRK,HILIM
2230  007000  004737  010770                  JSR     PC,RAND
2231  007004  010137  002406                  MOV     R1,D0TRK        ;GET RANDOM TRACK #
2232
2233  007010  005237  011072                  INC     LOLIM
2234  007014  012737  000032  011074          MOV     #32,HILIM
2235  007022  004737  010770                  JSR     PC,RAND
2236  007026  010137  002410                  MOV     R1,D0SEC        ;GET RANDOM SECTOR #
2237
2238  007032  004537  011754                  JSR     R5,CK0BAD       ;SEE IF THIS TRK/SEC FLAGGED BAD
2239  007036  000753                              BR  1$              ;BR IF RETURN HERE
2240
2241  007040  012737  013566  013174          MOV     #D0RSEC,D0DISP  ;SET DISPATCH TABLE TO POINT TO
2242                                                                  ;READ SECTOR HANDLER.
2243  007046  005037  003122                  CLR     FIN             ;DO DX1 WHEN SET
2244  007052  052737  000100  177170          BIS     #IE,RXCS        ;LET INTERRUPTS LOOSE!
2245
2246  007060  004737  011402                  JSR     PC,TIMD0
2247  007064  104041                          ERROR   41              ;DX0 NOT GETTING DONE FLAG
2248  007066  000240                          NOP
```

```
2249
2250  007070  032737  001000  001246  2$:     BIT     #BIT9,$DEVM     ;DROP DX1 TESTS?
2251  007076  001041                          BNE     4$              ;BR IF YES
2252
2253  007100  005037  011072          3$:     CLR     LOLIM
2254  007104  013737  003124  011074          MOV     MAXTRK,HILIM
2255  007112  004737  010770                  JSR     PC,RAND
2256  007116  010137  002656                  MOV     R1,D1TRK        ;GET RANDOM TRACK #
2257
2258  007122  005237  011072                  INC     LOLIM
2259  007126  012737  000032  011074          MOV     #32,HILIM
2260  007134  004737  010770                  JSR     PC,RAND
2261  007140  010137  002660                  MOV     R1,D1SEC        ;GET RANDOM SECTOR #
2262
2263  007144  004537  012066                  JSR     R5,CK1BAD       ;SEE IF THIS TRK/SEC FLAGGED BAD
2264  007150  000753                          BR      3$              ;BR IF RETURN HERE
2265
2266  007152  012737  015352  013202          MOV     #D1RSEC,D1DISP  ;SET DISPATCH TABLE TO POINT TO
2267                                                                  ;READ SECTOR HANDLER.
2268  007160  005037  003122                  CLR     FIN             ;GO BACK TO DX0 WHEN SET
2269  007164  052737  000120  177170          BIS     #<IE!DX1>,RXCS  ;LET INTERRUPTS LOOSE!
2270
2271  007172  004737  011476                  JSR     PC,TIMD1
2272  007176  104042                          ERROR   42              ;DX1 NOT GETTING DONE FLAG
2273  007200  000240                          NOP
2274
2275  007202  032737  000400  001246  4$:     BIT     #BIT8,$DEVM     ;DROP DX0?
2276  007210  001004                          BNE     5$              ;BR IF YES
2277  007212  023737  002402  003126          CMP     DOCNT,MAXSK     ;DID ALL SEEKS?
2278  007220  001262                          BNE     1$              ;BR IF NO
2279
2280  007222  032737  001000  001246  5$:     BIT     #BIT9,$DEVM     ;DROP DX1?
2281  007230  001004                          BNE     6$              ;BR IF YES
2282  007232  023737  002652  003126          CMP     D1CNT,MAXSK     ;DID ALL SEEKS?
2283  007240  001317                          BNE     3$              ;BR IF NO
2284
2285  007242  005737  003134          6$:     TST     COMP2           ;DOING COMPAT PASS 2?
2286  007246  001402                          BEQ     DXTST3          ;BR IF NO
2287  007250  000137  010200                  JMP     EOP             ;ELSE ALL DONE
2288
```

```
2289
2290                            ;;*****************************************************************
2291                            ;       INITIALIZE COMMAND TEST
2292                            ;
2293                            ;THE INITIALIZE  & EMPTY BUFFER COMMAND IS ISSUED TO DX0.
2294                            ;DATA IS VERIFIED TO BE THAT OF TRACK 1, SECTOR 1.
2295                            ;HARD ERRORS ARE THOSE ERRORS AFTER 10 MORE RETRIES.
2296                            ; 'INIT' CANNOT BE ISSUED TO DX1.
2297                            ;;*****************************************************************
2298
2299  007254  032737  000400  001246  DXTST3: BIT     #BIT8,$DEVM     ;DROP DX0 TESTS?
2300  007262  001062                          BNE     DXTST4          ;BR IF YES
2301
2302  007264  012737  000003  003120          MOV     #3,DXTST        ;3 = INIT TEST
2303  007272  005037  002404                  CLR     DOERR           ;FLAGS
2304  007276  005037  002414                  CLR     DOCERR
2305
2306  007302  012737  014632  013174          MOV     #DOINIT,DODISP  ;SET DISPATCH TABLE TO GO TO INIT HANDLER
2307
2308  007310  005037  003122                  CLR     FIN
2309  007314  052737  000100  177170          BIS     #IE,RXCS        ;LET LOOSE
2310
2311  007322  004737  011402                  JSR     PC,TIMDO
2312  007326  104046                          ERROR   46              ;DX0 HUNG ON INITIALIZE COMMAND
2313  007330  000437                          BR      DXTST4
2314
2315  007332  005737  177170                  TST     RXCS            ;ERROR?
2316  007336  100001                          BPL     1$              ;BR IF NO
2317  007340  104063                          ERROR   63              ;INIT COMMAND ERROR
2318
2319  007342  032737  000001  177172  1$:     BIT     #ID,RXES        ;INIT DONE BIT SET?
2320  007350  001001                          BNE     2$              ;BR IF YES
2321  007352  104055                          ERROR   55              ;INIT DONE NOT SET
2322
2323  007354  005037  002404          2$:     CLR     DOERR
2324  007360  005037  002414                  CLR     DOCERR
2325  007364  012737  000001  002406          MOV     #1,DOTRK        ;EXP TRK & SEC
2326  007372  012737  000001  002410          MOV     #1,DOSEC
2327
2328  007400  012737  014046  013174          MOV     #DOEBUF,DODISP  ;SETUP DISPATCH TABLE TO
2329                                                                  ;EMPTY BUFFER & VERIFY DATA
2330  007406  005037  003122                  CLR     FIN
2331  007412  052737  000100  177170          BIS     #IE,RXCS        ;LET LOOSE
2332
2333  007420  004737  011402                  JSR     PC,TIMDO
2334  007424  104056                          ERROR   56              ;DX0 HUNG ON EMPTY BUFF COMMAND
2335  007426  000240                          NOP
```

```
2336
2337
2338                                    ;;*****************************************************************
2339                                    ;       RESTORE COMMAND TEST
2340                                    ;
2341                                    ;THIS TEST VERIFIES THAT THE RESTORE COMMAND CAN BE ISSUED
2342                                    ;TO DX0 & DX1 & THAT NO ERRORS RESULT FROM IT.
2343                                    ;;*****************************************************************
2344  007430  032737  000400  001246  DXTST4: BIT     #BIT8,$DEVM     ;DROP DX0 TESTS?
2345  007436  001121                          BNE     DXTST5          ;BR IF YES
2346
2347  007440  012737  000004  003120          MOV     #4,DXTST        ;4 = RESTORE TEST
2348  007446  012737  014656  013174          MOV     #DOREST,DODISP  ;SETUP DISPATCH TABLE
2349  007454  005037  003122                  CLR     FIN
2350  007460  052737  000100  177170          BIS     #IE,RXCS
2351
2352  007466  004737  011402                  JSR     PC,TIMDO
2353  007472  104047                          ERROR   47              ;DX0 HUNG ON RESTORE COMMAND
2354  007474  000431                          BR      2$
2355
2356  007476  005737  177170                  TST     RXCS            ;ERROR?
2357  007502  100001                          BPL     1$              ;BR IF NO
2358  007504  104054                          ERROR   54              ;DX0 RESTORE COMMAND ERROR
2359
2360  007506  005037  002406          1$:     CLR     DOTRK           ;CHK FOR RECORD NOT FOUND (RNF)
2361  007512  012737  000001  002410          MOV     #1,DOSEC        ;SET TRK/SEC. SHOULD NOT MOVE
2362                                                                  ;LOGIC THINKS ITS ALREADY THERE.
2363  007520  005037  002404                  CLR     DOERR
2364  007524  005037  002414                  CLR     DOCERR
2365  007530  005037  003122                  CLR     FIN
2366  007534  012737  013566  013174          MOV     #DORSEC,DODISP  ;SETUP TO READ SECTOR
2367  007542  052737  000100  177170          BIS     #IE,RXCS        ;LET INTERR LOOSE
2368
2369  007550  004737  011402                  JSR     PC,TIMDO
2370  007554  104041                          ERROR   41              ;DX0 HUNG ON READ CMD
2371  007556  000240                          NOP
2372
2373  007560  032737  001000  001246  2$:     BIT     #BIT9,$DEVM     ;DROP DX1 TESTS?
2374  007566  001045                          BNE     DXTST5          ;BR IF YES
2375
2376  007570  012737  014674  013202          MOV     #D1REST,D1DISP  ;REPEAT FOR DX1
2377  007576  005037  003122                  CLR     FIN
2378  007602  052737  000120  177170          BIS     #<IE!DX1>,RXCS  ;LET INTERR LOOSE
2379
2380  007610  004737  011476                  JSR     PC,TIMD1
2381  007614  104042                          ERROR   42              ;DX1 HUNG ON RESTORE CMD
2382  007616  000431                          BR      DXTST5
2383
2384  007620  005737  177170                  TST     RXCS            ;ERROR?
2385  007624  100001                          BPL     3$              ;BR IF NO
2386  007626  104074                          ERROR   74              ;DX1 RESTORE CMD ERROR
```

```
2387
2388  007630  005037  002656          3$:     CLR     D1TRK           ;READ TO CHECK FOR RNF
2389  007634  012737  000001  002660          MOV     #1,D1SEC
2390  007642  005037  002654                  CLR     D1ERR
2391  007646  005037  002664                  CLR     D1CERR
2392  007652  005037  003122                  CLR     FIN
2393  007656  012737  015352  013202          MOV     #D1RSEC,D1DISP  ;SETUP TO READ SECTOR
2394  007664  052737  000120  177170          BIS     #<IE!DX1>,RXCS  ;LET INTERR LOOSE
2395
2396  007672  004737  011476                  JSR     PC,TIMD1
2397  007676  104042                          ERROR   42              ;DX1 HUNG ON READ CMD
2398  007700  000240                          NOP
2399
2400
2401
2402                                  ;;*****************************************************************
2403                                  .       WRITE SECTOR WITH DELETED DATA MARK TEST
2404                                  ;
2405                                  ;THIS TEST VERIFIES THAT THE WRITE DELETED DATA COMMAND CAN BE ISSUED
2406                                  ;& THAT THE 'DELETED DATA' BIT 5 IS SET IN RXCS AFTER READY IS RECV'D.
2407                                  ;HARD ERRORS ARE THOSE ERRORS AFTER 10 MORE RETRIES.
2408                                  ;;*****************************************************************
2409
2410  007702  032737  000400  001246  DXTST5: BIT     #BIT8,$DEVM     ;DROP DX0 TESTS?
2411  007710  001043                          BNE     DXTST6          ;BR IF YES
2412
2413  007712  012737  000005  003120          MOV     #5,DXTST        ;5 = WR DEL DATA TEST
2414  007720  005037  002404                  CLR     D0ERR
2415  007724  005037  002414                  CLR     D0CERR
2416
2417  007730  012737  013212  013174          MOV     #D0FBUF,D0DISP  ;SETUP DISPATCH TABLE
2418
2419  007736  005037  003122                  CLR     FIN
2420  007742  052737  000100  177170          BIS     #IE,RXCS        ;LET LOOSE
2421
2422  007750  004737  011402                  JSR     PC,TIMD0
2423  007754  104050                          ERROR   50              ;DX0 HUNG ON WRITE DEL DATA CMD
2424  007756  000420                          BR      DXTST6
2425
2426  007760  012737  014712  013174          MOV     #D0STAT,D0DISP  ;SETUP DISPATCH TABLE
2427
2428  007766  005037  003122                  CLR     FIN
2429  007772  052737  000100  177170          BIS     #IE,RXCS        ;LET LOOSE
2430
2431  010000  004737  011402                  JSR     PC,TIMD0
2432  010004  104051                          ERROR   51              ;DX0 HUNG ON READ STATUS
2433  010006  000404                          BR      DXTST6
2434
2435  010010  005737  177170                  TST     RXCS            ;ERROR?
2436  010014  100001                          BPL     .+4             ;BR IF NO
2437  010016  104057                          ERROR   57              ;READ STATUS ERROR
```

```
2438
2439                                    ;;*****************************************************************
2440                                    ;       INVALID ADDRESS TESTS
2441                                    ;
2442                                    ;A WRITE SECTOR COMMAND IS ISSUED TO INVALID TRACK 115(8)
2443                                    ;& RXCS IS CHECKED FOR THE INVALID ADDR BIT 1 TO BE SET AFTER RDY.
2444                                    ;THE ABOVE IS REPEATED FOR INVALID SECTOR 33(8).
2445                                    ;;*****************************************************************
2446
2447  010020  032737  000400  001246  DXTST6: BIT     #BIT8,$DEVM     ;DROP DXC TESTS?
2448  010026  001062                          BNE     4$              ;BR IF YES
2449
2450  010030  005037  010176                  CLR     INVCT
2451  010034  012737  000006  003120          MOV     #6,DXTST        ;6 = INV ADDR TESTS
2452  010042  012737  000115  002406          MOV     #115,DOTRK
2453  010050  012737  000001  002410          MOV     #1,DOSEC
2454  010056  012737  046401  177174          MOV     #46401,RXSA     ;INVALID TRACK
2455
2456  010064  012737  014730  013174  1$:     MOV     #DOINV,DODISP   ;SETUP DISPATCH TABLE
2457
2458  010072  005037  003122                  CLR     FIN
2459  010076  052737  000100  177170          BIS     #IE,RXCS        ;LET LOOSE
2460
2461  010104  004737  011402                  JSR     PC,TIMDO
2462  010110  104060                          ERROR   60              ;DXO HUNG ON INVALID ADDR
2463  010112  000432                          BR      EOP
2464
2465  010114  032737  000002  177172          BIT     #INVADR,RXES    ;INV ADDR BIT SET?
2466  010122  001002                          BNE     2$              ;BR IF YES
2467  010124  104061                          ERROR   61              ;INVALID ADDR BIT NOT SET
2468  010126  000404                          BR      3$
2469
2470  010130  005737  177170          2$:     TST     RXCS            ;ERROR BIT SET?
2471  010134  100401                          BMI     3$              ;BR IF YES
2472  010136  104062                          ERROR   62              ;ERROR BIT NOT SET
2473
2474  010140  005737  010176          3$:     TST     INVCT           ;DID WE JUST DO INV SECTOR?
2475  010144  001013                          BNE     4$              ;BR IF YES...DONE
2476
2477  010146  005237  010176                  INC     INVCT
2478  010152  005037  002406                  CLR     DOTRK
2479  010156  012737  000033  002410          MOV     #33,DOSEC
2480  010164  012737  000033  177174          MOV     #33,RXSA        ;INVALID SECTOR
2481  010172  000734                          BR      1$              ;REPEAT FOR INVALID SECTOR
2482
2483  010174  000401                  4$:     BR      .+4
2484
2485  010176  000000                  INVCT:  0                       ;0   DOING INVALID TRACK
2486                                                                  ;1   DOING INVALID SECTOR
2487
```

```
2488                                ;;*****************************************************************
2489                                ;       END OF PASS
2490                                ;;*****************************************************************
2491
2492  010200  005237  001176       EOP:    INC     $PASS           ;PASS CTR
2493  010204  104401  017764               TYPE    ,ENDPAS
2494  010210  013746  001176               MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
2495  010214  104405                       TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
2496  010216  104401  020001               TYPE    ,MSG1
2497  010222  013746  025766               MOV     TERR,-(SP)      ;;SAVE TERR FOR TYPEOUT
2498  010226  104405                       TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
2499  010230  104401  020030               TYPE    ,MSG2
2500  010234  013746  025770               MOV     TSERR,-(SP)     ;;SAVE TSERR FOR TYPEOUT
2501  010240  104405                       TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
2502  010242  104401  020064               TYPE    ,MSG3
2503  010246  013746  025772               MOV     PERR,-(SP)      ;;SAVE PERR FOR TYPEOUT
2504  010252  104405                       TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
2505  010254  104401  020121               TYPE    ,MSG4
2506  010260  013746  025774               MOV     PSERR,-(SP)     ;;SAVE PSERR FOR TYPEOUT
2507  010264  104405                       TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
2508  010266  104401  010336               TYPE    ,NULL           ;NULL CHAR
2509  010272  012737  000114  003124       MOV     #114.,MAXTRK    ;FOR ALL SUBSEQUENT PASSES
2510  010300  012737  000764  003126       MOV     #500.,MAXSK
2511
2512  010306  005737  003132               TST     COMP1           ;COMPAT PASS 1?
2513  010312  001407                       BEQ     1$              ;BR IF NO
2514  010314  104401  020350               TYPE    ,COMPAT         ;DONE MSG
2515  010320  000000                       HALT
2516  010322  005037  003132               CLR     COMP1
2517  010326  005237  003134               INC     COMP2
2518  010332  000137  003732       1$:     JMP     LOOP            ;REPEAT
2519
2520  010336      377     377     000  NULL:   .BYTE   -1,-1,0        ;NULL CHAR STRING
2521              010342                  .EVEN
2522
```

```
2523
2524                                    .SBTTL PROGRAM SUBROUTINES
2525
2526
2527  010342  012737  012142  000004  SIZE:   MOV     #INTSRV,ERRVEC  ;SETUP TIMEOUT TRAP ADDR
2528  010350  012737  000200  000006          MOV     #200,ERRVEC+2   ;LOCKOUT INTERR
2529  010356  005037  012150                  CLR     INTFLG
2530
2531  010362  012700  017776                  MOV     #17776,R0       ;SIZE MEM. START WITH 4K
2532  010366  005001                          CLR     R1              ;MEM MAP, 1=4K, 2=8K, ETC
2533  010370  005037  012150                  CLR     INTFLG
2534
2535  010374  005710                  1$:     TST     (R0)
2536  010376  005737  012150                  TST     INTFLG          ;GOT TIMEOUT INTERR?
2537  010402  001007                          BNE     2$              ;BR IF YES
2538  010404  005201                          INC     R1
2539  010406  020027  157776                  CMP     R0,#157776      ;DONE 28K MEM?
2540  010412  001403                          BEQ     2$
2541  010414  062700  020000                  ADD     #20000,R0       ;GO TO NEXT 4K
2542  010420  000765                          BR      1$
2543
2544  010422  010037  010766          2$:     MOV     R0,MAXMEM       ;SAVE
2545  010426  162737  020000  010766          SUB     #20000,MAXMEM   ;GO BACK TO LAST VALID BLOCK
2546  010434  006301                          ASL     R1              ;MULT BY 2
2547  010436  016137  010746  010446          MOV     MEMTBL-2(R1),3$
2548  010444  104401                          TYPE
2549  010446  000000                  3$:     .WORD   0               ;MEM SIZE
2550  010450  104401  017314                  TYPE    ,MEM
2551
2552  010454  005000                          CLR     R0
2553  010456  005037  012150                  CLR     INTFLG
2554  010462  012700  176500                  MOV     #TK1S,R0        ;SETUP CLUSTER CSR ADDR
2555  010466  005710                  4$:     TST     (R0)            ;DARE IT TO TIMEOUT
2556  010470  005737  012150                  TST     INTFLG          ;DID IT?
2557  010474  001011                          BNE     6$              ;BR IF YES
2558  010476  020027  176520                  CMP     R0,#TK3S        ;DID ALL 3?
2559  010502  001403                          BEQ     5$              ;BR IF YES
2560  010504  062700  000010                  ADD     #10,R0          ;ELSE DO NEXT
2561  010510  000766                          BR      4$
2562
2563  010512  005237  010742          5$:     INC     CLPRES          ;OPTION PRESENT
2564  010516  000406                          BR      7$              ;ALL 3 MUST RESPOND
2565
2566  010520  005037  010742          6$:     CLR     CLPRES          ;OPTION NOT PRESENT
2567  010524  104401  017102                  TYPE    ,CLOPT
2568  010530  104401  017334                  TYPE    ,NOTPRES
2569
2570  010534  012737  000006  000004  7$:     MOV     #ERRVEC+2,ERRVEC        ;RESET TMO VECTOR
2571  010542  005037  000006                  CLR     ERRVEC+2
```

```
2572
2573  010546  005000                        CLR     R0
2574  010550  012701  000200                MOV     #BIT7,R1
2575
2576  010554  030137  001246        8$:     BIT     R1,$DEVM        ;DEVICE DROPPED?
2577  010560  001006                        BNE     10$             ;BR IF YES
2578  010562  000241                9$:     CLC
2579  010564  006301                        ASL     R1
2580  010566  103424                        BCS     13$             ;BR IF ALL BITS TESTED
2581  010570  062700  000002                ADD     #2,R0           ;ELSE BUMP INDEX
2582  010574  000767                        BR      8$
2583
2584  010576  016037  010616  010606  10$:  MOV     12$(R0),11$
2585  010604  104401                        TYPE
2586  010606  000000                11$:    .WORD   0               ;DEVICE TO BE DROPPED
2587  010610  104401  017647                TYPE    ,DROP
2588  010614  000762                        BR      9$
2589
2590  010616  017244                12$:    CLOCK
2591  010620  017232                        DRV0
2592  010622  017237                        DRV1
2593  010624  017155                        SCOM
2594  010626  017170                        ACOM
2595  010630  017122                        CLT1
2596  010632  017133                        CLT2
2597  010634  017144                        CLT3
2598  010636  017204                        PRINT
2599
2600  010640  005037  010744        13$:    CLR     PRPRES
2601  010644  005037  010746                CLR     PRPORT
2602  010650  005737  177514                TST     PRS             ;SEE IF PRINTER ERR
2603  010654  100027                        BPL     16$             ;BR IF NO
2604
2605  010656  032737  000010  001246        BIT     #BIT3,$DEVM     ;COMM PORT IN EXT LOOPBACK?
2606  010664  001405                        BEQ     15$             ;BR IF YES
2607
2608  010666  104401  017204        14$:    TYPE    ,PRINT
2609  010672  104401  017334                TYPE    ,NOTPRES
2610  010676  000420                        BR      17$
2611
2612  010700  012737  037034  177420  15$:  MOV     #<AMOD!B9600!OPAP.CHAR8>,PARAM
2613  010706  042737  000006  176610        BIC     #<RTS.DTR>,AMRC ;MUST BE TOGGLED
2614  010714  052737  000006  176610        BIS     #<RTS.DTR>,AMRC
2615  010722  005737  177514                TST     PRS             ;NOW ANY ERRORS?
2616  010726  100757                        BMI     14$             ;BR IF YES
2617  010730  005237  010746                INC     PRPORT
2618  010734  005237  010744        16$.    INC     PRPRES          ;PRINTER PRESENT
2619
2620  010740  000207                17$:    RTS     PC
```

```
2621
2622  010742  000000                CLPRES: 0                       ;CLUSTER PRESENT = NON-ZERO
2623  010744  000000                PRPRES: 0                       ;PRINTER PRESENT = NON-ZERO
2624  010746  000000                PRPORT: 0                       ;1 = PRINTER ENABLED THRU LOOPBACK
2625
2626  010750  017253                MEMTBL: M4K                     ;MSG ADDRESSES
2627  010752  017257                        M8K
2628  010754  017263                        M12K
2629  010756  017270                        M16K
2630  010760  017275                        M20K
2631  010762  017302                        M24K
2632  010764  017307                        M30K
2633
2634  010766  000000                MAXMEM: 0                       ;MAX MEMORY
2635
2636
2637                                ;*****************************************************************
2638                                ;       RANDOM NUMBER GENERATOR
2639                                ;*****************************************************************
2640
2641  010770  013700  011100        RAND:   MOV     LONUM,R0
2642  010774  013701  011076                MOV     HINUM,R1
2643  011000  012702  177771                MOV     #-7,R2
2644  011004  006300                1$:     ASL     R0
2645  011006  006101                        ROL     R1              ;ROTATE CARRY TO R1
2646  011010  005202                        INC     R2              ;DONE?
2647  011012  001374                        BNE     1$              ;BR IN NO
2648  011014  063700  011100                ADD     LONUM,R0        ;ADD NUMBER TO MAKE X 129
2649  011020  005501                        ADC     R1
2650  011022  063701  011076                ADD     HINUM,R1        ;ADD NUMBER TO MAKE X 129
2651  011026  062700  001057                ADD     #1057,R0        ;ADD LO CONSTANT
2652  011032  005501                        ADC     R1
2653  011034  062701  047401                ADD     #47401,R1       ;ADD HI CONSTANT
2654  011040  010037  011100                MOV     R0,LONUM
2655  011044  010137  011076                MOV     R1,HINUM
2656                                                                ;SCALE TO BE WITHIN LIMITS
2657  011050  042701  177400                BIC     #177400,R1      ;SAVE LO BYTE ONLY
2658  011054  020137  011072                CMP     R1,LOLIM
2659  011060  103743                        BLO     RAND            ;BR IF N < LOLIM
2660  011062  020137  011074                CMP     R1,HILIM
2661  011066  101340                        BHI     RAND            ;BR IF N > HILIM
2662  011070  000207                        RTS     PC              ;ELSE EXIT WITH R1   N
2663
2664  011072  000000                LOLIM:  0
2665  011074  000000                HILIM:  0
2666
2667  011076  176543                HINUM:  .WORD 176543
2668  011100  123456                LONUM:  .WORD 123456
2669
2670
2671  011102  123727  001210  000001  EXAMKB: CMPB    $ENV,#1         ;APT?
2672  011110  001401                        BEQ     1$              ;BR IF YES
2673  011112  104407                        CKSWR
2674  011114  000207                1$:     RTS     PC
```

```
2675
2676                                    ;;*****************************************************************
2677                                    ;ROUTINE TO DISPLAY CURRENT DEVM & ALLOW OPERATOR TO INPUT NEW VALUE.
2678                                    ;;*****************************************************************
2679
2680  011116  104401  020445          GT$DEV: TYPE    ,DEVM           ;ELSE SHOW CURRENT
2681  011122  013746  001246                  MOV     $DEVM,-(SP)     ;;SAVE $DEVM FOR TYPEOUT
2682  011126  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2683  011130  104401  025136                  TYPE    ,$MNEW          ;GET NEW
2684
2685  011134  005046                  1$:     CLR     -(SP)           ;CLR CTR
2686  011136  005046                          CLR     -(SP)           ;CLR NEW DEVM
2687
2688  011140  105777  170000          2$:     TSTB    @$TKS           ;CHAR THERE?
2689  011144  100375                          BPL     2$              ;BR IF NO
2690  011146  117746  167774                  MOVB    @$TKB,-(SP)     ;ELSE GET CHAR
2691  011152  042716  177600                  BIC     #^C177,(SP)     ;MAKE IT 7 BIT ASCII
2692
2693  011156  021627  000015          3$:     CMP     (SP),#15        ;<CR>?
2694  011162  001013                          BNE     7$              ;BR IF NO
2695  011164  005766  000004                  TST     4(SP)           ;ELSE IS IT 1'ST CHAR?
2696  011170  001403                          BEQ     4$              ;BR IF YES
2697  011172  016637  000002  001246          MOV     2(SP),$DEVM     ;ELSE SAVE NEW DEVM
2698  011200  062706  000006          4$:     ADD     #6,SP           ;RESTORE STACK
2699  011204  104401  001165          5$:     TYPE    ,$CRLF
2700  011210  000002                  6$:     RTI
2701
2702  011212  004737  024236          7$:     JSR     PC,$TYPEC       ;ECHO CHAR
2703  011216  021627  000060                  CMP     (SP),#60        ;CHAR < 0?
2704  011222  002420                          BLT     9$              ;BR IF YES
2705  011224  021627  000067                  CMP     (SP),#67        ;CHAR >7?
2706  011230  003015                          BGT     9$              ;BR IF YES
2707  011232  042726  000060                  BIC     #60,(SP)+       ;STRIP OFF ASCII
2708  011236  005766  000002                  TST     2(SP)           ;IS IT 1'ST CHAR?
2709  011242  001403                          BEQ     8$              ;BR IF YES
2710  011244  006316                          ASL     (SP)            ;ELSE SHIFT PRESENT CHAR
2711  011246  006316                          ASL     (SP)            ;TO MAKE ROOM
2712  011250  006316                          ASL     (SP)            ;FOR NEW ONE
2713
2714  011252  005266  000002          8$:     INC     2(SP)           ;KEEP CHAR CT
2715  011256  056616  177776                  BIS     -2(SP),(SP)     ;SET IN NEW CHAR
2716  011262  000726                          BR      2$              ;GET NEW ONE
2717
2718  011264  104401  001164          9$:     TYPE    ,$QUES          ;TYPE ?<CRLF>
2719  011270  000747                          BR      6$              ;DO ALL OVER
2720
```

```
2721                                    ;;*****************************************************************
2722                                    ;* ROUTINE TO SET 2 CONSECUTIVE INTERRUPT VECTORS
2723                                    ;* 3 ARGUMENTS ARE PASSED THRU R5:
2724                                    ;*      VECTOR ADDRESS
2725                                    ;*      INTERRUPT SERVICE ROUTINE ADDRESS FOR RECVR
2726                                    ;*      INTERRUPT SERVICE ROUTINE ADDRESS FOR XMITTER.
2727                                    ;* PRIORITY WILL BE SET TO DISABLE FURTHUR INTERR.
2728                                    ;;*****************************************************************
2729
2730  011272  010046                    SETVEC: MOV     R0,-(SP)                ;SAVE
2731  011274  012500                            MOV     (R5)+,R0                ;GET VECTOR ADDR
2732  011276  012520                            MOV     (R5)+,(R0)+             ;PUT SERV ROUTINE ADDR THERE
2733  011300  012720  000200                    MOV     #200,(R0)+              ;DISABLE INTERR
2734  011304  012520                            MOV     (R5)+,(R0)+             ;GET XMIT SERV ADDR NEXT
2735  011306  012710  000200                    MOV     #200,(R0)
2736  011312  012600                            MOV     (SP)+,R0                ;RESTORE
2737  011314  000205                            RTS     R5
2738
2739
2740
2741
2742
2743                                    ;;*****************************************************************
2744                                    ;WATCHDOG TIMER TO PREVENT DEVICES FROM ENTERING AN ENDLESS WAIT LOOP
2745                                    ;RETURN    IF TIMED OUT
2746                                    ;RETURN +4 IF NOT TIMED OUT & JUMP OVER ERROR
2747                                    ;;*****************************************************************
2748
2749  011316  012537  011376            TIMER:  MOV     (R5)+,FLGHLD    ;GET FLAG
2750  011322  012537  011400                    MOV     (R5)+,CTHLD     ;GET COUNT THAT FLAG SHOULD GO TO
2751
2752  011326  012701  000010                    MOV     #10,R1
2753  011332  012700  177777            4$:     MOV     #-1,R0
2754  011336  027737  000034  011400    1$:     CMP     @FLGHLD,CTHLD   ;RESPONDING?
2755  011344  001411                            BEQ     2$              ;BR IF YES
2756  011346  004737  011102                    JSR     PC,EXAMKB
2757  011352  005300                            DEC     R0              ;ELSE DEC COUNTER
2758  011354  001370                            BNE     1$              ;BR IF NOT TIMED OUT
2759  011356  005237  001200                    INC     $DEVCT          ;FOR APT
2760  011362  005301                            DEC     R1
2761  011364  001362                            BNE     4$
2762  011366  000402                            BR      .+6             ;ELSE TIMED OUT
2763
2764  011370  062705  000004            2$:     ADD     #4,R5           ;JUMP OVER ERROR ON RETURN
2765  011374  000205                            RTS     R5
2766
2767  011376  000000            FLGHLD: 0                               ;FLAG
2768  011400  000000            CTHLD:  0                               ;COUNT THAT FLAG MUST REACH
2769
```

```
2770                                    ;;*****************************************************************
2771                                    ;WATCHDOG TIMER TO PREVENT DX0 FROM ENTERING AN ENDLESS WAIT LOOP
2772                                    ;RETURN     IF TIMED OUT
2773                                    ;RETURN +4 IF NOT TIMED OUT & JUMP OVER ERROR
2774                                    ;;*****************************************************************
2775
2776  011402  005037  011474           TIMDO:  CLR     DOTMO
2777  011406  012737  000010  011472           MOV     #10,CTREG1
2778  011414  012737  177777  011470   4$:     MOV     #-1,CTREG
2779  011422  005737  003122           1$:     TST     FIN             ;FINISHED?
2780  011426  001015                           BNE     2$              ;BR IF YES
2781  011430  004737  011102                   JSR     PC,EXAMKB
2782  011434  005337  011470                   DEC     CTREG           ;ELSE DEC CTR
2783  011440  001370                           BNE     1$              ;BR IF NOT TIMED OUT
2784  011442  005237  001200                   INC     $DEVCT          ;FOR APT
2785  011446  005337  011472                   DEC     CTREG1
2786  011452  001360                           BNE     4$
2787  011454  005237  011474                   INC     DOTMO           ;SET FLAG
2788  011460  000402                           BR      .+6             ;TIMED OUT
2789
2790  011462  062716  000004           2$:     ADD     #4,(SP)         ;BUMP RET
2791  011466  000207                           RTS     PC
2792
2793  011470  000000                   CTREG:  0
2794  011472  000000                   CTREG1: 0
2795  011474  000000                   DOTMO:  0                       ;1 = TIMEOUT OCCURED
2796
2797
2798                                    ;;*****************************************************************
2799                                    ;WATCHDOG TIMER TO PREVENT DX1 FROM ENTERING AN ENDLESS WAIT LOOP
2800                                    ;RETURN     IF TIMED OUT
2801                                    ;RETURN +4 IF NOT TIMED OUT & JUMP OVER ERROR
2802                                    ;;*****************************************************************
2803
2804  011476  005037  011564           TIMD1:  CLR     D1TMO
2805  011502  012737  000010  011472           MOV     #10,CTREG1
2806  011510  012737  177777  011470   4$:     MOV     #-1,CTREG
2807  011516  005737  003122           1$:     TST     FIN             ;FINISHED?
2808  011522  001015                           BNE     2$              ;BR IF YES
2809  011524  004737  011102                   JSR     PC,EXAMKB
2810  011530  005337  011470                   DEC     CTREG           ;ELSE DEC CTR
2811  011534  001370                           BNE     1$              ;BR IF NOT TIMED OUT
2812  011536  005237  001200                   INC     $DEVCT          ;FOR APT
2813  011542  005337  011472                   DEC     CTREG1
2814  011546  001360                           BNE     4$
2815  011550  005237  011564                   INC     D1TMO           ;SET FLAG
2816  011554  000402                           BR      .+6             ;TIMED OUT
2817
2818  011556  062716  000004           2$:     ADD     #4,(SP)         ;BUMP RET
2819  011562  000207                           RTS     PC
2820
2821  011564  000000                   D1TMO:  0                       ;1 = TIMEOUT OCCURED
2822
```

```
2823                            ;;*****************************************************************
2824                            ;       GENERAL TIMER
2825                            ;;*****************************************************************
2826
2827  011566  012700  177777   TIMER1: MOV     #-1,R0
2828  011572  005300                   DEC     R0
2829  011574  001376                   BNE     .-2
2830  011576  000207                   RTS     PC
2831
2832                            ;;*****************************************************************
2833                            ;WATCHDOG TIMER TO PREVENT DEVICES FROM ENTERING AN ENDLESS WAIT LOOP
2834                            ;RETURN     IF TIMED OUT
2835                            ;RETURN +4 IF NOT TIMED OUT & JUMP OVER ERROR
2836                            ;;*****************************************************************
2837
2838  011600  012537  011660   TIMER2: MOV     (R5)+,REGHLD    ;GET REG
2839  011604  012537  011662           MOV     (R5)+,BITHLD    ;GET BIT TO BE TESTED
2840
2841  011610  012701  000010           MOV     #10,R1
2842  011614  012700  177777   4$:     MOV     #-1,R0
2843  011620  033777  011662  000032 1$:      BIT     BITHLD,@REGHLD  ;BIT THERE?
2844  011626  001011                   BNE     2$              ;BR IF YES
2845  011630  004737  011102           JSR     PC,EXAMKB
2846  011634  005300                   DEC     R0              ;ELSE DEC COUNTER
2847  011636  001370                   BNE     1$              ;BR IF NOT TIMED OUT
2848  011640  005237  001200           INC     $DEVCT          ;FOR APT
2849  011644  005301                   DEC     R1
2850  011646  001362                   BNE     4$
2851  011650  000402                   BR      .+6             ;ELSE TIMED OUT
2852
2853  011652  062705  000004   2$:     ADD     #4,R5           ;JUMP OVER ERROR ON RETURN
2854  011656  000205                   RTS     R5
2855
2856  011660  000000           REGHLD: 0                       ;DEVICE REG
2857  011662  000000           BITHLD: 0                       ;DEV REG BIT TO BE TESTED
2858
2859
2860                            ;;*****************************************************************
2861                            ;ROUTINE TO CLEAR BOTH BAD TRACK/SECTOR TABLES
2862                            ;;*****************************************************************
2863
2864  011664  012700  002624   CLRBAD: MOV     #D0BAD,R0
2865  011670  005020           1$:     CLR     (R0)+
2866  011672  020027  002650           CMP     R0,#D0BAD+20.
2867  011676  001374                   BNE     1$
2868
2869  011700  012700  003074           MOV     #D1BAD,R0
2870  011704  005020           2$:     CLR     (R0)+
2871  011706  020027  003120           CMP     R0,#D1BAD+20.
2872  011712  001374                   BNE     2$
2873
2874  011714  000207                   RTS     PC
2875
```

```
2876                                    ;;*****************************************************************
2877                                    ;LOAD DX0 BAD TRACK/SECTOR TABLE
2878                                    ;;*****************************************************************
2879
2880  011716  010046                   LDOBAD: MOV     R0,-(SP)        ;SAVE
2881
2882  011720  012700  002624                   MOV     #DOBAD,R0
2883  011724  005710                   1$:     TST     (R0)            ;ENTRY PRESENT?
2884  011726  001003                           BNE     2$              ;BR IF YES
2885  011730  013710  177174                   MOV     RXSA,(R0)       ;ELSE STORE BAD RXSA IN TABLE
2886  011734  000405                           BR      3$              ;EXIT
2887
2888  011736  005720                   2$:     TST     (R0)+           ;BUMP PTR
2889  011740  020027  002650                   CMP     R0,#DOBAD+20.   ;AT END OF TABLE?
2890  011744  001367                           BNE     1$              ;BR IF NO
2891  011746  104104                           ERROR   104             ;10 BAD SECTORS...REPLACE
2892  011750  012600                   3$:     MOV     (SP)+,R0        ;RESTORE
2893  011752  000207                           RTS     PC
2894
2895
2896
2897
2898
2899                                    ;;*****************************************************************
2900                                    ;CHECK DX0 BAD TRACK/SECTOR TABLE BEFORE DOING RANDOM SEEKS.
2901                                    ;IF TRK/SEC IN TABLE, RETURN TO RE-CALCULATE NEW TRACK & SECTOR.
2902                                    ;;*****************************************************************
2903
2904  011754  010046                   CKOBAD: MOV     R0,-(SP)        ;SAVE
2905  011756  010146                           MOV     R1,-(SP)
2906
2907  011760  013701  002406                   MOV     DOTRK,R1
2908  011764  000301                           SWAB    R1
2909  011766  053701  002410                   BIS     DOSEC,R1
2910
2911  011772  012700  002624                   MOV     #DOBAD,R0
2912  011776  020027  002650           1$:     CMP     R0,#DOBAD+20.   ;END OF TABLE?
2913  012002  001405                           BEQ     2$              ;BR IF YES
2914  012004  005710                           TST     (R0)            ;ELSE ANY ENTRY?
2915  012006  001403                           BEQ     2$              ;BR IF NO
2916  012010  022001                           CMP     (R0)+,R1        ;ELSE COMPARE?
2917  012012  001371                           BNE     1$              ;BR IF NO
2918  012014  000402                           BR      3$              ;ELSE DONT BUMP RET ADDR
2919
2920  012016  062705  000002           2$:     ADD     #2,R5           ;BUMP RET ADDR
2921  012022  012601                   3$:     MOV     (SP)+,R1        ;RESTORE
2922  012024  012600                           MOV     (SP)+,R0
2923  012026  000205                           RTS     R5
```

```
                                ;;*****************************************************************
                                ;LOAD DX1 BAD TRACK/SECTOR TABLE
                                ;;*****************************************************************

012030  010046                  LD1BAD: MOV     R0,-(SP)        ;SAVE

012032  012700  003074                  MOV     #D1BAD,R0
012036  005710                  1$:     TST     (R0)            ;ENTRY PRESENT?
012040  001003                          BNE     2$              ;BR IF YES
012042  013710  177174                  MOV     RXSA,(R0)       ;ELSE STORE BAD RXSA IN TABLE
012046  000405                          BR      3$              ;EXIT

012050  005720                  2$:     TST     (R0)+           ;BUMP PTR
012052  020027  003120                  CMP     R0,#D1BAD+20.   ;AT END OF TABLE?
012056  001367                          BNE     1$              ;BR IF NO
012060  104105                          ERROR   105             ;10 BAD SECTORS...REPLACE
012062  012600                  3$:     MOV     (SP)+,R0        ;RESTORE
012064  000207                          RTS     PC



                                ;;*****************************************************************
                                ;CHECK DX1 BAD TRACK/SECTOR TABLE BEFORE DOING RANDOM SEEKS.
                                ;IF TRK/SEC IN TABLE, RETURN TO RE-CALCULATE NEW TRACK & SECTOR.
                                ;;*****************************************************************

012066  010046                  CK1BAD: MOV     R0,-(SP)        ;SAVE
012070  010146                          MOV     R1,-(SP)

012072  013701  002656                  MOV     D1TRK,R1
012076  000301                          SWAB    R1
012100  053701  002660                  BIS     D1SEC,R1

012104  012700  003074                  MOV     #D1BAD,R0
012110  020027  003120          1$:     CMP     R0,#D1BAD+20.   ;END OF TABLE?
012114  001405                          BEQ     2$              ;BR IF YES
012116  005710                          TST     (R0)            ;ELSE ANY ENTRY?
012120  001403                          BEQ     2$              ;BR IF NO
012122  022001                          CMP     (R0)+,R1        ;ELSE COMPARE?
012124  001371                          BNE     1$              ;BR IF NO
012126  000402                          BR      3$              ;ELSE DONT BUMP RET ADDR

012130  062705  000002          2$:     ADD     #2,R5           ;BUMP RET ADDR
012134  012601                  3$:     MOV     (SP)+,R1        ;RESTORE
012136  012600                          MOV     (SP)+,R0
012140  000205                          RTS     R5
```

```
2973                                    .SBTTL  INTERRUPT HANDLERS
2974                                    ;;*****************************************************************
2975                                    ; GENERAL SERVICE ROUTINE TO BUMP 'INTFLG'
2976                                    ; CALLING ROUTINE WILL KNOW WHAT TO DO
2977                                    ;;*****************************************************************
2978
2979  012142  005237  012150           INTSRV: INC     INTFLG
2980  012146  000002                           RTI
2981  012150  000000                   INTFLG: 0
2982
2983
2984                                    ;;*****************************************************************
2985                                    ;PRINTER  INTERRUPT HANDLER: VALUES FROM 40 THRU 176.
2986                                    ;;*****************************************************************
2987
2988  012152  013737  177514  012302   PRSRV:  MOV     PRS,SPRS                ;STORE
2989  012160  005737  012302                    TST     SPRS            ;ERROR?
2990  012164  100002                           BPL     1$              ;BR IF NO
2991  012166  104007                           ERROR   7               ;PRINTER STATUS ERROR
2992  012170  000002                           RTI
2993
2994  012172  013737  012276  177516   1$:     MOV     PRCHR,PRB       ;ELSE PRINT CHAR
2995  012200  023727  012276  000015           CMP     PRCHR,#CR       ;JUST DID CR?
2996  012206  001413                           BEQ     2$              ;BR IF YES
2997  012210  023727  012276  000012           CMP     PRCHR,#LF       ;JUST DID LF?
2998  012216  001413                           BEQ     3$              ;BR IF YES
2999  012220  023727  012276  000176           CMP     PRCHR,#176      ;JUST DID LAST CHAR?
3000  012226  001413                           BEQ     4$              ;BR IF YES
3001  012230  005237  012276                    INC     PRCHR           ;ELSE BUMP CHAR FOR NEXT INTERRUPT
3002  012234  000417                           BR      5$              ;EXIT
3003
3004  012236  012737  000012  012276   2$:     MOV     #LF,PRCHR       ;DO LF NEXT
3005  012244  000413                           BR      5$
3006  012246  012737  000040  012276   3$:     MOV     #40,PRCHR       ;DO SPACE NEXT
3007  012254  000407                           BR      5$
3008  012256  012737  000015  012276   4$:     MOV     #CR,PRCHR       ;DO CR NEXT & START OVER
3009  012264  005237  012300                    INC     LINCT
3010  012270  005237  001200                    INC     $DEVCT          ;FOR APT
3011  012274  000002                   5$:     RTI
3012
3013  012276  000000                   PRCHR:  0                       ;CHAR TO BE PRINTED
3014  012300  000000                   LINCT:  0                       ;LINE CTR
3015  012302  000000                   SPRS:   0                       ;STORE PRINTER CSR
```

```
3016
3017                                    ;;*****************************************************************
3018                                    ;CLUSTER TERMINAL #1 INTERRUPT HANDLER: LOOP VALUES 0 THRU 377.
3019                                    ;;*****************************************************************
3020
3021  012304  042737  000100  176504  TP1SRV: BIC     #IE,TP1S        ;DISABLE INTER, RECV'R WILL TURN BACK ON
3022  012312  013737  012400  176506           MOV     T1CHR,TP1B      ;PRINT CHAR
3023  012320  000002                           RTI
3024
3025  012322  013737  176502  012402  TK1SRV: MOV     TK1B,T1HLD      ;STORE CHAR
3026  012330  023737  012402  012400           CMP     T1HLD,T1CHR     ;OK?
3027  012336  001401                           BEQ     1$              ;BR IF YES
3028  012340  104010                           ERROR   10              ;CHAR COMP ERROR
3029
3030  012342  023727  012400  000377  1$:      CMP     T1CHR,#377      ;LAST CHAR?
3031  012350  001403                           BEQ     2$              ;BR IF YES
3032  012352  005237  012400                   INC     T1CHR           ;ELSE BUMP
3033  012356  000404                           BR      3$
3034
3035  012360  005037  012400          2$:      CLR     T1CHR           ;START OVER
3036  012364  005237  001200                   INC     $DEVCT          ;FOR APT
3037  012370  052737  000100  176504  3$:      BIS     #IE,TP1S        ;ALLOW PRINTER INTERR
3038  012376  000002                           RTI
3039
3040  012400  000000                  T1CHR:   0                       ;CHAR TO XMITT
3041  012402  000000                  T1HLD:   0                       ;REC'D CHAR
3042
3043
3044                                    ;;*****************************************************************
3045                                    ;CLUSTER TERMINAL #2 INTERRUPT HANDLER: LOOP VALUES 0 THRU 377.
3046                                    ;;*****************************************************************
3047
3048  012404  042737  000100  176514  TP2SRV: BIC     #IE,TP2S        ;DISABLE INTER, RECV'R WILL TURN BACK ON
3049  012412  013737  012500  176516           MOV     T2CHR,TP2B      ;PRINT CHAR
3050  012420  000002                           RTI
3051
3052  012422  013737  176512  012502  TK2SRV: MOV     TK2B,T2HLD      ;STORE CHAR
3053  012430  023737  012502  012500           CMP     T2HLD,T2CHR     ;OK?
3054  012436  001401                           BEQ     1$              ;BR IF YES
3055  012440  104011                           ERROR   11              ;CHAR COMP ERROR
3056
3057  012442  023727  012500  000377  1$:      CMP     T2CHR,#377      ;LAST CHAR?
3058  012450  001403                           BEQ     2$              ;BR IF YES
3059  012452  005237  012500                   INC     T2CHR           ;ELSE BUMP
3060  012456  000404                           BR      3$
3061
3062  012460  005037  012500          2$:      CLR     T2CHR           ;START OVER
3063  012464  005237  001200                   INC     $DEVCT          ;FOR APT
3064  012470  052737  000100  176514  3$:      BIS     #IE,TP2S        ;ALLOW PRINTER INTERR
3065  012476  000002                           RTI
3066
3067  012500  000000                  T2CHR:   0                       ;CHAR TO XMITT
3068  012502  000000                  T2HLD:   0                       ;REC'D CHAR
```

```
3069
3070                                    ;;*****************************************************************
3071                                    ;CLUSTER TERMINAL #3 INTERRUPT HANDLER: LOOP VALUES 0 THRU 377.
3072                                    ;;*****************************************************************
3073
3074  012504  042737  000100  176524  TP3SRV: BIC     #IE,TP3S        ;DISABLE INTER, RECV'R WILL TURN BACK ON
3075  012512  013737  012600  176526           MOV     T3CHR,TP3B      ;PRINT CHAR
3076  012520  000002                          RTI
3077
3078  012522  013737  176522  012602  TK3SRV: MOV     TK3B,T3HLD      ;STORE CHAR
3079  012530  023737  012602  012600           CMP     T3HLD,T3CHR     ;OK?
3080  012536  001401                          BEQ     1$              ;BR IF YES
3081  012540  104012                          ERROR   12              ;CHAR COMP ERROR
3082
3083  012542  023727  012600  000377  1$:     CMP     T3CHR,#377      ;LAST CHAR?
3084  012550  001403                          BEQ     2$              ;BR IF YES
3085  012552  005237  012600                  INC     T3CHR           ;ELSE BUMP
3086  012556  000404                          BR      3$
3087
3088  012560  005037  012600          2$:     CLR     T3CHR           ;START OVER
3089  012564  005237  001200                  INC     $DEVCT          ;FOR APT
3090  012570  052737  000100  176524  3$:     BIS     #IE,TP3S        ;ALLOW PRINTER INTERR
3091  012576  000002                          RTI
3092
3093  012600  000000          T3CHR:  0               ;CHAR TO XMITT
3094  012602  000000          T3HLD:  0               ;REC'D CHAR
3095
3096
3097                                    ;;*****************************************************************
3098                                    ;ASYNC COMM PORT INTERRUPT HANDLER: LOOP VALUES 0 THRU 377.
3099                                    ;;*****************************************************************
3100
3101  012604  042737  000100  176614  AMXSRV: BIC     #IE,AMXC        ;DISABLE INTER, RECV'R WILL TURN BACK ON
3102  012612  013737  012700  176616           MOV     COMCHR,AMXB     ;XMITT CHAR
3103  012620  000002                          RTI
3104
3105  012622  013737  176612  012702  AMRSRV: MOV     AMRB,COMHLD     ;STORE CHAR
3106  012630  023737  012702  012700           CMP     COMHLD,COMCHR   ;OK?
3107  012636  001401                          BEQ     1$              ;BR IF YES
3108  012640  104013                          ERROR   13              ;CHAR COMPARE ERROR
3109
3110  012642  023727  012700  000377  1$:     CMP     COMCHR,#377     ;LAST CHAR?
3111  012650  001403                          BEQ     2$              ;BR IF YES
3112  012652  005237  012700                  INC     COMCHR          ;ELSE BUMP
3113  012656  000404                          BR      3$
3114
3115  012660  005037  012700          2$:     CLR     COMCHR          ;START OVER
3116  012664  005237  001200                  INC     $DEVCT          ;FOR APT
3117  012670  052737  000100  176614  3$:     BIS     #IE,AMXC        ;ALLOW XMIT INTERR
3118  012676  000002                          RTI
3119
3120  012700  000000          COMCHR: 0               ;XMIT CHAR
3121  012702  000000          COMHLD: 0               ;REC'D CHAR
```

```
3122
3123                                    ;;*****************************************************************
3124                                    ;SYNC COMM PORT INTERRUPT HANDLER: LOOP VALUES 27 THRU 377.
3125                                    ;;*****************************************************************
3126
3127  012704  042737  000100  176624  SMXSRV: BIC     #IE,SMXC        ;DISABLE INTER, RECV'R WILL TURN BACK ON
3128  012712  013737  012700  176626          MOV     COMCHR,SMXB     ;XMITT CHAR
3129  012720  023727  012700  000377          CMP     COMCHR,#377     ;LAST CHAR?
3130  012726  001020                          BNE     2$              ;BR IF NO
3131
3132  012730  005737  012772                  TST     LSTCHR
3133  012734  001010                          BNE     1$
3134  012736  005237  012772                  INC     LSTCHR
3135  012742  052737  000100  176624          BIS     #IE,SMXC        ;ALLOW RDY INTR SO CAN SHUT DOWN FAST
3136  012750  005237  001200                  INC     $DEVCT          ;FOR APT
3137  012754  000002                          RTI
3138
3139  012756  042737  000020  176624  1$:     BIC     #SEND,SMXC      ;NO MORE TO SEND
3140  012764  005237  001200                  INC     $DEVCT          ;FOR APT
3141  012770  000002                  2$:     RTI
3142
3143  012772  000000                  LSTCHR: 0                       ;SET WHEN LAST CHAR XMIT
3144
3145
3146
3147
3148
3149  012774  013737  176622  012702  SMRSRV: MOV     SMRB,COMHLD     ;STORE CHAR
3150  013002  123727  012702  000026          CMPB    COMHLD,#026     ;IGNORE SYNC CHARS
3151  013010  001430                          BEQ     4$
3152  013012  023737  012702  012700          CMP     COMHLD,COMCHR   ;OK?
3153  013020  001401                          BEQ     1$              ;BR IF YES
3154  013022  104014                          ERROR   14              ;CHAR COMPARE ERROR
3155
3156  013024  023727  012700  000377  1$:     CMP     COMCHR,#377     ;LAST CHAR?
3157  013032  001403                          BEQ     2$              ;BR IF YES
3158  013034  005237  012700                  INC     COMCHR          ;ELSE BUMP
3159  013040  000411                          BR      3$
3160
3161  013042  012737  177777  012700  2$:     MOV     #-1,COMCHR      ;INDICATE DONE
3162  013050  042737  000100  176620          BIC     #IE,SMRC        ;ALL DONE
3163  013056  005237  001200                  INC     $DEVCT          ;FOR APT
3164  013062  000403                          BR      4$
3165
3166  013064  052737  000100  176624  3$:     BIS     #IE,SMXC        ;ALLOW XMIT INTERR
3167  013072  000002                  4$:     RTI
3168
```

```
3169
3170                                    ;;*****************************************************************
3171                                    ;                   DISK INTERRUPT HANDLERS
3172                                    ;ALL DISK INTERRUPTS ENTER THRU RXSRV & DISPATCH TO THE
3173                                    ;CURRENT SERVICE ROUTINE POINTED TO BY:
3174                                    ;                   D0DISP FOR DX0 INTERRUPTS
3175                                    ;                   D1DISP FOR DX1 INTERRUPTS
3176                                    ;WILL GENERALLY BE IN THE ORDER OF SERVICE ROUTINES BELOW.
3177                                    ;;*****************************************************************
3178
3179  013074  105737  177170           RXSRV:  TSTB    RXCS            ;CONTR RDY?
3180  013100  100413                           BMI     1$              ;BR IF YES
3181  013102  013737  177170  013204           MOV     RXCS,SRXCS      ;ELSE SAVE FOR ERROR TYPEOUTS
3182  013110  013737  177172  013206           MOV     RXES,SRXES
3183  013116  013737  177174  013210           MOV     RXSA,SRXSA
3184  013124  104015                           ERROR   15              ;UNEXPECTED INTERRUPT
3185  013126  000002                           RTI                     ;GO BACK
3186
3187  013130  005737  177170           1$:     TST     RXCS            ;ERROR?
3188  013134  100011                           BPL     2$              ;BR IF NO
3189
3190  013136  013737  177170  013204           MOV     RXCS,SRXCS      ;ELSE SAVE FOR ERROR TYPEOUTS
3191  013144  013737  177172  013206           MOV     RXES,SRXES
3192  013152  013737  177174  013210           MOV     RXSA,SRXSA
3193
3194  013160  032737  000020  177170  2$:     BIT     #USEL,RXCS      ;CHECK UNIT SELECT BIT
3195  013166  001003                           BNE     D1INT           ;BR IF INTER FROM DX1
3196
3197  013170  000177  000000                   JMP     @D0DISP         ;ELSE DISPATCH TO DX0 SERVICE ROUTINE
3198  013174  000000           D0DISP: 0
3199
3200  013176  000177  000000           D1INT:  JMP     @D1DISP         ;DX1 INTERRUPT DISPATCH
3201  013202  000000           D1DISP: 0
3202
3203
3204  013204  000000           SRXCS:  0               ;SAVE RXCS
3205  013206  000000           SRXES:  0               ;SAVE RXES
3206  013210  000000           SRXSA:  0               ;SAVE RXSA
3207
```

```
3208
3209                                    ;;*****************************************************************
3210                                    ; HANDLER TO FILL THE DRIVE BUFFER FOR DX0
3211                                    ;& POINT TO THE WRITE SECTOR HANDLER.
3212                                    ;;*****************************************************************
3213
3214  013212  012703  000100           DOFBUF: MOV     #100,R3          ;WORD CT
3215  013216  013701  002406                   MOV     DOTRK,R1
3216  013222  000301                           SWAB    R1
3217  013224  053701  002410                   BIS     DOSEC,R1
3218  013230  012737  013256  013174           MOV     #DOWSEC,DODISP   ;POINT TO WRITE SECTOR AFTER RTI
3219  013236  012737  000101  177170           MOV     #<FBUF!IE>,RXCS ;ISSUE FILL BUFFER CMD
3220  013244  010137  177172           1$:     MOV     R1,RXDB          ;FILL ENTIRE SECTOR WITH ITS ADDRESS
3221  013250  005303                           DEC     R3
3222  013252  001374                           BNE     1$
3223  013254  000002                           RTI
3224
3225
3226
3227
3228
3229                                    ;;*****************************************************************
3230                                    ;HANDLER TO WRITE THE SECTOR FOR DX0 & POINT TO WRITE CHECK.
3231                                    ;;*****************************************************************
3232
3233  013256  013701  002406           DOWSEC: MOV     DOTRK,R1
3234  013262  000301                           SWAB    R1
3235  013264  053701  002410                   BIS     DOSEC,R1
3236  013270  010137  177174                   MOV     R1,RXSA ;LOAD TRACK & SECTOR ADDR
3237  013274  012737  013332  013174           MOV     #DOWCHK,DODISP  ;POINT TO WRITE CHECK HANDLER
3238  013302  023727  003120  000005           CMP     DXTST,#5
3239  013310  001004                           BNE     1$
3240  013312  012737  000115  177170           MOV     #<WDDSEC!IE>,RXCS       ;ONLY FOR WRITE DELETED DATA TEST
3241  013320  000403                           BR      2$
3242  013322  012737  000105  177170    1$:    MOV     #<WSEC!IE>,RXCS ;ISSUE WRITE SECTOR COMMAND
3243  013330  000002                    2$:    RTI
3244
```

```
                                ;;*****************************************************************
                                ;HANDLER TO WRITE CHECK ON DX0.
                                ;WILL GO TO FILL BUFFER TO DO SAME SECTOR ON SOFT ERROR
                                ;OR NEXT SECTOR ON NO ERROR OR HARD ERROR.
                                ;WILL GO TO READ SECTOR AFTER ALL SECTORS ON TRACK ARE WRITTEN.
                                ;A RESTORE IS PERFORMED AFTER ANY SEEK ERROR (RNF).
                                ;;*****************************************************************

013332  005737  177170        DOWCHK: TST     RXCS            ;ERROR?
013336  100024                        BPL     1$              ;BR IN NC
013340  023727  002404  000012        CMP     DOERR,#10.      ;10 ERRORS?
013346  001072                        BNE     6$              ;BR IF NO & TRY AGAIN
013350  004737  011716              JSR     PC,LDOBAD       ;LOAD BAD TRK/SEC TABLE
013354  032737  000020  013206        BIT     #RNF,SRXES      ;SEEK ERROR?
013362  001410                        BEQ     7$              ;BR IF NO
013364  104064                        ERROR   64              ;HARD SEEK ERROR WRITE SECTOR
013366  012737  013444  013174        MOV     #2$,DODISP      ;GO TO 2$ AFTER RESTORE
013374  012737  000117  177170        MOV     #<RESTOR!IE>,RXCS
013402  000002                        RTI

013404  104016          7$:     ERROR   16              ;HARD ERROR WRITE SECTOR
013406  000416                  BR      2$              ;GO TO NEXT SECTOR

013410  005737  002404          1$:     TST     DOERR           ;ANY PREV ERRORS?
013414  001413                  BEQ     2$              ;BR IF NO
013416  005237  025770                  INC     TSERR           ;TOTAL SOFT ERR CT
013422  005237  025774                  INC     PSERR           ;PASS  SOFT ERR CT
013426  032737  000020  013206        BIT     #RNF,SRXES      ;SEEK ERROR?
013434  001402                        BEQ     8$              ;BR IF NO
013436  104065                        ERROR   65              ;SOFT SEEK ERROR WRITE SECTOR
013440  000401                        BR      2$
013442  104017          8$:     ERROR   17              ;SOFT ERROR WRITE SECTOR

013444  005037  002404          2$:     CLR     DOERR
013450  023727  002410  000031        CMP     DOSEC,#31       ;LAST ODD SECTOR?
013456  001004                        BNE     3$              ;BR IF NO
013460  012737  000002  002410        MOV     #2,DOSEC        ;ELSE DO EVEN SECTORS NOW
013466  000407                        BR      4$
013470  023727  002410  000032  3$:     CMP     DOSEC,#32       ;LAST EVEN SECTOR?
013476  001405                        BEQ     5$              ;BR IF YES
013500  062737  000002  002410        ADD     #2,DOSEC        ;ELSE BUMP SECTOR
013506  000137  013212          4$:     JMP     DOFBUF          ;GO TO FILL BUFFER & DO ANOTHER

013512  012737  000001  002410  5$:     MOV     #1,DOSEC        ;ALL SECTORS DONE...CHECK DATA
013520  005037  002404                  CLR     DOERR
013524  005037  002412                  CLR     DOCMER
013530  000137  013566                  JMP     DORSEC          ;GO TO READ SECTOR HANDLER

013534  005237  002404          6$:     INC     DOERR
013540  032737  000020  013206        BIT     #RNF,SRXES      ;SEEK ERROR?
013546  001757                        BEQ     4$              ;BR IN NO
013550  012737  013212  013174        MOV     #DOFBUF,DODISP  ;GOTO FILL BUFF AFTER RESTORE
013556  012737  000117  177170        MOV     #<RESTOR.IE>,RXCS
013564  000002                        RTI
```

```
3301                                    ;;*****************************************************************
3302                                    ;HANDLER TO READ A SECTOR ON DX0 & POINT TO CHECK.
3303                                    ;;*****************************************************************
3304
3305  013566  013701  002406           DORSEC: MOV     DOTRK,R1
3306  013572  000301                           SWAB    R1
3307  013574  053701  002410                   BIS     DOSEC,R1
3308  013600  010137  177174                   MOV     R1,RXSA ;LOAD TRK & SECTOR ADDR
3309
3310  013604  012737  013622  013174           MOV     #DORCHK,DODISP  ;POINT TO READ CHECK HANDLER
3311  013612  012737  000107  177170           MOV     #<RSEC!IE>,RXCS ;ISSUE READ SECTOR COMMAND
3312  013620  000002                           RTI
3313
3314
3315                                    ;;*****************************************************************
3316                                    ;HANDLER TO READ CHECK ON DX0.
3317                                    ,WILL GO TO NEXT SECTOR/TRK IF HARD ERROR.
3318                                    ;WILL GO TO EMPTY BUFFER IF NO ERROR OR SOFT ERROR.
3319                                    ;WILL GO TO READ SECTOR (SAME) ON ERROR.
3320                                    ;A RESTORE IS PERFORMED AFTER ANY SEEK ERROR (RNF).
3321                                    ;;*****************************************************************
3322
3323  013622  005737  177170           DORCHK: TST     RXCS            ;ERROR?
3324  013626  100054                           BPL     1$              ;BR IF NO
3325  013630  023727  002404  000012           CMP     DOERR,#10.      ;10 ERRORS?
3326  013636  001415                           BEQ     3$              ;BR IF YES
3327  013640  005237  002404                   INC     DOERR           ;ELSE DO AGAIN
3328  013644  032737  000020  013206           BIT     #RNF,SRXES      ;SEEK ERROR?
3329  013652  001745                           BEQ     DORSEC          ;BR IF NO
3330  013654  012737  013566  013174           MOV     #DORSEC,DODISP  ;ELSE DO RESTORE & RET TO RD SEC
3331  013662  012737  000117  177170           MOV     #<RESTOR!IE>,RXCS
3332  013670  000002                           RTI
3333
3334  013672  004737  011716           3$:     JSR     PC,LDOBAD       ;LOAD BAD TRK/SEC TABLE
3335  013676  032737  000020  013206           BIT     #RNF,SRXES      ;SEEK ERROR?
3336  013704  001410                           BEQ     7$              ;BR IF NO
3337  013706  104066                           ERROR   66              ;SEEK ERROR READ SECTOR
3338
3339  013710  012737  014610  013174           MOV     #DONEXT,DODISP  ;DO RESTORE
3340  013716  012737  000117  177170           MOV     #<RESTOR.IE>,RXCS
3341  013724  000002                           RTI
3342
3343  013726  032737  000010  013206  7$:      BIT     #CRC,SRXES      ;CRC ERROR?
3344  013734  001404                           BEQ     10$             ;BR IF NO
3345  013736  005237  002422                   INC     DOCRC           ;ELSE SET FLAG
3346  013742  104106                           ERROR   106             ;HARD CRC ERROR
3347  013744  000423                           BR      2$              ;& GO TO EMP BUFF TO CHK DATA
3348
3349  013746  005037  002422           10$:    CLR     DOCRC
3350  013752  104020                           ERROR   20              ;HARD READ ERROR
3351  013754  000137  014610                   JMP     DONEXT
```

```
3352
3353  013760  005737  002404          1$:     TST     DOERR           ;ANY PREV ERRORS?
3354  013764  001413                          BEQ     2$              ;BR IF NO
3355  013766  005237  025770                  INC     TSERR           ;TOTAL SOFT ERR CT
3356  013772  005237  025774                  INC     PSERR           ;PASS  SOFT ERR CT
3357  013776  032737  000020  013206          BIT     #RNF,SRXES      ;SEEK ERROR?
3358  014004  001402                          BEQ     9$              ;BR IF NO
3359  014006  104067                          ERROR   67              ;SEEK ERROR READ SECTOR
3360  014010  000401                          BR      2$
3361  014012  104021                  9$:     ERROR   21              ;SOFT ERROR  READ SECTOR
3362
3363  014014  005037  002404          2$:     CLR     DOERR
3364  014020  023727  003120  000005          CMP     DXTST,#5        ;DOING WRITE DELETED DATA TESTS?
3365  014026  001005                          BNE     6$              ;BR IF NO
3366  014030  032737  000040  177172          BIT     #DD,RXES        ; 'DELETED DATA' SET?
3367  014036  001001                          BNE     6$              ;BR IF YES
3368  014040  104052                          ERROR   52              ;DD NOT SET
3369
3370  014042  000137  014046          6$:     JMP     DOEBUF          ;GO TO EMPTY BUFFER
3371
3372
3373
3374
3375
3376
3377                                  ;;*******************************************************************
3378                                  ; HANDLER TO EMPTY THE DRIVE BUFFER FOR DX0
3379                                  ;& POINT TO THE CHECK DATA HANDLER.
3380                                  ;;*******************************************************************
3381
3382  014046  012703  000100          DOEBUF: MOV     #100,R3         ;WORD CT
3383  014052  012701  002424                  MOV     #DOBUF,R1       ;BUFFER ADDRESS
3384  014056  012737  014104  013174          MOV     #DOCDAT,DODISP  ;POINT TO CHK DATA AFT INTER.
3385  014064  012737  000103  177170          MOV     #<EBUF!IE>,RXCS ;ISSUE EMPTY BUFFER CMD
3386  014072  013721  177172          1$:     MOV     RXDB,(R1)+      ;GET WORD & STORE IT
3387  014076  005303                          DEC     R3
3388  014100  001374                          BNE     1$
3389  014102  000002                          RTI
3390
3391
```

```
3392                                    ;;*****************************************************************
3393                                    ; HANDLER TO CHECK DATA AFTER EMPTY BUFFER CMD ON DX0.
3394                                    ;WILL GO TO NEXT SECTOR HANDLER WHEN NO ERR OR HARD ERR.
3395                                    ;OR TO READ SECTOR (SAME) WHEN SOFT ERR.
3396                                    ;THIS ROUTINE IS ALSO ENTERED FROM A CRC ERROR FROM A READ COMMAND.
3397                                    ;;*****************************************************************
3398
3399  014104  005037  002412           DOCDAT: CLR     DOCMER
3400  014110  012703  000100                   MOV     #100,R3         ;WORD CTR
3401  014114  013701  002406                   MOV     DOTRK,R1
3402  014120  000301                           SWAB    R1
3403  014122  053701  002410                   BIS     DOSEC,R1        ;R1 NOW HAS EXPECTED DATA
3404  014126  012702  002424                   MOV     #DOBUF,R2       ;GET BUFFER ADDR
3405  014132  020112                   1$:     CMP     R1,(R2)         ;COMPARE OK?
3406  014134  001407                           BEQ     2$              ;BR IF YES
3407  014136  010137  002416                   MOV     R1,DOEXP        ;ELSE SAVE
3408  014142  011237  002420                   MOV     (R2),DOREC
3409  014146  005237  002412                   INC     DOCMER
3410  014152  000405                           BR      3$
3411
3412  014154  005303                   2$:     DEC     R3
3413  014156  001403                           BEQ     3$
3414  014160  062702  000002                   ADD     #2,R2
3415  014164  000762                           BR      1$
3416
3417  014166  005737  002412           3$:     TST     DOCMER          ;ANY COMP ERR?
3418  014172  001421                           BEQ     4$              ;BR IF NO
3419  014174  005737  002422                   TST     DOCRC           ;HERE FROM CRC ERROR?
3420  014200  001402                           BEQ     14$             ;BR IF NO
3421  014202  104107                           ERROR   107             ;DATA CRC ERROR
3422  014204  000431                           BR      5$
3423
3424  014206  023727  002414  000012   14$:    CMP     DOCERR,#10.     ;ELSE, 10 ERRS YET?
3425  014214  001033                           BNE     7$              ;BR IF NO
3426  014216  023727  003120  000003           CMP     DXTST,#3        ;DOING INIT TEST?
3427  014224  001002                           BNE     8$              ;BR IN NO
3428  014226  104053                           ERROR   53              ;INITIALIZE ERROR
3429  014230  000417                           BR      5$
3430
3431  014232  104022                   8$:     ERROR   22              ;HARD ERROR  DATA COMPARE
3432  014234  000415                           BR      5$
3433
3434  014236  005737  002422           4$:     TST     DOCRC           ;HERE FROM CRC ERROR?
3435  014242  001402                           BEQ     15$             ;BR IF NO
3436  014244  104110                           ERROR   110             ;CRC ERR WITH DATA OK
3437  014246  000410                           BR      5$
3438
3439  014250  005737  002414           15$:    TST     DOCERR          ;ANY PREV ERR?
3440  014254  001405                           BEQ     5$              ;BR IF NO
3441  014256  005237  025770                   INC     TSERR           ;TOTAL SOFT ERR CT
3442  014262  005237  025774                   INC     PSERR           ;PASS  SOFT ERR CT
3443  014266  104023                           ERROR   23              ;SOFT ERROR  DATA COMP
```

```
3444
3445  014270  005037  002414          5$:     CLR     DOCERR
3446  014274  005037  002422                  CLR     DOCRC
3447  014300  000137  014610                  JMP     DONEXT
3448
3449  014304  005237  002414          7$:     INC     DOCERR
3450  014310  023727  003120  000003          CMP     DXTST,#3        ;DOING INIT TEST?
3451  014316  001402                          BEQ     12$             ;BR IF YES
3452  014320  000137  013566                  JMP     DORSEC          ;ELSE READ SECTOR OVER AGAIN
3453
3454  014324  000137  014746          12$:    JMP     DODUN           ;ALL DONE
3455
3456
3457
3458                                  ;;*****************************************************************
3459                                  ;HANDLER TO SETUP NEXT SECTOR/TRACK FOR DX0.
3460                                  ;WILL GO TO READ SECTOR (NEXT) IF ALL SECTORS ON TRACK NOT DONE.
3461                                  ;WILL GO TO FILL BUFFER (NEXT TRACK) IF ALL SECTORS ON TRACK DONE.
3462                                  ;WILL SHUT OFF DRV INTERRUPTS IF ALL TRACKS DONE, WITH A MESSAGE.
3463                                  ;;*****************************************************************
3464
3465  014330  005037  002404          DONXT1: CLR     DOERR
3466  014334  005037  002412                  CLR     DOCMER
3467  014340  023727  002410  000031          CMP     DOSEC,#31       ;LAST ODD SECTOR?
3468  014346  001004                          BNE     1$              ;BR IF NO
3469  014350  012737  000002  002410          MOV     #2,DOSEC        ;ELSE DO EVEN SECTORS NOW
3470  014356  000407                          BR      2$
3471  014360  023727  002410  000032  1$:     CMP     DOSEC,#32       ;LAST EVEN SECTOR?
3472  014366  001405                          BEQ     3$              ;BR IF YES
3473  014370  062737  000002  002410          ADD     #2,DOSEC        ;ELSE BUMP SECTOR
3474  014376  000137  013566          2$:     JMP     DORSEC          ;GO READ SECTOR
3475
3476  014402  032777  010000  164530  3$:     BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
3477  014410  001411                          BEQ     6$
3478  014412  104401  017232                  TYPE    ,DRV0
3479  014416  104401  017705                  TYPE    ,TRK
3480  014422  013746  002406                  MOV     DOTRK,-(SP)     ;;SAVE DOTRK FOR TYPEOUT
3481  014426  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
3482  014430  104401  017756                  TYPE    ,DUN
3483  014434  005237  003122          6$:     INC     FIN             ;SETUP TO GO TO DX1
3484  014440  023737  002406  003124          CMP     DOTRK,MAXTRK    ;LAST TRACK?
3485  014446  001406                          BEQ     4$              ;BR IF YES
3486  014450  005237  002406                  INC     DOTRK           ;BUMP TRACK
3487  014454  012737  000001  002410          MOV     #1,DOSEC        ;INIT SECTOR
3488  014462  000414                          BR      5$
3489
3490  014464  032777  010000  164446  4$:     BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
3491  014472  001406                          BEQ     7$
3492  014474  104401  017232                  TYPE    ,DRV0
3493  014500  104401  017712                  TYPE    ,PATT
3494  014504  104401  017756                  TYPE    ,DUN
3495  014510  005237  002400          7$:     INC     DOPAT           ;PATT DONE FLAG
3496  014514  042737  000100  177170  5$:     BIC     #IE,RXCS        ;DISABLE DX0 INTERRUPTS
3497  014522  000002                          RTI
```

```
3498
3499                                    ;;*****************************************************************
3500                                    ;HANDLER TO SETUP TO GO TO NEXT RANDOM TRACK/SECTOR FOR DX0.
3501                                    ;;*****************************************************************
3502
3503  014524  005037  002404           DONXT2: CLR     DOERR
3504  014530  005037  002412                   CLR     DOCMER
3505  014534  005237  003122                   INC     FIN             ;SETUP TO GO TO DX1
3506  014540  005237  002402                   INC     DOCNT
3507  014544  023737  002402  003126           CMP     DOCNT,MAXSK     ;DID ALL SEEKS?
3508  014552  001012                           BNE     1$              ;BR IF NO
3509  014554  032777  010000  164356           BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
3510  014562  001406                           BEQ     1$
3511  014564  104401  017232                   TYPE    ,DRV0
3512  014570  104401  017725                   TYPE    ,RANDOM
3513  014574  104401  017756                   TYPE    ,DUN
3514  014600  042737  000100  177170    $:     BIC     #IE,RXCS        ;DISABLE DX0 INTERRUPTS
3515  014606  000002                           RTI
3516
3517
3518
3519
3520
3521  014610  023727  003120  000001   DONEXT: CMP     DXTST,#1
3522  014616  001644                           BEQ     DONXT1
3523  014620  023727  003120  000002           CMP     DXTST,#2
3524  014626  001736                           BEQ     DONXT2
3525  014630  000446                           BR      DODUN
3526
```

```
3527
3528                                    ;;*****************************************************************
3529                                    ;HANDLER TO SETUP INITIALIZE COMMAND TEST
3530                                    ;;*****************************************************************
3531
3532  014632  012737  046032  177174  DOINIT: MOV     #46032,RXSA     ;TRY TO FAKE DRIVE TO GO TO TRK 76 SEC 26
3533                                                                    ;INIT SHOULD GO TO TRK 1, SEC 1.
3534  014640  012737  014746  013174          MOV     #DODUN,DODISP   ;POINT TO DONE HANDLER
3535  014646  012737  000111  177170          MOV     #<INITAL!IE>,RXCS       ;DO INIT COMMAND
3536  014654  000002                          RTI
3537
3538
3539                                    ;;*****************************************************************
3540                                    ;HANDLERS TO SETUP RESTORE COMMAND TEST
3541                                    ;;*****************************************************************
3542
3543  014656  012737  014746  013174  DOREST: MOV     #DODUN,DODISP   ;POINT TO DONE HANDLER
3544  014664  012737  000117  177170          MOV     #<RESTOR!IE>,RXCS
3545  014672  000002                          RTI
3546
3547  014674  012737  014762  013202  D1REST: MOV     #D1DUN,D1DISP
3548  014702  012737  000137  177170          MOV     #<RESTOR!IE!DX1>,RXCS
3549  014710  000002                          RTI
3550
3551                                    ;;*****************************************************************
3552                                    ;HANDLER TO SETUP READ STATUS COMMAND TEST
3553                                    ;;*****************************************************************
3554
3555  014712  012737  014746  013174  DOSTAT: MOV     #DODUN,DODISP   ;POINT TO DONE
3556  014720  012737  000113  177170          MOV     #<RSTAT!IE>,RXCS        ;DO READ STATUS COMMAND
3557  014726  000002                          RTI
3558
3559                                    ;;*****************************************************************
3560                                    ;HANDLER TO START INVALID ADDRESS TESTS
3561                                    ;;*****************************************************************
3562
3563  014730  012737  014746  013174  DOINV:  MOV     #DODUN,DODISP   ;POINT TO DONE
3564  014736  012737  000105  177170          MOV     #<WSEC!IE>,RXCS ;DO WRITE SECTOR COMMAND
3565  014744  000002                          RTI
3566
3567
3568                                    ;;*****************************************************************
3569                                    ;HANDLERS TO FINISH INITIALIZE, RESTORE, WRITE DELETED DATA,
3570                                    ;READ STATUS & INVALID ADDRESS TESTS.
3571                                    ;;*****************************************************************
3572
3573  014746  005237  003122          DODUN:  INC     FIN             ;SET DONE FLAG
3574  014752  042737  000100  177170          BIC     #IE,RXCS        ;DISABLE FURTHUR DX0 INTERRUPTS
3575  014760  000002                          RTI
3576
3577  014762  005237  003122          D1DUN:  INC     FIN
3578  014766  042737  000120  177170          BIC     #<IE!DX1>,RXCS
3579  014774  000002                          RTI
3580
```

```
3581
3582
3583                                    ;;*****************************************************************
3584                                    ; HANDLER TO FILL THE DRIVE BUFFER FOR DX1
3585                                    ;& POINT TO THE WRITE SECTOR HANDLER.
3586                                    ;;*****************************************************************
3587
3588  014776  012703  000100           D1FBUF: MOV     #100,R3         ;WORD CT
3589  015002  013701  002656                   MOV     D1TRK,R1
3590  015006  000301                           SWAB    R1
3591  015010  053701  002660                   BIS     D1SEC,R1
3592
3593  015014  005737  003132                   TST     COMP1           ;DONT SET BIT15 IN COMPATABILTIY TESTS
3594  015020  001005                           BNE     2$
3595  015022  005737  003134                   TST     COMP2
3596  015026  001002                           BNE     2$
3597  015030  052701  100000                   BIS     #BIT15,R1       ;TO DISTINGUISH DX1 DATA FROM DX0 DATA
3598
3599  015034  012737  015062  013202   2$:     MOV     #D1WSEC,D1DISP  ;POINT TO WRITE SECTOR AFTER RII
3600  015042  012737  000121  177170           MOV     #<FBUF!IE!DX1>,RXCS     ;ISSUE FILL BUFFER CMD
3601  015050  010137  177172           1$:     MOV     R1,RXDB         ;FILL ENTIRE SECTOR WITH ITS ADDRESS
3602  015054  005303                           DEC     R3              ;& BIT 15 SET
3603  015056  001374                           BNE     1$
3604  015060  000002                           RTI
3605
3606
3607                                    ;;*****************************************************************
3608                                    ;HANDLER TO WRITE THE SECTOR FOR DX1 & POINT TO WRITE CHECK.
3609                                    ;;*****************************************************************
3610
3611  015062  013701  002656           D1WSEC: MOV     D1TRK,R1
3612  015066  000301                           SWAB    R1
3613  015070  053701  002660                   BIS     D1SEC,R1
3614  015074  010137  177174                   MOV     R1,RXSA ;LOAD TRACK & SECTOR ADDR
3615  015100  012737  015116  013202           MOV     #D1WCHK,D1DISP  ;POINT TO WRITE CHECK HANDLER
3616  015106  012737  000125  177170           MOV     #<WSEC!IE!DX1>,RXCS     ;ISSUE WRITE SECTOR COMMAND
3617  015114  000002                           RTI
3618
```

```
3619                                    ;;*****************************************************************
3620                                    ;HANDLER TO WRITE CHECK ON DX1.
3621                                    ;WILL GO TO FILL BUFFER TO DO SAME SECTOR ON SOFT ERROR
3622                                    ;OR NEXT SECTOR ON NO ERROR OR HARD ERROR.
3623                                    ;WILL GO TO READ SECTOR AFTER ALL SECTORS ON TRACK ARE WRITTEN.
3624                                    ;A RESTORE IS PERFORMED AFTER ANY SEEK ERROR (RNF).
3625                                    ;;*****************************************************************
3626
3627  015116  005737  177170       D1WCHK: TST     RXCS            ;ERROR?
3628  015122  100024                       BPL     1$              ;BR IN NO
3629  015124  023727  002654  000012       CMP     D1ERR,#10.      ;10 ERRORS?
3630  015132  001072                       BNE     6$              ;BR IF NO & TRY AGAIN
3631  015134  004737  012030               JSR     PC,LD1BAD       ;LOAD BAD TRK/SEC TABLE
3632  015140  032737  000020  013206       BIT     #RNF,SRXES      ;SEEK ERROR?
3633  015146  001410                       BEQ     7$              ;BR IF NO
3634  015150  104070                       ERROR   70              ;SEEK ERROR WRITE SECTOR
3635  015152  012737  015230  013202       MOV     #2$,D1DISP      ;RET TO 2$ AFT RESTORE
3636  015160  012737  000137  177170       MOV     #<RESTOR!IE!DX1>,RXCS
3637  015166  000002                       RTI
3638
3639  015170  104024                 7$:   ERROR   24              ;ERROR WRITE SECTOR
3640  015172  000416                       BR      2$              ;GO TO NEXT SECTOR
3641
3642  015174  005737  002654         1$:   TST     D1ERR           ;ANY PREV ERRORS?
3643  015200  001413                       BEQ     2$              ;BR IF NO
3644  015202  005237  025770               INC     TSERR           ;TOTAL SOFT ERR CT
3645  015206  005237  025774               INC     PSERR           ;PASS  SOFT ERR CT
3646  015212  032737  000020  013206       BIT     #RNF,SRXES      ;SEEK ERROR?
3647  015220  001402                       BEQ     8$              ;BR IF NO
3648  015222  104071                       ERROR   71              ;SEEK ERROR WRITE SECTOR
3649  015224  000401                       BR      2$
3650  015226  104025                 8$:   ERROR   25              ;ERROR WRITE SECTOR
3651
3652  015230  005037  002654         2$:   CLR     D1ERR
3653  015234  023727  002660  000031       CMP     D1SEC,#31       ;LAST ODD SECTOR?
3654  015242  001004                       BNE     3$              ;BR IF NO
3655  015244  012737  000002  002660       MOV     #2,D1SEC        ;ELSE DO EVEN SECTORS NOW
3656  015252  000407                       BR      4$
3657  015254  023727  002660  000032  3$:   CMP     D1SEC,#32       ;LAST EVEN SECTOR?
3658  015262  001405                       BEQ     5$              ;BR IF YES
3659  015264  062737  000002  002660       ADD     #2,D1SEC        ;ELSE BUMP SECTOR
3660  015272  000137  014776         4$:   JMP     D1FBUF          ;GO TO FILL BUFFER & DO ANOTHER
3661
3662  015276  012737  000001  002660  5$:   MOV     #1,D1SEC        ;ALL SECTORS DONE...CHECK DATA
3663  015304  005037  002654               CLR     D1ERR
3664  015310  005037  002662               CLR     D1CMER
3665  015314  000137  015352               JMP     D1RSEC          ;GO TO READ SECTOR HANDLER
3666
3667  015320  005237  002654         6$:   INC     D1ERR
3668  015324  032737  000020  013206       BIT     #RNF,SRXES      ;SEEK ERROR?
3669  015332  001757                       BEQ     4$              ;BR IF NO
3670  015334  012737  014776  013202       MOV     #D1FBUF,D1DISP  ;RET TO FILL BUFF AFT RESTORE
3671  015342  012737  000137  177170       MOV     #<RESTOR!IE.DX1>,RXCS
3672  015350  000002                       RTI
3673
3674                                    ;;*****************************************************************
```

```
3675                                    ;HANDLER TO READ A SECTOR ON DX1 & POINT TO CHECK.
3676                                    ;;*****************************************************************
3677
3678  015352  013701  002656            D1RSEC: MOV     D1TRK,R1
3679  015356  000301                            SWAB    R1
3680  015360  053701  002660                    BIS     D1SEC,R1
3681  015364  010137  177174                    MOV     R1,RXSA ;LOAD TRK & SECTOR ADDR
3682
3683  015370  012737  015406  013202            MOV     #D1RCHK,D1DISP  ;POINT TO READ CHECK HANDLER
3684  015376  012737  000127  177170            MOV     #<RSEC!IE!DX1>,RXCS     ;ISSUE READ SECTOR COMMAND
3685  015404  000002                            RTI
3686
3687
3688
3689                                    ;;*****************************************************************
3690                                    ;HANDLER TO READ CHECK ON DX1.
3691                                    ;WILL GO TO NEXT SECTOR/TRK IF HARD ERROR.
3692                                    ;WILL GO TO EMPTY BUFFER IF NO ERROR OR SOFT ERROR.
3693                                    ;WILL GO TO READ SECTOR (SAME) ON ERROR.
3694                                    ;A RESTORE IS PERFORMED AFTER ANY SEEK ERROR (RNF).
3695                                    ;;*****************************************************************
3696
3697  015406  005737  177170            D1RCHK: TST     RXCS            ;ERROR?
3698  015412  100054                            BPL     1$              ;BR IF NO
3699  015414  023727  002654  000012            CMP     D1ERR,#10.      ;10 ERRORS?
3700  015422  001415                            BEQ     3$              ;BR IF YES
3701  015424  005237  002654                    INC     D1ERR           ;ELSE TRY AGAIN
3702  015430  032737  000020  013206            BIT     #RNF,SRXES      ;SEEK ERROR?
3703  015436  001745                            BEQ     D1RSEC          ;BR IF NO
3704  015440  012737  015352  013202            MOV     #D1RSEC,D1DISP  ;ELSE DO RESTORE & RET TO RD SEC
3705  015446  012737  000137  177170            MOV     #<RESTOR!IE!DX1>,RXCS
3706  015454  000002                            RTI
3707
3708  015456  004737  012030            3$:     JSR     PC,LD1BAD       ;LOAD BAD TRK/SEC TABLE
3709  015462  032737  000020  013206            BIT     #RNF,SRXES      ;SEEK ERROR?
3710  015470  001410                            BEQ     7$              ;BR IF NO
3711  015472  104072                            ERROR   72              ;SEEK ERROR READ SECTOR
3712
3713  015474  012737  016342  013202            MOV     #D1NEXT,D1DISP  ;DO RESTORE
3714  015502  012737  000137  177170            MOV     #<RESTOR!IE!DX1>,RXCS
3715  015510  000002                            RTI
3716
3717  015512  032737  000010  013206  7$:       BIT     #CRC,SRXES      ;CRC ERROR?
3718  015520  001404                            BEQ     10$             ;BR IF NO
3719  015522  005237  002672                    INC     D1CRC           ;ELSE SET FLAG
3720  015526  104111                            ERROR   111             ;HARD CRC ERROR
3721  015530  000423                            BR      2$              ;& GO TO EMP BUFF TO CHK DATA
3722
3723  015532  005037  002672            10$:    CLR     D1CRC
3724  015536  104026                            ERROR   26              ;HARD READ ERROR
3725  015540  000137  016342                    JMP     D1NEXT
```

```
3726
3727  015544  005737  002654          1$:     TST     D1ERR           ;ANY PREV ERRORS?
3728  015550  001413                          BEQ     2$              ;BR IF NO
3729  015552  005237  025770                  INC     TSERR           ;TOTAL SOFT ERR CT
3730  015556  005237  025774                  INC     PSERR           ;PASS  SOFT ERR CT
3731  015562  032737  000020  013206          BIT     #RNF,SRXES      ;SEEK ERROR?
3732  015570  001402                          BEQ     9$              ;BR IF NO
3733  015572  104073                          ERROR   73              ;SEEK ERROR READ SECTOR
3734  015574  000401                          BR      2$
3735  015576  104027                  9$:     ERROR   27              ;SOFT ERROR  READ SECTOR
3736
3737  015600  005037  002654          2$:     CLR     D1ERR
3738  015604  000137  015610                  JMP     D1EBUF          ;GO TO EMPTY BUFFER
3739
3740
3741
3742
3743
3744
3745                                  ;;*******************************************************************
3746                                  ; HANDLER TO EMPTY THE DRIVE BUFFER FOR DX1
3747                                  ;& POINT TO THE CHECK DATA HANDLER.
3748                                  ;;*******************************************************************
3749
3750  015610  012703  000100          D1EBUF: MOV     #100,R3         ;WORD CT
3751  015614  012701  002674                  MOV     #D1BUF,R1       ;BUFFER ADDRESS
3752  015620  012737  015646  013202          MOV     #D1CDAT,D1DISP  ;POINT TO CHK DATA AFT INTER.
3753  015626  012737  000123  177170          MOV     #<EBUF!IE.DX1>,RXCS     ;ISSUE EMPTY BUFFER CMD
3754  015634  013721  177172          1$:     MOV     RXDB,(R1)+      ;GET WORD & STORE IT
3755  015640  005303                          DEC     R3
3756  015642  001374                          BNE     1$
3757  015644  000002                          RTI
3758
```

```
3759                                    ;;*****************************************************************
3760                                    ; HANDLER TO CHECK DATA AFTER EMPTY BUFFER CMD ON DX1.
3761                                    ;WILL GO TO NEXT SECTOR HANDLER WHEN NO ERR OR HARD ERR.
3762                                    ;OR TO READ SECTOR (SAME) WHEN SOFT ERR.
3763                                    ;THIS ROUTINE IS ALSO ENTERED FROM A CRC ERROR FROM A READ COMMAND.
3764                                    ;;*****************************************************************
3765
3766  015646  005037  002662           D1CDAT: CLR     D1CMER
3767  015652  012703  000100                   MOV     #100,R3         ;WORD CTR
3768  015656  013701  002656                   MOV     D1TRK,R1
3769  015662  000301                           SWAB    R1
3770  015664  053701  002660                   BIS     D1SEC,R1
3771
3772  015670  005737  003132                   TST     COMP1           ;DONT SET BIT15 IN COMPATABILITY TESTS
3773  015674  001005                           BNE     6$
3774  015676  005737  003134                   TST     COMP2
3775  015702  001002                           BNE     6$
3776  015704  052701  100000                   BIS     #BIT15,R1       ;TO DISTINGUISH DX1 DATA FROM DX0 DATA
3777
3778  015710  012702  002674           6$:     MOV     #D1BUF,R2       ;GET BUFFER ADDR
3779  015714  020112                   1$:     CMP     R1,(R2)         ;COMPARE OK?
3780  015716  001407                           BEQ     2$              ;BR IF YES
3781  015720  010137  002666                   MOV     R1,D1EXP        ;ELSE SAVE
3782  015724  011237  002670                   MOV     (R2),D1REC
3783  015730  005237  002662                   INC     D1CMER
3784  015734  000405                           BR      3$
3785  015736  005303                   2$:     DEC     R3
3786  015740  001403                           BEQ     3$
3787  015742  062702  000002                   ADD     #2,R2
3788  015746  000762                           BR      1$
3789
3790  015750  005737  002662           3$:     TST     D1CMER          ;ANY COMP ERR?
3791  015754  001413                           BEQ     4$              ;BR IF NO
3792  015756  005737  002672                   TST     D1CRC           ;HERE FROM CRC ERROR?
3793  015762  001402                           BEQ     14$             ;BR IF NO
3794  015764  104112                           ERROR   112             ;DATA CRC ERROR
3795  015766  000423                           BR      5$
3796  015770  023727  002664  000002   14$:    CMP     D1CERR,#10.     ;ELSE, 10 ERRS YET?
3797  015776  001025                           BNE     7$              ;BR IF NO
3798  016000  104030                           ERROR   30              ;HARD ERROR  DATA COMP
3799  016002  000415                           BR      5$
3800
3801  016004  005737  002672           4$:     TST     D1CRC           ;HERE FROM CRC ERROR?
3802  016010  001402                           BEQ     15$             ;BR IF NO
3803  016012  104113                           ERROR   113             ;CRC ERR WITH DATA OK
3804  016014  000410                           BR      5$
3805  016016  005737  002664           15$:    TST     D1CERR          ;ANY ERR?
3806  016022  001405                           BEQ     5$
3807  016024  005237  025770                   INC     TSERR           ;TOTAL SOFT ERR CT
3808  016030  005237  025774                   INC     PSERR           ;PASS  SOFT ERR CT
3809  016034  104031                           ERROR   31              ;SOFT ERROR  DATA COMP
```

```
3810
3811  016036  005037  002664          5$:     CLR     D1CERR
3812  016042  005037  002672                  CLR     D1CRC
3813  016046  000137  016342                  JMP     D1NEXT
3814
3815  016052  005237  002664          7$:     INC     D1CERR
3816  016056  000137  015352                  JMP     D1RSEC          ;READ SECTOR OVER AGAIN
3817
3818
3819                                  ;;*****************************************************************
3820                                  ;HANDLER TO SETUP NEXT SECTOR/TRACK FOR DX1.
3821                                  ;WILL GO TO READ SECTOR (NEXT) IF ALL SECTORS ON TRACK NOT DONE.
3822                                  ;WILL GO TO FILL BUFFER (NEXT TRACK) IF ALL SECTORS ON TRACK DONE.
3823                                  ;WILL SHUT OFF DRV INTERRUPTS IF ALL TRACKS DONE, WITH A MESSAGE.
3824                                  ;;*****************************************************************
3825
3826  016062  005037  002654          D1NXT1: CLR     D1ERR
3827  016066  005037  002662                  CLR     D1CMER
3828  016072  023727  002660  000031          CMP     D1SEC,#31       ;LAST ODD SECTOR?
3829  016100  001004                          BNE     1$              ;BR IF NO
3830  016102  012737  000002  002660          MOV     #2,D1SEC        ;ELSE DO EVEN SECTORS NOW
3831  016110  000407                          BR      2$
3832  016112  023727  002660  000032  1$:     CMP     D1SEC,#32       ;LAST EVEN SECTOR?
3833  016120  001405                          BEQ     3$              ;BR IF YES
3834  016122  062737  000002  002660          ADD     #2,D1SEC        ;ELSE BUMP SECTOR
3835  016130  000137  015352          2$:     JMP     D1RSEC          ;GO READ SECTOR
3836
3837  016134  032777  010000  162776  3$:     BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
3838  016142  001411                          BEQ     6$
3839  016144  104401  017237                  TYPE    ,DRV1
3840  016150  104401  017705                  TYPE    ,TRK
3841  016154  013746  002656                  MOV     D1TRK,-(SP)     ;;SAVE D1TRK FOR TYPEOUT
3842  016160  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
3843  016162  104401  017756                  TYPE    ,DUN
3844  016166  005237  003122          6$:     INC     FIN             ;SETUP TO GO TO DX0
3845  016172  023737  002656  003124          CMP     D1TRK,MAXTRK    ;LAST TRACK?
3846  016200  001406                          BEQ     4$              ;BR IF YES
3847  016202  005237  002656                  INC     D1TRK           ;BUMP TRACK
3848  016206  012737  000001  002660          MOV     #1,D1SEC        ;INIT SECTOR
3849  016214  000414                          BR      5$
3850
3851  016216  032777  010000  162714  4$:     BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
3852  016224  001406                          BEQ     7$
3853  016226  104401  017237                  TYPE    ,DRV1
3854  016232  104401  017712                  TYPE    ,PATT
3855  016236  104401  017756                  TYPE    ,DUN
3856  016242  005237  002650          7$:     INC     D1PAT           ;PATT DONE FLAG
3857  016246  042737  000120  177170  5$:     BIC     #<IE.DX1>,RXCS  ;DISABLE DX1 INTERRUPTS
3858  016254  000002                          RTI
3859
```

```
3860
3861                                    ;;*****************************************************************
3862                                    ;HANDLER TO SETUP TO GO TO NEXT RANDOM TRACK/SECTOR FOR DX1.
3863                                    ;;*****************************************************************
3864
3865  016256  005037  002654           D1NXT2: CLR     D1ERR
3866  016262  005037  002662                   CLR     D1CMER
3867  016266  005237  003122                   INC     FIN             ;SETUP TO GO TO DX0
3868  016272  005237  002652                   INC     D1CNT
3869  016276  023737  002652  003126           CMP     D1CNT,MAXSK     ;DID ALL SEEKS?
3870  016304  001012                           BNE     1$              ;BR IF NO
3871  016306  032777  010000  162624           BIT     #BIT12,@SWR     ;SKIP TYPEOUT IF NOT SET
3872  016314  001406                           BEQ     1$
3873  016316  104401  017237                   TYPE    ,DRV1
3874  016322  104401  017725                   TYPE    ,RANDOM
3875  016326  104401  017756                   TYPE    ,DUN
3876  016332  042737  000120  177170    $:     BIC     #<IE.DX1>,RXCS  ;DISABLE DX1 INTERRUPTS
3877  016340  000002                           RTI
3878
3879
3880
3881
3882  016342  023727  003120  000001   D1NEXT: CMP     DXTST,#1
3883  016350  001644                           BEQ     D1NXT1
3884  016352  000741                           BR      D1NXT2
3885
```

```
                                ;;*****************************************************************
                                ;THE FOLLOWING CODE, UP TO THE PROGRAM MESSAGES, IS A UTILITY TO COPY
                                ;THE DISKETTE CONTAINING ONLY THE BOOT & THE EXERCISER FROM DX0 TO DX1.
                                ;THE OPERATOR CAN TYPE 'P' TO DO ANOTHER COPY OR
                                ;TYPE '240G' TO ENTER NORMAL TESTING.
                                ;ENTIRE CODE, INCLUDING TEXT & ERROR TABLE, IS 400(10) WORDS.
                                ;;*****************************************************************

016354  005037  003130        COPY:   CLR     COPFLG                  ;DONT NEED FLAG ANY MORE
016360  104401  020155                TYPE    ,COPMSG                 ;INSERT SCRATCH IN DX1 & TYPE P
016364  000000                        HALT                            ;WAIT FOR PROCEED

016366  004737  016674                JSR     PC,GETSEC               ;CALCULATE MAX SECTORS TO BE COPIED

016372  005037  016666                CLR     ERFLG                   ;INITIALIZE
016376  012737  000001  016670        MOV     #1,TRACK
016404  012737  000001  016672        MOV     #1,SECT
016412  105737  177170                TSTB    RXCS                    ;DONE?  (CTL RDY)
016416  100375                        BPL     .-4                     ;BR IF NO & TRY AGAIN

016420  004537  016744        1$:     JSR     R5,RDWR         ;READ DX0
016424  000007                           RSEC

016426  005737  177170                TST     RXCS                    ;ERROR?
016432  100011                        BPL     2$                      ;BR IF NO
016434  005237  016666                INC     ERFLG
016440  023727  016666  000012        CMP     ERFLG,#10.              ;10 ERRORS?
016446  001364                        BNE     1$                      ;BR IF NO & TRY AGAIN
016450  104100                        ERROR   100                     ;HARD ERR READ SECTOR DX0
016452  000000                        HALT
016454  000776                        BR      .-2                     ;FORCE A RESTART FROM 250

016456  005037  016666        2$:     CLR     ERFLG
016462  004537  017004                JSR     R5,EBUFF        ;EMPTY BUFFER INTO DOBUF
016466  002424                           DOBUF

016470  004537  016744        3$:     JSR     R5,RDWR         ;WRITE DX1
016474  000025                           <WSEC!DX1>

016476  005737  177170                TST     RXCS                    ;ERROR?
016502  100011                        BPL     4$                      ;BR IF NO
016504  005237  016666                INC     ERFLG
016510  023727  016666  000012        CMP     ERFLG,#10.              ;10 ERRORS?
016516  001364                        BNE     3$                      ;BR IF NO & TRY AGAIN
016520  104101                        ERROR   101                     ;ERROR WRITE SECTOR
016522  000000                        HALT
016524  000776                        BR      .-2                     ;FORCE RESTART AT 250
```

```
3934
3935  016526  005037  016666          4$:     CLR     ERFLG
3936  016532  004537  016744                  JSR     R5,RDWR         ;READ DX1
3937  016536  000027                             <RSEC!DX1>
3938
3939  016540  005737  177170                  TST     RXCS                    ;ERROR?
3940  016544  100011                          BPL     5$                      ;BR IF NO
3941  016546  005237  016666                  INC     ERFLG
3942  016552  023727  016666  000012          CMP     ERFLG,#10.              ;10 ERRORS?
3943  016560  001362                          BNE     4$                      ;BR IF NO & TRY AGAIN
3944  016562  104103                          ERROR   103                     ;HARD ERR READ SECTOR DX1
3945  016564  000000                          HALT
3946  016566  000776                          BR      .-2                     ;FORCE 250 RESTART
3947
3948  016570  004537  017004          5$:     JSR     R5,EBUFF        ;EMPTY BUFFER INTO D1BUF
3949  016574  002674                             D1BUF
3950
3951  016576  004737  017046                  JSR     PC,DATCHK       ;COMPARE D0BUF WITH D1BUF
3952
3953  016602  005237  016664                  INC     SECCNT                  ;TOTAL DONE
3954
3955  016606  023737  016664  016662          CMP     SECCNT,MAXSEC           ;DID TOTAL # SECTORS?
3956  016614  001415                          BEQ     8$                      ;BR IF YES
3957
3958  016616  023727  016672  000032          CMP     SECT,#32                ;ELSE DID WE DO LAST SECTOR ON TRACK?
3959  016624  001403                          BEQ     6$                      ;BR IF YES
3960  016626  005237  016672                  INC     SECT                    ;ELSE BUMP & DO ANOTHER
3961  016632  000672                          BR      1$
3962
3963  016634  005237  016670          6$:     INC     TRACK                   ;BUMP TRK
3964  016640  012737  000001  016672          MOV     #1,SECT                 ;INIT SECTOR
3965  016646  000664                          BR      1$                      ;& DO ANOTHER
3966
3967  016650  104401  020256          8$:     TYPE    ,COPDUN                 ;DONE MSG
3968  016654  000000                          HALT
3969  016656  000137  016354                  JMP     COPY                    ;DO AGAIN IF 'P' TYPED
3970
3971  016662  000000                  MAXSEC: 0                               ;TOTAL # SECTORS TO READ FROM DX0
3972  016664  000000                  SECCNT: 0                               ;TOTAL # SECTORS WRITTEN TO DX1
3973  016666  000000                  ERFLG:  0
3974  016670  000000                  TRACK:  0
3975  016672  000000                  SECT:   0
```

```
3976
3977                            ;;*****************************************************************
3978                            ;ROUTINE TO CALCULATE THE MAX # OF SECTORS TO READ FROM DX0
3979                            ;;*****************************************************************
3980
3981  016674  012700  026532   GETSEC: MOV     #LSTAD,R0               ;GET MAX ADDR
3982  016700  006200                   ASR     R0                      ;GET WORD CT
3983  016702  005500                   ADC     R0                      ;DONT LOOSE ANY
3984  016704  062700  000377           ADD     #377,R0
3985  016710  042700  000377           BIC     #377,R0                 ;ROUND OFF TO NEAREST WHOLE WORD
3986
3987  016714  000300                   SWAB    R0
3988  016716  006300                   ASL     R0
3989  016720  006300                   ASL     R0                      ;DIVIDE BY 100(8) FOR SECTOR CT.
3990
3991  016722  062700  000074           ADD     #<15.*4>,R0             ;ADD SECTOR CT FOR BLOCKS 0 - 14 OF DX0
3992  016726  062700  000010           ADD     #8.,R0                  ;2 MORE BLOCKS (NECESSARY FUDGE FACTOR)
3993
3994  016732  010037  016662           MOV     R0,MAXSEC               ;SAVE
3995                                                                   ;THIS IS THE MAX SECTORS WE WILL READ OFF DX0
3996                                                                   ;& WRITE TO DX1 STARTING AT TRK 1, SECT 1
3997  016736  005037  016664           CLR     SECCNT                  ;COUNT OF TOTAL SECTORS WRITTEN TO DX1
3998  016742  000207                   RTS     PC
3999
4000
4001                            ;;*****************************************************************
4002                            ;ROUTINE TO READ OF WRITE DX0 OR DX1
4003                            ;R5 POINTS TO THE COMMAND & DISK #
4004                            ;;*****************************************************************
4005
4006  016744  013700  016670   RDWR:   MOV     TRACK,R0
4007  016750  000300                   SWAB    R0
4008  016752  053700  016672           BIS     SECT,R0
4009  016756  010037  177174           MOV     R0,RXSA                 ;LOAD TRK/SEC
4010  016762  012537  177170           MOV     (R5)+,RXCS              ;ISSUE COMMAND
4011  016766  004537  011600           JSR     R5,TIMER2               ;WAIT FOR DONE
4012  016772  177170                       RXCS
4013  016774  000200                       DONE
4014  016776  104075                   ERROR   75                      ;NO DONE
4015  017000  000000                   HALT
4016  017002  000205                   RTS     R5
4017
```

```
4018
4019                                    ;;*****************************************************************
4020                                    ;ROUTINE TO EMPTY BUFFER INTO D0BUF OR D1BUF
4021                                    ;R5 POINTS TO IT
4022                                    ;;*****************************************************************
4023
4024  017004  012700  000100      EBUFF:  MOV     #100,R0                 ;WD CT
4025  017010  012501                      MOV     (R5)+,R1                ;BUFFER ADDR
4026  017012  012737  000003  177170      MOV     #EBUF,RXCS              ;ISSUE EMPTY BUFFER COMMAND
4027  017020  013721  177172      1$:     MOV     RXDB,(R1)+              ;PUT IT IN TABLE
4028  017024  005300                      DEC     R0
4029  017026  001374                      BNE     1$
4030  017030  004537  011600              JSR     R5,TIMER2               ;WAIT FOR DONE
4031  017034  177170                          RXCS
4032  017036  000200                          DONE
4033  017040  104075                      ERROR   75                      ;NO DONE
4034  017042  000000                      HALT
4035  017044  000205                      RTS     R5
4036
4037
4038                                    ;;*****************************************************************
4039                                    ;ROUTINE TO COMPARE D0BUF WITH D1BUF
4040                                    ;;*****************************************************************
4041
4042  017046  012700  000100      DATCHK: MOV     #100,R0                 ;WD CT
4043  017052  012701  002424              MOV     #D0BUF,R1
4044  017056  012702  002674              MOV     #D1BUF,R2
4045  017062  022122              1$:     CMP     (R1)+,(R2)+             ;COMPARE?
4046  017064  001403                      BEQ     2$                      ;BR IF YES
4047  017066  104102                      ERROR   102                     ;MISCOMPARE
4048  017070  000000                      HALT
4049  017072  000776                      BR      .-2
4050
4051  017074  005300              2$:     DEC     R0
4052  017076  001371                      BNE     1$
4053  017100  000207                      RTS     PC
4054
```

```
4055
4056                                    .SBTTL  PROGRAM MESSAGES
4057
4058                                    ;;*****************************************************************
4059                                    ; MESSAGES & ERROR FORMATS
4060                                    ;;*****************************************************************
4061
4062                                    .NLIST  BEX

017102  041600  052514  052123  CLOPT:  .ASCIZ  <CRLF>/CLUSTER OPTION/
017122  052200  051105  020115  CLT1:   .ASCIZ  <CRLF>/TERM #1/
017133     200  042524  046522  CLT2:   .ASCIZ  <CRLF>/TERM #2/
017144  052200  051105  020115  CLT3:   .ASCIZ  <CRLF>/TERM #3/
017155     200  054523  041516  SCOM:   .ASCIZ  <CRLF>/SYNC COMM/
017170  040600  054523  041516  ACOM:   .ASCIZ  <CRLF>/ASYNC COMM/
017204  050200  044522  052116  PRINT:  .ASCIZ  <CRLF>/PRINTER/
017215     040  047520  052122  PORT:   .ASCIZ  / PORT TESTED/
017232  042200  030130     000  DRV0:   .ASCIZ  <CRLF>/DX0/
017237     200  054104  000061  DRV1:   .ASCIZ  <CRLF>/DX1/
017244  041600  047514  045503  CLOCK:  .ASCIZ  <CRLF>/CLOCK/
017253     200  045464     000  M4K:    .ASCIZ  <CRLF>/4K/
017257     200  045470     000  M8K:    .ASCIZ  <CRLF>/8K/
017263     200  031061  000113  M12K:   .ASCIZ  <CRLF>/12K/
017270  030600  045466     000  M16K:   .ASCIZ  <CRLF>/16K/
017275     200  030062  000113  M20K:   .ASCIZ  <CRLF>/20K/
017302  031200  045464     000  M24K:   .ASCIZ  <CRLF>/24K/
017307     200  030063  000113  M30K:   .ASCIZ  <CRLF>/30K/
017314  046440  046505  051117  MEM:    .ASCIZ  / MEMORY PRESENT/


017334  047040  052117  050040  NOTPRES:        .ASCIZ  / NOT PRESENT/
017351     200  044412  051516  WARNING:        .ASCII  <CRLF><LF>/INSERT SCRATCH DISKS, TYPE 'P' FOR NORMAL TESTING/
017434  023600  032062  043460          .ASCII  <CRLF>/'240G' FOR NORMAL RESTARTS/
017467     200  031047  030065          .ASCII  <CRLF>/'250G' TO COPY SYS EXERCISER DISK/
017531     200  031047  030066          .ASCII  <CRLF>/'260G' FOR COMPATABILITY PASS 1: WRITE/
017600  023600  033462  043460          .ASCIZ  <CRLF>/'270G' FOR COMPATABILITY PASS 2: READ/
017647     040  042524  052123  DROP:   .ASCIZ  / TESTING DROPPED/
017670  051040  047125  044516  RUN:    .ASCIZ  / RUNNING/
017701     040  045517     000  OK:     .ASCIZ  / OK/
017705     040  051124  000113  TRK:    .ASCIZ  / TRK/
017712  042040  052101  020101  PATT:   .ASCIZ  / DATA PATT/
017725     040  040522  042116  RANDOM: .ASCIZ  / RANDOM SEEKS/
017743     200  042515  020115  MEMORY: .ASCIZ  <CRLF>/MEM TESTS/
017756  042040  047117  000105  DUN:    .ASCIZ  / DONE/
017764  005200  047105  020104  ENDPAS: .ASCIZ  <CRLF><LF>/END PASS #/
020001     011  047524  040524  MSG1:   .ASCIZ  /       TOTAL ERRORS:          /
020030  004600  004411  047524  MSG2:   .ASCIZ  <CRLF>/                    TOTAL SOFT ERRORS:     /
020064  005200  004411  052011  MSG3:   .ASCIZ  <CRLF><LF>/                       TOTAL ERRORS THIS PASS:/
020121     200  004411  051411  MSG4:   .ASCIZ  <CRLF>/                    SOFT  ERRORS THIS PASS:/
020155     200  047503  054520  COPMSG: .ASCII  <CRLF>/COPY DX0 TO DX1/
020175     200  047111  042523          .ASCIZ  <CRLF>/INSERT SCRATCH DISK IN DX1, TYPE 'P' TO PROCEED/
020256  042200  047117  027105  COPDUN: .ASCIZ  <CRLF>/DONE...TYPE 'P' TO DO AGAIN OR '240G' FOR NORMAL TESTING/
020350  005200  047503  050115  COMPAT: .ASCII  <CRLF><LF>/COMPATABILITY PASS 1 (WRITE) DONE/
020413     200  047504  023440          .ASCIZ  <CRLF>/DO 'P' FOR PASS 2 (READ)/
020445     200  042504  046526  DEVM:   .ASCIZ  <CRLF>/DEVM - /
020456  042515  047515  054522  EM1:    .ASCIZ  /MEMORY TIMEOUT/
```

```
020475     115  046505  042040  EM2:    .ASCIZ  /MEM DATA COMP ERR/
020517     123  047131  020103  EM3:    .ASCIZ  /SYNC COMM DID NOT RECEIVE SYNC CHAR/
020563     120  044522  052116  EM4:    .ASCIZ  /PRINTER STATUS ERR/
020606  042524  046522  021440  EM5:    .ASCIZ  /TERM #1 DATA COMP ERR/
020634  042524  046522  021440  EM6:    .ASCIZ  /TERM #2 DATA COMP ERR/
020662  042524  046522  021440  EM7:    .ASCIZ  /TERM #3 DATA COMP ERR/
020710  051501  047131  020103  EM8:    .ASCIZ  /ASYNC COMM DATA COMP ERR/
020741     123  047131  020103  EM9:    .ASCIZ  /SYNC COMM DATA COMP ERR/
020771     125  042516  050130  EM10:   .ASCIZ  /UNEXP DISK INTERRUPT/
021016  054104  020060  051127  EM11:   .ASCIZ  /DX0 WRITE ERR/
021034  054104  020060  040510  EM13:   .ASCIZ  /DX0 HARD ERR - READ SECT/
021065     104  030130  051440  EM14:   .ASCIZ  /DX0 SOFT ERR - READ SECT/
021116  054104  020060  040510  EM15:   .ASCIZ  /DX0 HARD ERR - DATA COMP/
021147     104  030130  051440  EM16:   .ASCIZ  /DX0 SOFT ERR - DATA COMP/
021200  054104  020061  051127  EM17:   .ASCIZ  /DX1 WRITE ERR/
021216  054104  020061  040510  EM19:   .ASCIZ  /DX1 HARD ERR - READ SECT/
021247     104  030530  051440  EM20:   .ASCIZ  /DX1 SOFT ERR - READ SECT/
021300  054104  020061  040510  EM21:   .ASCIZ  /DX1 HARD ERR - DATA COMP/
021331     104  030530  051440  EM22:   .ASCIZ  /DX1 SOFT ERR - DATA COMP/
021362  046103  020113  052510  EM23:   .ASCIZ  /CLK HUNG/
021373     120  044522  052116  EM24:   .ASCIZ  /PRINTER HUNG/
021410  042524  046522  021440  EM25:   .ASCIZ  /TERM #1 HUNG/
021425     124  051105  020115  EM26:   .ASCIZ  /TERM #2 HUNG/
021442  042524  046522  021440  EM27:   .ASCIZ  /TERM #3 HUNG/
021457     123  047131  020103  EM28:   .ASCIZ  /SYNC COMM HUNG/
021476  051501  047131  020103  EM29:   .ASCIZ  /ASYNC COMM HUNG/
021516  054104  020060  052510  EM30:   .ASCIZ  /DX0 HUNG/
021527     104  030530  044040  EM31:   .ASCIZ  /DX1 HUNG/
021540  054523  041516  041440  EM32:   .ASCIZ  /SYNC COMM - DATA NOT AVAIL NOT SET/
021603     123  047131  020103  EM33:   .ASCIZ  /SYNC COMM - RECVR ACTIVE NOT SET/
021644  054523  041516  041440  EM34:   .ASCIZ  /SYNC COMM - RECVR DONE NOT SET/
021703     104  030130  023440  EM35:   .ASCIZ  /DX0 'DEL DATA' BIT 5 NOT SET IN RXES/
021750  054104  020060  047111  EM36:   .ASCIZ  /DX0 INIT CMD DID NOT READ TRK 1, SEC 1/
022017     104  030130  051040  EM37:   .ASCIZ  /DX0 RESTORE CMD ERR/
022043     104  030130  023440  EM38:   .ASCIZ  /DX0 'INIT DONE' BIT 0 NOT SET IN RXES/
022111     104  030130  051040  EM39:   .ASCIZ  /DX0 READ STATUS CMD ERR/
022141     104  030130  044440  EM40:   .ASCIZ  /DX0 INV ADDR BIT 1 NOT SET IN RXES/
022204  054104  020060  051105  EM41:   .ASCIZ  /DX0 ERR BIT NOT SET IN RXCS/
022240  054104  020060  047111  EM42:   .ASCIZ  /DX0 INIT CMD ERR/
022261     104  030130  051440  EM43:   .ASCIZ  /DX0 SEEK ERR - WRITE SECT/
022313     104  030130  051440  EM45:   .ASCIZ  /DX0 SEEK ERR - READ SECT/
022344  054104  020061  042523  EM47:   .ASCIZ  /DX1 SEEK ERR - WRITE SECT/
022376  054104  020061  042523  EM49:   .ASCIZ  /DX1 SEEK ERR - READ SECT/
022427     104  030530  051040  EM51:   .ASCIZ  /DX1 RESTORE CMD ERR/
022453     104  051511  020113  EM52:   .ASCIZ  /DISK 'DONE' NOT SET/
022477     106  046111  020114  EM53:   .ASCIZ  /FILL & EMPTY BUFF TEST... DATA MISCOMP/
022546  040504  040524  046440  EM54:   .ASCIZ  /DATA MISCOMP/
022563     122  050105  040514  EM55:   .ASCIZ  /REPLACE DX0...10 BAD SECTORS/
022620  042522  046120  041501  EM56:   .ASCIZ  /REPLACE DX1...10 BAD SECTORS/
022655     104  030130  044040  EM57:   .ASCIZ  /DX0 HARD CRC ERR/
022676  054104  020060  040504  EM58:   .ASCIZ  /DX0 DATA CRC ERR/
022717     104  030130  050040  EM59:   .ASCIZ  /DX0 PREV CRC ERR WITH DATA OK/
022755     104  030530  044040  EM60:   .ASCIZ  /DX1 HARD CRC ERR/
022776  054104  020061  040504  EM61:   .ASCIZ  /DX1 DATA CRC ERR/
023017     104  030530  050040  EM62:   .ASCIZ  /DX1 PREV CRC ERR WITH DATA OK/
```

```
023055     105  051122  051117  DH1:    .ASCIZ  /ERROR #       ERR PC/
023074  051105  047522  020122  DH2:    .ASCIZ  /ERROR #       ERR PC  ADDR/
023120  051105  047522  020122  DH3:    .ASCIZ  /ERROR #       ERR PC  ADDR    EXPECT  RECVD/
023162  051105  047522  020122  DH4:    .ASCIZ  /ERROR #       ERR PC  PR STATUS/
023213     105  051122  051117  DH5:    .ASCIZ  /ERROR #       ERR PC  EXPECT  RECVD/
023250  051105  047522  020122  DH6:    .ASCIZ  /ERROR #       DXTST#  ERR PC  RXCS    RXES    RXSA/
023315     105  051122  051117  DH7:    .ASCIZ  /ERROR #       DXTST#  ERR PC  RXCS    RXES    TRACK   SECTOR  # RETRIE
023404  054104  051524  020124  DH8:    .ASCIZ  /DXTST #       ERR PC  RXCS    RXES    RXSA    EXPECT  RECVD  # RETRIES

        023474                  .EVEN

023474  025764  001116  000000  DT1:    .WORD   ERRNUM,$ERRPC,0
023502  025764  001116  004462  DT2:    .WORD   ERRNUM,$ERRPC,ADDR,0
023512  025764  001116  004462  DT3:    .WORD   ERRNUM,$ERRPC,ADDR,PAT,MEMHLD,0
023526  025764  001116  012302  DT4:    .WORD   ERRNUM,$ERRPC,SPRS,0
023536  025764  001116  012400  DT5:    .WORD   ERRNUM,$ERRPC,T1CHR,T1HLD,0
023550  025764  001116  012500  DT6:    .WORD   ERRNUM,$ERRPC,T2CHR,T2HLD,0
023562  025764  001116  012600  DT7:    .WORD   ERRNUM,$ERRPC,T3CHR,T3HLD,0
023574  025764  001116  012700  DT8:    .WORD   ERRNUM,$ERRPC,COMCHR,COMHLD,0
023606  025764  003120  001116  DT9:    .WORD   ERRNUM,DXTST,$ERRPC,SRXCS,SRXES,SRXSA,0
023624  025764  003120  001116  DT10:   .WORD   ERRNUM,DXTST,$ERRPC,SRXCS,SRXES,D0TRK,D0SEC,D0ERR,0
023646  003120  001116  177170  DT11:   .WORD   DXTST,$ERRPC,RXCS,RXES,RXSA,D0EXP,D0REC,D0CERR,0
023670  025764  003120  001116  DT12:   .WORD   ERRNUM,DXTST,$ERRPC,SRXCS,SRXES,D1TRK,D1SEC,D1ERR,0
023712  003120  001116  177170  DT13:   .WORD   DXTST,$ERRPC,RXCS,RXES,RXSA,D1EXP,D1REC,D1CERR,0
023734  025764  003120  001116  DT14:   .WORD   ERRNUM,DXTST,$ERRPC,RXCS,RXES,RXSA,0
023752  025764  001116  006252  DT15:   .WORD   ERRNUM,$ERRPC,EXP0,EBHLD,0
023764  025764  001116  006254  DT16:   .WORD   ERRNUM,$ERRPC,EXP1,EBHLD,0

023776  000000                  DF:     .WORD   0               ;OCTAL FORMAT FOR ALL
024000  000000                          .WORD   0
024002  000000                          .WORD   0

024004  000000                  DF1:    .WORD   0               ;OCTAL,OCTAL
024006  000000                          .WORD   0               ;OCTAL,OCTAL
024010  000400                          .WORD   400             ;DECIMAL,OCTAL
024012  000401                          .WORD   401             ;DECIMAL,DECIMAL

024014  000000                  DF2:    .WORD   0               ;OCTAL,OCTAL
024016  000000                          .WORD   0               ;OCTAL,OCTAL
024020  000000                          .WORD   0               ;OCTAL,OCTAL
024022  000400                          .WORD   400             ;DECIMAL,OCTAL

                                .LIST   BEX
```

```
                                        .SBTTL  TYPE ROUTINE
                                        ;;*********************************************************************
                                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                        ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                        ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                        ;*
                                        ;*CALL:
                                        ;*1) USING A TRAP INSTRUCTION
                                        ;*       TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                        ;*OR
                                        ;*       TYPE
                                        ;*       MESADR
                                        ;*

024024  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
024030  100002                          BPL     1$              ;;BR IF YES
024032  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
024034  000430                          BR      3$              ;;LEAVE
024036  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
024040  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
024044  122737  000001  001210          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
024052  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
024054  132737  000100  001211          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
024062  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
024064  010037  024074                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
024070  004737  025156                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
024074  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
024076  132737  000040  001211  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
024104  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
024106  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
024110  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
024112  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
024114  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
024116  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
024122  000002                          RTI                     ;;RETURN
024124  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
024130  001430                          BEQ     8$
024132  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
024136  001006                          BNE     5$
024140  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
024142  104401                          TYPE                    ;;TYPE A CR AND LF
024144  001165                          $CRLF
024146  105037  024302                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
024152  000755                          BR      2$              ;;GET NEXT CHARACTER
024154  004737  024236          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
024160  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
024164  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
024166  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
024172  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
024176  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
024200  004737  024236                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
024204  105337  024302                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
```

```
 4119  024210  000770                        BR      7$              ;;LOOP
 4120
 4121                                  ;HORIZONTAL TAB PROCESSOR
 4122
 4123  024212  112716  000040          8$:   MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 4124  024216  004737  024236          9$:   JSR     PC,$TYPEC       ;;TYPE A SPACE
 4125  024222  132737  000007  024302        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 4126  024230  001372                        BNE     9$              ;;TAB STOP
 4127  024232  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
 4128  024234  000724                        BR      2$              ;;GET NEXT CHARACTER
 4129  024236  105777  154706        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 4130  024242  100375                        BPL     $TYPEC
 4131  024244  116677  000002  154700        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 4132  024252  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 4133  024260  001003                        BNE     1$              ;;BRANCH IF NO
 4134  024262  105037  024302                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 4135  024266  000406                        BR      $TYPEX          ;;EXIT
 4136  024270  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 4137  024276  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
 4138  024300  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
 4139  024302  000000        $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 4140  024304  000207        $TYPEX: RTS     PC
 4141
```

```
                                        .SBTTL  TTY INPUT ROUTINE

                                        ;;*******************************************************************
                                        .ENABL  LSB

                                        ;;*******************************************************************
                                        ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                        ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                        ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                        ;*WHEN OPERATING IN TTY FLAG MODE.
024306  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
024314  001114                        BNE     15$             ;;BRANCH IF NO
024316  105777  154622                TSTB    @$TKS           ;;CHAR THERE?
024322  100111                        BPL     15$             ;;IF NO, DON'T WAIT AROUND
024324  117746  154616                MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
024330  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
024334  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
024340  001102                        BNE     15$             ;;NO, RETURN TO USER
024342  123727  001134  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
024350  001476                        BEQ     15$             ;;BRANCH IF YES

024352  104401  025120                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
024356  104401  025125        $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
024362  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
024366  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
024370  104401  025136                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
024374  005046                19$:    CLR     -(SP)           ;;CLEAR COUNTER
024376  005046                        CLR     -(SP)           ;;THE NEW SWR
024400  105777  154540        7$:     TSTB    @$TKS           ;;CHAR THERE?
024404  100375                        BPL     7$              ;;IF NOT TRY AGAIN

024406  117746  154534                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
024412  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII

024416  021627  000003                CMP     (SP),#3         ;;IS IT A CONTROL-C?
024422  001015                        BNE     9$              ;;BRANCH IF NOT
024424  104401  025106                TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
024430  062706  000006                ADD     #6,SP           ;;CLEAN UP STACK
024434  123727  001135  000001        CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
024442  001003                        BNE     8$              ;;BRANCH IF NO
024444  012777  000100  154472        MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
024452  000137  011116        8$:     JMP     GT$DEV          ;;CONTROL-C RESTART

024456  021627  000025        9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
024462  001005                        BNE     10$             ;;BRANCH IF NOT
024464  104401  025113                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
024470  062706  000006        20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
024474  000737                        BR      19$             ;;LET'S TRY IT AGAIN

024476  021627  000015        10$:    CMP     (SP),#15        ;;IS IT A <CR>?
024502  001022                        BNE     16$             ;;BRANCH IF NO
024504  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
024510  001403                        BEQ     1?$             ;;BRANCH IF YES
024512  016677  000002  154420        MOV     2(SP),@SWR      ;;SAVE NEW SWR
```

```
4198  024520  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
4199  024524  104401  001165          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
4200  024530  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
4201  024536  001003                          BNE     15$             ;;BRANCH IF NOT
4202  024540  012777  000100  154376          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
4203  024546  000002                  15$:    RTI                     ;;RETURN
4204  024550  004737  024236          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
4205  024554  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
4206  024560  002420                          BLT     18$             ;;BRANCH IF YES
4207  024562  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
4208  024566  003015                          BGT     18$             ;;BRANCH IF YES
4209  024570  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
4210  024574  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
4211  024600  001403                          BEQ     17$             ;;BRANCH IF YES
4212  024602  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
4213  024604  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
4214  024606  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
4215  024610  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
4216  024614  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
4217  024620  000667                          BR      7$              ;;GET THE NEXT ONE
4218  024622  104401  001164          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
4219  024626  000720                          BR      20$             ;;SIMULATE CONTROL-U
4220                                  .DSABL  LSB
4221
4222
4223                                  ;;**********************************************************************
4224                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
4225                                  ;*CALL:
4226                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
4227                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
4228                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
4229                                  ;
4230
4231  024630  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
4232  024632  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
4233  024640  105777  154300          1$:     TSTB    @$TKS           ;;WAIT FOR
4234  024644  100375                          BPL     1$              ;;A CHARACTER
4235  024646  117766  154274  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
4236  024654  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
4237  024662  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
4238  024670  001013                          BNE     3$              ;;BRANCH IF NO
4239  024672  105777  154246          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
4240  024676  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
4241  024700  117746  154242                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
4242  024704  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
4243  024710  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
4244  024714  001366                          BNE     2$              ;;IF NOT DISCARD IT
4245  024716  000750                          BR      1$              ;;YES, RESUME
4246  024720  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
4247  024726  002407                          BLT     4$              ;;BRANCH IF YES
4248  024730  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
4249  024736  003003                          BGT     4$              ;;BRANCH IF YES
4250  024740  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
4251  024746  000002                  4$:     RTI                     ;;GO BACK TO USER
4252                                  ;;*************** **********************************************************
4253                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
```

```
                                        ;*CALL:
                                        ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                        ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                        ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

024750  010346                 $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
024752  012703  025076           1$:   MOV     #$TTYIN,R3              ;;GET ADDRESS
024756  022703  025106           2$:   CMP     #$TTYIN+8.,R3           ;;BUFFER FULL?
024762  101415                        BLOS    4$                      ;;BR IF YES
024764  104410                        RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
024766  112613                        MOVB    (SP)+,(R3)              ;;GET CHARACTER
024770  122713  000003                CMPB    #3,(R3)                 ;;IS IT A CONTROL-C?
024774  001005                        BNE     10$                     ;;BRANCH IF NO
024776  104401  025106                TYPE    ,$CNTLC                 ;;TYPE A CONTROL-C (^C)
025002  012603                        MOV     (SP)+,R3                ;;RESTORE R3
025004  000137  011116                JMP     GT$DEV                  ;;GOTO CONTROL-C RESTART
025010  122713  000177           0$:   CMPB    #177,(R3)               ;;IS IT A RUBOUT
025014  001003                        BNE     3$                      ;;SKIP IF NOT
025016  104401  001164           4$:   TYPE    ,$QUES                  ;;TYPE A '?'
025022  000753                        BR      1$                      ;;CLEAR THE BUFFER AND LOOP
025024  111337  025074           3$:   MOVB    (R3),9$                 ;;ECHO THE CHARACTER
025030  104401  025074                TYPE    ,9$
025034  122723  000015                CMPB    #15,(R3)+               ;;CHECK FOR RETURN
025040  001346                        BNE     2$                      ;;LOOP IF NOT RETURN
025042  105063  177777                CLRB    -1(R3)                  ;;CLEAR RETURN (THE 15)
025046  104401  001166                TYPE    ,$LF                    ;;TYPE A LINE FEED
025052  012603                        MOV     (SP)+,R3                ;;RESTORE R3
025054  011646                        MOV     (SP),-(SP)              ;;ADJUST THE STACK AND PUT ADDRESS OF THE
025056  016666  000004  000002        MOV     4(SP),2(SP)             ;;      FIRST ASCII CHARACTER ON IT
025064  012766  025076  000004        MOV     #$TTYIN,4(SP)
025072  000002                        RTI                             ;;RETURN
025074     000                  9$:   .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
025075     000                        .BYTE   0                       ;;TERMINATOR
025076  000010              $TTYIN:   .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
025106  041536  005015  000 $CNTLC:   .ASCIZ  /^C/<15><12>            ;;CONTROL ''C''
025113     136  006525  000012$CNTLU: .ASCIZ  /^U/<15><12>            ;;CONTROL ''U''
025120  043536  005015  000 $CNTLG:   .ASCIZ  /^G/<15><12>            ;;CONTROL ''G''
025125     015  051412  051127 $MSWR:  .ASCIZ  <15><12>/SWR - /
025132  036440  000040
025136  020040  042516  020127 $MNEW:  .ASCIZ  /  NEW = /
025144  020075     000
025150                          .EVEN
```

```
                                .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;*****************************************************************
025150  112737  000001  025414  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
025156  112737  000001  025412  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
025164  000403                         BR      $ATYC
025166  112737  000001  025414  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
025174                          $ATYC:
025174  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
025176  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
025200  105737  025412                 TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
025204  001450                         BEQ     5$              ;;IF NOT:  BR
025206  122737  000001  001210         CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
025214  001031                         BNE     3$              ;;IF NOT:  BR
025216  132737  000100  001211         BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
025224  001425                         BEQ     3$              ;;IF NOT:  BR
025226  017600  000004                 MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
025232  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
025240  005737  001170          1$:    TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
025244  001375                         BNE     1$              ;;IF NOT:  WAIT
025246  010037  001204                 MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
025252  105720                  2$:    TSTB    (R0)+           ;;FIND END OF MESSAGE
025254  001376                         BNE     2$
025256  163700  001204                 SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
025262  006200                         ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
025264  010037  001206                 MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
025270  012737  000004  001170         MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
025276  000413                         BR      5$
025300  017637  000004  025324  3$:    MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
025306  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
025314  013746  177776                 MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
025320  004737  024024                 JSR     PC,$TYPE        ;;CALL TYPE MACRO
025324  000000                  4$:    .WORD   0
025326                          5$:
025326  105737  025414          10$:   TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
025332  001416                         BEQ     12$             ;;IF NOT:  BR
025334  005737  001210                 TST     $ENV            ;;RUNNING UNDER APT?
025340  001413                         BEQ     12$             ;;IF NOT:  BR
025342  005737  001170          11$:   TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
025346  001375                         BNE     11$             ;;IF NOT:  WAIT
025350  017637  000004  001172         MOV     @4(SP),$FATAL   ;;GET ERROR #
025356  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
025364  005237  001170                 INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
025370  105037  025414          12$:   CLRB    $FFLG           ;;CLEAR FATAL FLAG
025374  105037  025413                 CLRB    $LFLG           ;;CLEAR LOG FLAG
025400  105037  025412                 CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
025404  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
025406  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
025410  000207                         RTS     PC              ;;RETURN
025412     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
025413     000          $LFLG:  .BYTE   0               ;;LOG FLAG
025414     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
        025416                          .EVEN
        000200          APTSIZE=200
        000001          APTENV=001
        000100          APTSPOOL=100
```

```
 4352            000040                  APTCSUP=040
```

```
                                        .SBTTL  ERROR HANDLER ROUTINE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                        ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                        ;*AND GO TO MYTYPE ON ERROR
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW15=1         HALT ON ERROR
                                        ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                        ;*SW10=1         BELL ON ERROR
                                        ;*CALL
                                        ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

025416                                  $ERROR:
025416  104407                                  CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
025420  105237  001103                  7$:     INCB    $ERFLG                  ;;SET THE ERROR FLAG
025424  001775                                  BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
025426  013777  001102  153506                  MOV     $TSTNM,@DISPLAY         ;;DISPLAY TEST NUMBER AND ERROR FLAG
025434  032777  002000  153476                  BIT     #BIT10,@SWR             ;;BELL ON ERROR?
025442  001402                                  BEQ     1$                      ;;NO - SKIP
025444  104401  001160                          TYPE    ,$BELL                  ;;RING BELL
025450  005237  001112                  1$:     INC     $ERTTL                  ;;COUNT THE NUMBER OF ERRORS
025454  011637  001116                          MOV     (SP),$ERRPC             ;;GET ADDRESS OF ERROR INSTRUCTION
025460  162737  000002  001116                  SUB     #2,$ERRPC
025466  117737  153424  001114                  MOVB    @$ERRPC,$ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
025474  032777  020000  15343(                  BIT     #BIT13,@SWR             ;;SKIP TYPEOUT IF SET
025502  001004                                  BNE     20$                     ;;SKIP TYPEOUTS
025504  004737  025732                          JSR     PC,MYTYPE               ;;GO TO USER ERROR ROUTINE
025510  104401  001165                          TYPE    ,$CRLF
025514                                  20$:
025514  122737  000001  001210                  CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
025522  001007                                  BNE     2$                      ;;NO,SKIP APT ERROR REPORT
025524  113737  001114  025536                  MOVB    $ITEMB,21$              ;;SET ITEM NUMBER AS ERROR NUMBER
025532  004737  025166                          JSR     PC,$ATY4                ;;REPORT FATAL ERROR TO APT
025536     000                  21$:    .BYTE   0
025537     000                          .BYTE   0
025540  000777                  22$:    BR      22$                     ;;APT ERROR LOOP
025542  005777  153372          2$:     TST     @SWR                    ;;HALT ON ERROR
025546  100002                          BPL     3$                      ;;SKIP IF CONTINUE
025550  000000                          HALT                            ;;HALT ON ERROR!
025552  104407                          CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
025554                          3$:
025554  000002                          RTI                             ;;RETURN
```

```
 4396                                   .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
 4397
 4398                                   ;;*********************************************************************
 4399                                   ;*THIS ROUTINE USES THE ''ITEM CONTROL BYTE'' ($ITEMB) TO DETERMINE WHICH
 4400                                   ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE 'ERROR TABLE'' ($ERRTB),
 4401                                   ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
 4402
 4403  025556                           $ERRTYP:
 4404  025556  104401  001165                   TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & 'LINE FEED''
 4405  025562  010046                           MOV     R0,-(SP)        ;;SAVE R0
 4406  025564  005000                           CLR     R0              ;;PICKUP THE ITEM INDEX
 4407  025566  153700  001114                   BISB    @#$ITEMB,R0
 4408  025572  001004                           BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
 4409                                                                   ;;TYPE THE PC OF THE ERROR
 4410  025574  013746  001116                   MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
 4411                                                                   ;;ERROR ADDRESS
 4412  025600  104402                           TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 4413  025602  000445                           BR      10$             ;;GET OUT
 4414  025604  005300                   1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
 4415  025606  006300                           ASL     R0              ;;      WORK FOR THE ERROR TABLE
 4416  025610  006300                           ASL     R0
 4417  025612  006300                           ASL     R0
 4418  025614  062700  001250                   ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
 4419  025620  012037  025630                   MOV     (R0)+,2$        ;;PICKUP ''ERROR MESSAGE'' POINTER
 4420  025624  001404                           BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
 4421  025626  104401                           TYPE                    ;;TYPE THE ''ERROR MESSAGE''
 4422  025630  000000                   2$:     .WORD   0               ;;''ERROR MESSAGE'' POINTER GOES HERE
 4423  025632  104401  001165                   TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & 'LINE FEED''
 4424  025636  012037  025646           3$:     MOV     (R0)+,4$        ;;PICKUP 'DATA HEADER'' POINTER
 4425  025642  001404                           BEQ     5$              ;;SKIP TYPEOUT IF 0
 4426  025644  104401                           TYPE                    ;;TYPE THE 'DATA HEADER''
 4427  025646  000000                   4$:     .WORD   0               ;;'DATA HEADER'' POINTER GOES HERE
 4428  025650  104401  001165                   TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & 'LINE FEED''
 4429  025654  010146                   5$:     MOV     R1,-(SP)        ;;SAVE R1
 4430  025656  012001                           MOV     (R0)+,R1        ;;PICKUP 'DATA TABLE'' POINTER
 4431  025660  001415                           BEQ     9$              ;;BR IF NO DATA TO BE TYPED
 4432  025662  012000                           MOV     (R0)+,R0        ;;PICKUP 'DATA FORMAT'' POINTER
 4433  025664  105720                   6$:     TSTB    (R0)+           ;;''OCTAL'' OR 'DECIMAL''
 4434  025666  001003                           BNE     7$              ;;BR IF DECIMAL
 4435  025670  013146                           MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
 4436  025672  104402                           TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 4437  025674  000402                           BR      8$
 4438  025676                           7$:
 4439  025676  013146                           MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
 4440  025700  104405                           TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 4441  025702  005711                   8$:     TST     (R1)            ;;IS THERE ANOTHER NUMBER?
 4442  025704  001403                           BEQ     9$              ;;BR IF NO
 4443  025706  104401  025726                   TYPE    ,11$            ;;TYPE TWO(2) SPACES
 4444  025712  000764                           BR      6$              ;;LOOP
 4445
 4446  025714  012601                   9$:     MOV     (SP)+,R1        ;;RESTORE R1
 4447  025716  012600                   10$:    MOV     (SP)+,R0        ;;RESTORE R0
 4448  025720  104401  001165                   TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & 'LINE FEED''
 4449  025724  000207                           RTS     PC              ;;RETURN
 4450  025726  020040      000          11$:    .ASCIZ  /  /            ;;TWO(2) SPACES
 4451          025732                           .EVEN
```

```
 4452
 4453
 4454
 4455  025732  113737  001114  025764  MYTYPE: MOVB    $ITEMB,ERRNUM   ;FOR ERROR TYPEOUT
 4456  025740  013737  013210  001174          MOV     SRXSA,$TESTN    ;FOR APT
 4457                                                                  ;APT PRINTS $TESTN & $FATAL IN THAT ORDER.
 4458                                                                  ;$TESTN WILL BE FLOPPY RXSA
 4459                                                                  ;$FATAL IS ERR #
 4460  025746  004737  025556                  JSR     PC,$ERRTYP      ;TYPE ERR MSG
 4461  025752  005237  025766                  INC     TERR
 4462  025756  005237  025772                  INC     PERR
 4463  025762  000207                          RTS     PC
 4464
 4465  025764  000000                  ERRNUM: 0                       ;FOR ERR TYPEOUT
 4466
 4467  025766  000000                  TERR:   0                       ;TOTAL ERROR COUNT
 4468  025770  000000                   SERR:  0                       ;TOTAL SOFT ERR COUNT
 4469  025772  000000                  PERR:   0                       ;PASS ERR COUNT
 4470  025774  000000                  PSERR:  0                       ;PASS SOFT ERR COUNT... ALL FOR EOP TYPEOUT
```

```
                                        .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                        ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                        ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                        ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                        ;*REPLACED WITH SPACES.
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                        ;*      TYPDS                   ;;GO TO THE ROUTINE

025776                                  $TYPDS:
025776  010046                                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
026000  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
026002  010246                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
026004  010346                                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
026006  010546                                 MOV     R5,-(SP)        ;;PUSH R5 ON STACK
026010  012746  020200                         MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
026014  016605  000020                         MOV     20(SP),R5       ;;GET THE INPUT NUMBER
026020  100004                                 BPL     1$              ;;BR IF INPUT IS POS.
026022  005405                                 NEG     R5              ;;MAKE THE BINARY NUMBER POS.
026024  112766  000055  000001                 MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
026032  005000                         1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
026034  012703  026212                         MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
026040  112723  000040                         MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
026044  005002                         2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
026046  016001  026202                         MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
026052  160105                         3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
026054  002402                                 BLT     4$              ;;BR IF DONE
026056  005202                                 INC     R2              ;;INCREASE THE BCD DIGIT BY 1
026060  000774                                 BR      3$
026062  060105                         4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
026064  005702                                 TST     R2              ;;CHECK IF BCD DIGIT=0
026066  001002                                 BNE     5$              ;;FALL THROUGH IF 0
026070  105716                                 TSTB    (SP)            ;;STILL DOING LEADING 0'S?
026072  100407                                 BMI     7$              ;;BR IF YES
026074  106316                         5$:     ASLB    (SP)            ;;MSD?
026076  103003                                 BCC     6$              ;;BR IF NO
026100  116663  000001  177777                 MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
026106  052702  000060                 6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
026112  052702  000040                 7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
026116  110223                                 MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
026120  005720                                 TST     (R0)+           ;;JUST INCREMENTING
026122  020027  000010                         CMP     R0,#10          ;;CHECK THE TABLE INDEX
026126  002746                                 BLT     2$              ;;GO DO THE NEXT DIGIT
026130  003002                                 BGT     8$              ;;GO TO EXIT
026132  010502                                 MOV     R5,R2           ;;GET THE LSD
026134  000764                                 BR      6$              ;;GO CHANGE TO ASCII
026136  105726                         8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
026140  100003                                 BPL     9$              ;;BR IF NO
026142  116663  177777  177776                 MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
026150  105013                         9$:     CLRB    (R3)            ;;SET THE TERMINATOR
026152  012605                                 MOV     (SP)+,R5        ;;POP STACK INTO R5
026154  012603                                 MOV     (SP)+,R3        ;;POP STACK INTO R3
026156  012602                                 MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
 4527  026160  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 4528  026162  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 4529  026164  104401  026212                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
 4530  026170  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 4531  026176  012616                        MOV     (SP)+,(SP)
 4532  026200  000002                        RTI                     ;;RETURN TO USER
 4533  026202  023420                $DTBL:  10000.
 4534  026204  001750                        1000.
 4535  026206  000144                        100.
 4536  026210  000012                        10.
 4537  026212  000004                $DBLK:  .BLKW   4
```

```
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
                                        ;;*********************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                        ;*      .BYTE   M               ;;M=1 OR 0
                                        ;*                                      ;;1=TYPE LEADING ZEROS
                                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                        ;*
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT

026222  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
026226  116637  000001  026445         MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
026234  112637  026447                 MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
026240  062716  000002                 ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
026244  000406                         BR      $TYPON
026246  112737  000001  026445 $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
026254  112737  000006  026447         MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
026262  112737  000005  026444 $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
026270  010346                         MOV     R3,-(SP)        ;;SAVE R3
026272  010446                         MOV     R4,-(SP)        ;;SAVE R4
026274  010546                         MOV     R5,-(SP)        ;;SAVE R5
026276  113704  026447                 MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
026302  005404                         NEG     R4
026304  062704  000006                 ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
026310  110437  026446                 MOVB    R4,$OMODE       ;;SAVE IT FOR USE
026314  113704  026445                 MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
026320  016605  000012                 MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
026324  005003                         CLR     R3              ;;CLEAR THE OUTPUT WORD
026326  006105                 1$:     ROL     R5              ;;ROTATE MSB INTO "C"
026330  000404                         BR      3$              ;;GO DO MSB
026332  006105                 2$:     ROL     R5              ;;FORM THIS DIGIT
026334  006105                         ROL     R5
026336  006105                         ROL     R5
026340  010503                         MOV     R5,R3
026342  006103                 3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
026344  105337  026446                 DECB    $OMODE          ;;TYPE THIS DIGIT?
026350  100016                         BPL     7$              ;;BR IF NO
026352  042703  177770                 BIC     #177770,R3      ;;GET RID OF JUNK
026356  001002                         BNE     4$              ;;TEST FOR 0
026360  005704                         TST     R4              ;;SUPPRESS THIS 0?
026362  001403                         BEQ     5$              ;;BR IF YES
```

```
 4594  026364  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 4595  026366  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 4596  026372  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 4597  026376  110337  026442                  MOVB    R3,8$           ;;SAVE FOR TYPING
 4598  026402  104401  026442                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
 4599  026406  105337  026444          7$:     DECB    $OCNT           ;;COUNT BY 1
 4600  026412  003347                          BGT     2$              ;;BR IF MORE TO DO
 4601  026414  002402                          BLT     6$              ;;BR IF DONE
 4602  026416  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 4603  026420  000744                          BR      2$              ;;GO DO THE LAST DIGIT
 4604  026422  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
 4605  026424  012604                          MOV     (SP)+,R4        ;;RESTORE R4
 4606  026426  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 4607  026430  016666  000002  000004         MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 4608  026436  012616                          MOV     (SP)+,(SP)
 4609  026440  000002                          RTI                     ;;RETURN
 4610  026442     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 4611  026443     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 4612  026444     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 4613  026445     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 4614  026446  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                        .SBTTL  TRAP DECODER

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
                                        ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                        ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                        ;*GO TO THAT ROUTINE.

026450  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
026452  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
026456  005740                          TST     -(R0)           ;;BACKUP BY 2
026460  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
026462  006300                          ASL     R0              ;;POSITION FOR INDEXING
026464  016000  026504                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
026470  000200                          RTS     R0              ;;GO TO ROUTINE


                                        ;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO

026472  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
026474  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
026502  000002                          RTI                     ;;RESTORE THE PSW

                                        .SBTTL  TRAP TABLE

                                        ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                        ;*BY THE 'TRAP' INSTRUCTION.

                                        ;       ROUTINE
                                        ;       -------
026504  026472                  $TRPAD: .WORD   $TRAP2
026506  024024                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
026510  026246                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
026512  026222                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
026514  026262                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
026516  025776                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

026520  024356                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

026522  024306                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
026524  024630                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
026526  024750                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
026530  011116                          GT$DEV  ;;CALL=GTDEVM   TRAP+12(104412) ^C WILL OPEN UP $DEVM

026532                          LSTAD:
                                        ;MAX ADDR FOR 8K WITH APT = 37400
                                        ;MAX ADDR FOR 8K NO APT BUT TO SAVE ABS LOADER = 37500
        003174                  .END    START
```

```
ABASE = 176500            669#   1151    1192
ACDW1 = 000000           1151
ACDW2 = 000000           1151
ACOM    017170           1907    2594    4062#
ACPUOP= 000000           1151    1166
ADDR    004462           1673*   1687*   1692*   1720*   1725*   1753#   4062
ADDW0 = 000000           1151
ADDW1 = 000000           1151
ADDW10= 000000           1151
ADDW11= 000000           1151
ADDW12= 000000           1151
ADDW13= 000000           1151
ADDW14= 000000           1151
ADDW15= 000000           1151
ADDW2 = 000000           1151
ADDW3 = 000000           1151
ADDW4 = 000000           1151
ADDW5 = 000000           1151
ADDW6 = 000000           1151
ADDW7 = 000000           1151
ADDW8 = 000000           1151
ADDW9 = 000000           1151
ADEVCT= 000000           1151    1157
ADEVM = 000017            669#   1151    1193
AENV  = 000000           1151    1162
AENVM = 000000           1151    1163
AFATAL= 000000           1151    1154
AMADR1= 000000           1151    1179
AMADR2= 000000           1151    1183
AMADR3= 000000           1151    1186
AMADR4= 000000           1151    1189
AMAMS1= 000000           1151    1173
AMAMS2= 000000           1151    1181
AMAMS3= 000000           1151    1184
AMAMS4= 000000           1151    1187
AMOD  = 030000            883#   1884    1886    2612
AMRB  = 176612            842#   1895    3105
AMRC  = 176610            841#   1596*   1597*   1627*   1887*   1896*   1920*   1921*   2613*   2614*
AMRSRV  012622           1891    3105#
AMRVEC= 000330            815#   1890
AMSGAD= 000000           1151    1159
AMSGLG= 000000           1151    1160
AMSGTY= 000000           1151    1153
AMTST   005200           1822    1827    1871    1879#
AMTYP1= 000000           1151    1174
AMTYP2= 000000           1151    1182
AMTYP3= 000000           1151    1185
AMTYP4= 000000           1151    1188
AMXB  = 176616            844#   3102*
AMXC  = 176614            843#   1897*   3101*   3117*
AMXSRV  012604           1892    3101#
AMXVEC= 000334            816#
APASS - 000000           1151    1156
APRIOR= 000000           1151
APTCSU- 000040           4093    4352#
APTENV- 000001           4086    4308    4350#   4383
```

```
APTSIZ= 000200  1559  4349#
APTSPO= 000100  4088  4310  4351#
ASWREG= 000000  1151  1164
ATESTN= 000000  1151  1155
AUNIT = 000000  1151  1158
AUSWR = 000000   669#  1151  1165
AVECT1= 000300   669#  1151  1190
AVECT2= 000000  1151  1191
BITHLD  011662  2839*  2843  2857#
BIT0  = 000001   788#   956  1052  1926
BIT00 = 000001   778#   788
BIT01 = 000002   777#   787
BIT02 = 000004   776#   786
BIT03 = 000010   775#   785
BIT04 = 000020   774#   784
BIT05 = 000040   773#   783
BIT06 = 000100   772#   782
BIT07 = 000200   771#   781
BIT08 = 000400   770#   780
BIT09 = 001000   769#   779
BIT1  = 000002   787#   921   943   979  1051  1960
BIT10 = 002000   768#   936   970  1820  4371
BIT11 = 004000   767#   969  1010  1879
BIT12 = 010000   766#   935   950   968   985  1009  1740  1773  1807  1870  1905  1923  1948
                1982  2016  3476  3490  3509  3837  3851  3871
BIT13 = 020000   765#   934   949   967  1957  4378
BIT14 = 040000   764#   933   948   966   984   992  1991
BIT15 = 100000   763#   932   947   965   983  1008  1027  1782  3597  3776
BIT2  = 000004   786#   942   978  1050  1994
BIT3  = 000010   785#   941   977  1016  1049  1833  1882  2605
BIT4  = 000020   784#   976  1015  1030  1031  1048
BIT5  = 000040   783#   940   975  1014  1047
BIT6  = 000100   782#   939   955   974  1013  1029
BIT7  = 000200   781#   938   954   973  1012  1028  1046  1760  2574
BIT8  = 000400   780#   972  1011  2032  2128  2166  2225  2275  2299  2344  2410  2447
BIT9  = 001000   779#   937   971  2034  2147  2174  2250  2280  2373
BPTVEC= 000014   795#
BREAK = 000001   956#
B110  = 001000   890#
B1200 = 003400   895#
B134  = 001400   891#
B150  - 002000   892#
B1800 = 004000   896#
B19200= 007400   903#
B2000 = 004400   897#
B2400 = 005000   898#  1928  1930  1962  1964  1996  1998
B300  = 002400   893#
B3600 = 005400   899#
B4800 = 006000   900#
B50   = 000000   888#
B600  = 003000   894#
B7200 - 006400   901#
B75   = 000400   889#
B9600 - 007000   902#  1787  1884  1886  2612
CDET    010000   935#   968#
CHAR5   000000   914#
```

```
CHAR6 = 000004            915#
CHAR7 = 000010            916#
CHAR8 = 000014            917#  1787  1884  1886  1928  1930  1962  1964  1996  1998  2612
CHR5  = 000000            996#
CHR6  = 002000            997#
CHR7  = 004000            998#
CHR8  = 006000            999#  1836
CKSWR = 104407           2673   4367  4393  4654#
CK0BAD  011754           2238   2904#
CK1BAD  012066           2263   2952#
CLK   = 177546            836#  1621*  1765*
CLKTST  004466           1741   1744  1760#
CLKVEC= 000100            813#  1512*  1513*  1762*  1763*
CLOCK   017244           1775   2590  4062#
CLOPT   017102           2567   4062#
CLPRES  010742           1914   2563*  2566*  2622#
CLRBAD  011664           1635   2864#
CLT1    017122           1950   2595  4062#
CLT2    017133           1984   2596  4062#
CLT3    017144           2018   2597  4062#
CL1TST  005340           1880   1903  1906  1914#
CL2TST  005530           1924   1946  1949  1957#
CL3TST  005662           1958   1980  1983  1991#
COMCHR  012700           1859*  1864  1894*  1900  3102  3106  3110  3112*  3115*  3120#  3128  3129  3152
                         3156   3158*  3161*  4062
COMHLD  012702           1848*  1849  1895*  3105*  3106  3121#  3149*  3150  3152  4062
COMPAT  020350           2514   4062#
COMP1   003132           1494#  1500*  1504*  1508*  1515*  1585  1637  2182  2512  2516*  3593  3772
COMP2   003134           1495#  1503*  1509*  1516*  1587  1639  2134  2153  2285  2517*  3595  3774
CONSOL= 010000            878#
COPDUN  020256           3967   4062#
COPFLG  003130           1493#  1507*  1517*  1582  1646  3895*
COPMSG  020155           3896   4062#
COPY    016354           1648   3895#  3969
CR    = 000015            703#  1792  2995  3008  4132  4142
CRC   = 000010           1049#  3343  3717
CRLF  = 000200            704#  1581  4062  4103  4142
CTHLD   011400           2750*  2754  2768#
CTREG   011470           2778*  2782*  2793#  2806*  2810*
CTREG1  011472           2777*  2785*  2794#  2805*  2813*
CTS   = 020000            934#   967#
CT1     000000            879#  1928  1930
CT2   = 050000            880#  1962  1964
CT3   = 060000            881#  1996  1998
DATCHK  017046           3951   4042#
DD      000040           1047#  3366
DDISP - 177570            710#  1134  1547
DEVM    020445           2680   4062#
DF      023776           1213   1216  1219  1222  1225  1228  1231  1234  1237  1240  1243  1246  1249
                         1288   1291  1294  1297  1300  1303  1306  1309  1312  1315  1318  1321  1324
                         1327   1330  1333  1336  1342  1345  1348  1351  1354  1357  1360  1363  1390
                         1393   1396  1399  1405  1408  1411  1414  4062#
DF1     024004           1252   1255  1258  1261  1270  1273  1276  1279  1366  1369  1372  1375  1378
                         1381   1384  1387  1429  1438  4062#
DF2     024014           1264   1267  1282  1285  1339  1432  1441  4062#
DH1     023055           1228   1288  1294  1297  1300  1303  1306  1315  1318  1321  4062#
```

```
DH2     023074  1213  1216  1222  4062#
DH3     023120  1219  1225  4062#
DH4     023162  1231  1291  4062#
DH5     023213  1234  1237  1240  1243  1246  1396  1399  4062#
DH6     023250  1249  1309  1312  1324  1327  1330  1333  1336  1342  1345  1348  1351  1354
                1357  1360  1363  1390  1393  1405  1408  1411  1414  4062#
DH7     023315  1252  1255  1258  1261  1270  1273  1276  1279  1366  1369  1372  1375  1378
                1381  1384  1387  1429  1438  4062#
DH8     023404  1264  1267  1282  1285  1339  1432  1441  4062#
DISPLA  001142  1134#  1547*  1555*  4370*
DISPRE  000174  1061#  1555
DNA   = 100000  1008#
DNAIE = 000040  1014#
DONE  = 000200   938#   973#  1028#  1844  2039  2047  2061  2075  4073  4032
DROP    017647  2587  4062#
DRV0    017232  2591  3478  3492  3511  4062#
DRV1    017237  2592  3839  3853  3873  4062#
DSC   = 100000   965#
DSI   = 100000   932#
DSIE  = 000040   940#   975#
DSRDY = 001000   937#   971#
DSWR  = 177570   709#  1133  1546
DTR   = 000002   943#   979#  1596  1597  1627  1837  1887  1920  1921  2613  2614
DT1     023474  1228  1288  1294  1297  1300  1303  1306  1315  1318  1321  4062#
DT10    023624  1252  1255  1258  1261  1366  1369  1372  1375  1429  4062#
DT11    023646  1264  1267  1339  1432  4062#
DT12    023670  1270  1273  1276  1279  1378  1381  1384  1387  1438  4062#
DT13    023712  1282  1285  1441  4062#
DT14    023734  1309  1312  1324  1327  1330  1333  1548  1354  1393  1405  1408  1411  1414
                4062#
DT15    023752  1396  4062#
DT16    023764  1399  4062#
DT2     023502  1213  1216  1222  4062#
DT3     023512  1219  1225  4062#
DT4     023526  1231  1291  4062#
DT5     023536  1234  4062#
DT6     023550  1237  4062#
DT7     023562  1240  4062#
DT8     023574  1243  1246  4062#
DT9     023606  1249  1336  1342  1345  1351  1357  1360  1363  1390  4062#
DUN     017756  1743  1873  3482  3494  3513  3843  3855  3875  4062#
DXTST   003120  1481#  2030*  2113*  2214*  2302*  2347*  2413*  2451*  3238  3364  3426  3450  3521
                3523  3882  4062
DXTST0  006014  1916  1992  2014  2017  2030#
DXTST1  006256  1644  2035  2092  2113#
DXTST2  006712  2187  2197  2214#
DXTST3  007254  2286  2299#
DXTST4  007430  2300  2313  2344#
DXTST5  007702  2345  2374  2382  2410#
DXTST6  010020  2411  2424  2433  2447#
DX1   = 000020  1030#  2160  2269  2378  2394  3548  3578  3600  3616  3636  3671  3684  3705
                3714  3753  3857  3876  3924  3937
DOBAD   002624  1461#  2864  2866  2882  2889  2911  2912
DOBUF   002424  1460#  2067  2080  2090  3383  3404  3921  4043
DOCDAT  014104  3384  3399#
DOCERR  002414  1456#  2120*  2218*  2304*  2324*  2364*  2415*  3424  3439  3445*  3449*  4062
```

```
DOCMER  002412  1455# 3290* 3399* 3409* 3417  3466* 3504*
DOCNT   002402  1451# 2219* 2277  3506* 3507
DOCRC   002422  1459# 3345* 3349* 3419  3434  3446*
DODISP  013174  2136* 2139* 2241* 2306* 2328* 2348* 2366* 2417* 2426* 2456* 3197  3198# 3218*
                3237* 3261* 3296* 3310* 3330* 3339* 3384* 3534* 3543* 3555* 3563*
DODUN   014746  3454  3525  3534  3543  3555  3563  3573#
DOEBUF  014046  2328  3370  3382#
DOERR   002404  1452# 2119* 2217* 2303* 2323* 2363* 2414* 3255  3268  3278* 3289* 3293* 3325
                3327* 3353  3363* 3465* 3503* 4062
DOEXP   002416  1457# 3407* 4062
DOFBUF  013212  2139  2417  3214# 3286  3296
DOINIT  014632  2306  3532#
DOINV   014730  2456  3563#
DONEXT  014610  3339  3351  3447  3521#
DONXT1  014330  3465# 3522
DONXT2  014524  3503# 3524
DOPAT   002400  1450# 2118* 2168  3495*
DORCHK  013622  3310  3323#
DOREC   002420  1458# 3408* 4062
DOREST  014656  2348  3543#
DORSEC  013566  2136  2241  2366  3291  3305# 3329  3330  3452  3474
DOSEC   002410  1454# 2121* 2236* 2326* 2361* 2453* 2479* 2909  3217  3235  3279  3281* 3283
                3285* 3288* 3307  3403  3467  3469* 3471  3473* 3487* 4062
DOSTAT  014712  2426  3555#
DOTMO   011474  2170  2776* 2787* 2795#
DOTRK   002406  1453# 2115* 2191* 2198* 2231* 2325* 2360* 2452* 2478* 2907  3215  3233  3305
                3401  3480  3484  3486* 4062
DOWCHK  013332  3237  3253#
DOWSEC  013256  3218  3233#
D1BAD   003074  1479# 2869  2871  2930  2937  2959  2960
D1BUF   002674  1478# 3751  3778  3949  4044
D1CDAT  015646  3752  3766#
D1CERR  002664  1474# 2125* 2222* 2391* 3796  3805  3811* 3815* 4062
D1CMER  002662  1473# 3664* 3766* 3783* 3790  3827* 3866*
D1CNT   002652  1469# 2223* 2282  3868* 3869
D1CRC   002672  1477# 3719* 3723* 3792  3801  3812*
D1DISP  013202  2155* 2158* 2266* 2376* 2393* 3200  3201# 3547* 3599* 3615* 3635* 3670* 3683*
                3704* 3713* 3752*
D1DUN   014762  3547  3577#
D1EBUF  015610  3738  3750#
D1ERR   002654  1470# 2124* 2221* 2390* 3629  3642  3652* 3663* 3667* 3699  3701* 3727  3737*
                3826* 3865* 4062
D1EXP   002666  1475# 3781* 4062
D1FBUF  014776  2158  3588# 3660  3670
D1INT   013176  3195  3200#
D1NEXT  016342  3713  3725  3813  3882#
D1NXT1  016062  3826# 3883
D1NXT2  016256  3865# 3884
D1PAT   002650  1468# 2123* 2176  3856*
D1RCHK  015406  3683  3697#
D1REC   002670  1476# 3782* 4062
D1REST  014674  2376  3547#
D1RSEC  015352  2155  2266  2393  3665  3678# 3703  3704  3816  3835
D1SEC   002660  1472# 2126* 2261* 2389* 2957  3591  3613  3653  3655* 3657  3659* 3662* 3680
                3770  3828  3830* 3832  3934* 3848* 4062
D1TMO   011564  2178  2804* 2815* 2821#
```

```
D1TRK   002656          1471#   2116*   2192*   2199*   2256*   2388*   2955    3589    3611    3678    3768    3841    3845
                        3847*   4062
D1WCHK  015116          3615    3627#
D1WSEC  015062          3599    3611#
EBHLD   006250          2081*   2085*   2086    2096#   4062
EBUF  = 000003          1036#   2068    3385    3753    4026
EBUFF   017004          3920    3948    4024#
EMTVEC= 000030           798#   1538*   1539*
EM1     020456          1213    1216    1222    4062#
EM10    020771          1249    4062#
EM11    021016          1252    1255    4062#
EM13    021034          1258    1405    4062#
EM14    021065          1261    4062#
EM15    021116          1264    4062#
EM16    021147          1267    4062#
EM17    021200          1270    1273    1408    4062#
EM19    021216          1276    1414    4062#
EM2     020475          1219    1225    4062#
EM20    021247          1279    4062#
EM21    021300          1282    4062#
EM22    021331          1285    4062#
EM23    021362          1288    4062#
EM24    021373          1291    4062#
EM25    021410          1294    4062#
EM26    021425          1297    4062#
EM27    021442          1300    4062#
EM28    021457          1303    4062#
EM29    021476          1306    4062#
EM3     020517          1228    4062#
EM30    021516          1309    1324    1327    1330    1333    1348    1354    4062#
EM31    021527          1312    4062#
EM32    021540          1315    4062#
EM33    021603          1318    4062#
EM34    021644          1321    4062#
EM35    021703          1336    4062#
EM36    021750          1339    4062#
EM37    022017          1342    4062#
EM38    022043          1345    4062#
EM39    022111          1351    4062#
EM4     020563          1231    4062#
EM40    022141          1357    4062#
EM41    022204          1360    4062#
EM42    022240          1363    4062#
EM43    022261          1366    1369    4062#
EM45    022313          1372    1375    4062#
EM47    022344          1378    1381    4062#
EM49    022376          1384    1387    4062#
EM5     020606          1234    4062#
EM51    022427          1390    4062#
EM52    022453          1393    4062#
EM53    022477          1396    1399    4062#
EM54    022546          1411    4062#
EM55    022563          1420    4062#
EM56    022620          1423    4062#
EM57    022655          1429    4062#
EM58    022676          1432    4062#
```

```
EM59    022717          1435    4062#
EM6     020634          1237    4062#
EM60    022755          1438    4062#
EM61    022776          1441    4062#
EM62    023017          1444    4062#
EM7     020662          1240    4062#
EM8     020710          1243    4062#
EM9     020741          1246    4062#
ENDPAS  017764          2493    4062#
ENVPAR= 001400          1003#
EOP     010200          2094    2184    2287    2463    2492#
EPAR  = 000060           909#
ERFLG   016666          3901*   3912*   3913    3919*   3928*   3929    3935*   3941*   3942    3973#
ERR   = 100000           947#    983#   1027#
ERRNUM  025764          4062    4455*   4465#
ERRVEC= 000004           791#   1544    1545*   1556*   1662*   1663*   1737*   1738*   2527*   2528*   2570*   2571*
EXAMKB  011102          1660    2671#   2756    2781    2809    2845
EXP0    006252          2097#   4062
EXP1    006254          2098#   4062
FBUF  = 000001          1035#   2053    3219    3600
FIN     003122          1488#   2140*   2159*   2243*   2268*   2308*   2330*   2349*   2365*   2377*   2392*   2419*   2428*
                        2458*   2779    2807    3483*   3505*   3573*   3577*   3844*   3867*
FLGHLD  011376          2749*   2754    2767#
FRERR = 020000           949#
GETSEC  016674          3899    3981#
GNS   = ****** U        1580    4646    4647    4648    4649    4650    4652    4654    4655    4656    4657
GTDEVM= 104412          1603    4657#
GTSWR = 104406          1575    4652#
GT$DEV  011116          2680#   4183    4269    4657
HFDUP = 000010          1016#
HILIM   011074          2229*   2234*   2254*   2259*   2660    2665#
HINUM   011076          2642    2650    2655*   2667#
HLD42   004464          1525*   1696    1754#
HT    - 000011           701#   4101    4142
ID    - 000001          1052#   2319
IE      000100           939#    955#    974#   1013#   1029#   1621    1765    1794    1805    1860    1861    1896    1897
                        1939    1940    1973    1974    2007    2008    2141    2160    2244    2269    2309    2331    2350
                        2367    2378    2394    2420    2429    2459    3021    3037    3048    3064    3074    3090    3101
                        3117    3127    3135    3162    3166    3219    3240    3242    3262    3297    3311    3331    3340
                        3385    3496    3514    3535    3544    3548    3556    3564    3574    3578    3600    3616    3636
                        3671    3684    3705    3714    3753    3857    3876
INITAL  000011          1039#   2043    3535
INTFLG  012150          1669*   1671    1683*   1685    1716*   1718    1764*   1768    2529*   2533*   2536    2553*   2556
                        2979*   2981#
INTSRV  012142          1662    1762    2527    2979#
INVADR- 000002          1051#   2465
INVCT   010176          2450*   2474    2477*   2485#
IOTVEC= 000020           796#
LDATA = 000004          1050#
LDOBAD  011716          2880#   3257    3334
LD1BAD  012030          2928#   3631    3708
LF    - 000012           702#   2997    3004    4062    4136    4142
LINCT   012300          1791*   1798    3009*   3014#
LOLIM   011072          2228*   2233*   2253*   2258*   2658    2664#
LONUM   011100          2641    2648    2654*   2668#
LOOP    003732          1583    1586    1588    1607    1611    1616#   2518
```

```
LSTAD   026532            1676    1681    3981    4659#
LSTCHR  012772            1829*   3132    3134*   3143#
MAINT = 000002             921#   1884    1928    1962    1996
MAXMEM  010766            1665    1711    2544*   2545*   2634#
MAXSEC  016662            3955    3971#   3994*
MAXSK   003126            1491#   1591*   2277    2282    2510*   3507    3869
MAXTRK  003124            1489#   1590*   1643*   2189    2193*   2196    2200*   2215*   2229    2254    2509*   3484    3845
MCLK  = 004000            1010#   1831    1835
MEM     017314            2550    4062#
MEMCT   004454            1679*   1705    1707*   1731*   1732    1746#
MEMHLD  004456            1689*   1690    1722*   1723    1751#   4062
MEMORY  017743            1742    4062#
MEMTBL  010750            2547    2626#
MEMTST  004046            1647    1660#
MM    = 010000            1009#   1835
MR    = 000400            1011#   1830    1869
MSG1    020001            2496    4062#
MSG2    020030            2499    4062#
MSG3    020064            2502    4062#
MSG4    020121            2505    4062#
MYTYPE  025732            4380    4455#
M12K    017263            2628    4062#
M16K    017270            2629    4062#
M20K    017275            2630    4062#
M24K    017302            2631    4062#
M30K    017307            2632    4062#
M4K     017253            2626    4062#
M8K     017257            2627    4062#
NOTPRE  017334            2568    2609    4062#
NULL    010336            2508    2520#
ODDPAR= 001000            1004#   1836
OK      017701            4062#
OPAR  = 000020             910#   1884    1886    2612
ORERR = 040000             948#    984#
PARAM = 177420             834#   1787*   1884*   1886*   1928*   1930*   1962*   1964*   1996*   1998*   2612*
PARERR= 010000             950#    985#
PAT     004460            1680*   1684    1690    1708*   1714*   1717    1723    1734*   1752#   4062
PATT    017712            3493    3854    4062#
PERR    025772            1632*   2503    4462*   4469#
PIRQ  = 177772             708#
PIRQVE= 000240             802#
PORT    017215            1812    4062#
PRB   = 177516             839#   2994*
PRCHR   012276            1792*   2994    2995    2997    2999    3001*   3004*   3006*   3008*   3013#
PRINT   017204            1809    2598    2608    4062#
PRPORT  010746            1803    1810    2601*   2617*   2624#
PRPRES  010744            1784    2600*   2618*   2623#
PRS   = 177514             838#   1794*   1805*   2602    2615    2988
PRSRV   012152            1788    2988#
PRT   = 020000             882#   1787
PRTST   004560            1761    1771    1774    1782#
PRVEC - 000200             811#   1788*   1789*
PR0   - 000000             725#   1625
PR1     000040             726#
PR2     000100             727#
PR3     000140             728#
```

```
PR4    = 000200            729#   1524
PR5    = 000240            730#
PR6    = 000300            731#
PR7    = 000340            732#
PS     = 177776            705#    706
PSERR    025774           1633*   2506   3271*  3356*  3442*  3645*  3730*  3808*  4470#
PSW    = 177776            706#
PWRVEC= 000024             797#
RACT   = 004000            969#
RAND     010770           2230    2235   2255   2260   2641#  2659   2661
RANDOM   017725           3512    3874   4062#
RDCHR  = 104410           4263    4655#
RDLIN  = 104411           4656#
RDWR     016744           3907    3923   3936   4006#
RDY    = 000200            954#   1012#  1046#
REGHLD   011660           2838*   2843   2856#
RESTOR= 000017            1042#   3262   3297   3331   3340   3544   3548   3636   3671   3705   3714
RESVEC= 000010             792#
RING   = 040000            933#    966#
RNF    = 000020           1048#   3258   3272   3294   3328   3335   3357   3632   3646   3668   3702   3709   3731
RSEC   = 000007           1038#   3311   3684   3908   3937
RSTAT  = 000013           1040#   3556
RTS    = 000004            942#    978#  1596   1597   1627   1837   1887   1920   1921   2613   2614
RUN      017670           1776    1814   1908   1951   1985   2019   4062#
RXCS   = 177170            852#   2038   2043*  2046   2053*  2060   2068*  2074   2141*  2160*  2244*  2269*  2309*
                          2315    2331*  2350*  2356   2367*  2378*  2384   2394*  2420*  2429*  2435   2459*  2470
                          3179    3181   3187   3190   3194   3219*  3240*  3242*  3253   3262*  3297*  3311*  3323
                          3331*   3340*  3385*  3496*  3514*  3535*  3544*  3548*  3556*  3564*  3574*  3578*  3600*
                          3616*   3627   3636*  3671*  3684*  3697   3705*  3714*  3753*  3857*  3876*  3904   3910
                          3926    3939   4010*  4012   4026*  4031   4062
RXDB   - 177172            853#   2054*  2069   3220*  3386   3601*  3754   4027
RXES   = 177172            854#   2319   2465   3182   3191   3366   4062
RXSA   = 177174            855#   2454*  2480*  2885   2933   3183   3192   3236*  3308*  3532*  3614*  3681*  4009*
                          4062
RXSRV    013074           2131    2150   3179#
RXVEC  = 000264            821#   2131*  2132*  2150*  2151*
SCOM     017155           1872    2593   4062#
SECCNT   016664           3953*   3955   3972#  3997*
SECT     016672           3903*   3958   3960*  3964*  3975#  4008
SEND   = 000020           1015#   1839   3139
SETVEC   011272           1854    1889   1932   1966   2000   2730#
SIZE     010342           1605    2527#
SMPAR  - 176622            848#   1836*
SMRB   = 176622            847#   1848   3149
SMRC   = 176620            846#   1837*  1843   1860*  3162*
SMRSRV   012774           1856    3149#
SMRVEC= 000340             818#   1855
SMTST    004724           1783    1785   1801   1808   1813   1820#
SMXB   = 176626            850#   1840*  3128*
SMXC   = 176624            849#   1830*  1831*  1835*  1839*  1861*  1869*  3127*  3135*  3139*  3166*
SMXSRV   012704           1857    3127#
SMXVEC= 000344             819#
SPRS     012302           2988*   2989   3015#  4062
SREC   = 002000            936#    970#
SRSYN  = 000020            976#   1837
SRXCS    013204           3181*   3190*  3204#  4062
```

```
SRXES   013206          3182* 3191* 3205# 3258  3272  3294  3328  3335  3343  3357  3632  3646  3668
                        3702  3709  3717  3731  4062
SRXSA   013210          3183* 3192* 3206# 4062  4456
STACK = 001100           696# 1519  1536
START   003174          1069  1072  1512# 4662
START1  003136          1077  1500#
START2  003144          1079  1503#
START3  003156          1075  1507#
STKLMT= 177774           707#
STSYN = 000400           972#
SWR     001140          1133# 1534  1546* 1548  1554* 1561* 1573  1740  1773  1807  1870  1905  1948
                        1982  2016  3476  3490  3509  3837  3851  3871  4152  4197* 4371  4378  4390
SWREG   000176          1062# 1554  1573  4152  4165
SW0   = 000001           760#
SW00  = 000001           750#  760
SW01  = 000002           749#  759
SW02  = 000004           748#  758
SW03  = 000010           747#  757
SW04  = 000020           746#  756
SW05  = 000040           745#  755
SW06  = 000100           744#  754
SW07  = 000200           743#  753
SW08  = 000400           742#  752
SW09  = 001000           741#  751
SW1   = 000002           759#
SW10  = 002000           740#
SW11  = 004000           739#
SW12  = 010000           738#
SW13  = 020000           737#
SW14  = 040000           736#
SW15  = 100000           735#
SW2   = 000004           758#
SW3   = 000010           757#
SW4   = 000020           756#
SW5   = 000040           755#
SW6   = 000100           754#
SW7   = 000200           753#
SW8   = 000400           752#
SW9   = 001000           751#
SXMIT = 000010           941#  977#
SYNC1 = 040000           992#
SYNC2 = 000000           991#
S1      003214          1501  1516#
S2      003220          1505  1517#
S3      003224          1510  1519#
TBITVE= 000014           793#
TERR    025766          1593* 2497  4461* 4467#
TIMD0   011402          2143  2246  2311  2333  2352  2369  2422  2431  2461  2776#
TIMD1   011476          2162  2271  2380  2396  2304#
TIMER   011316          1767  1797  1863  1899  1942  1976  2010  2749#
TIMER1  011566          1618  1630  2827#
TIMER2  011600          1842  2037  2045  2059  2073  2838#  4011  4030
TKVEC = 000060           800#
TK1B  = 176502           858#  1938  3025
TK1S    176500           857#  1939* 2554
TK1SRV  012322          1934  3025#
```

```
TK1VEC= 000300   823#  1933
TK2B  = 176512   863#  1972  3052
TK2S  = 176510   862#  1973*
TK2SRV  012422  1968  3052#
TK2VEC= 000310   825#  1967
TK3B  = 176522   868#  2006  3078
TK3S  = 176520   867#  2007*  2558
TK3SRV  012522  2002  3078#
TK3VEC= 000320   827#  2001
TPVEC = 000064   801#
TP1B  = 176506   860#  3022*
TP1S  = 176504   859*  1940*  3021*  3037*
TP1SRV  012304  1935  3021#
TP1VEC= 000304   824#
TP2B  = 176516   865#  3049*
TP2S  = 176514   864#  1974*  3048*  3064*
TP2SRV  012404  1969  3048#
TP2VEC= 000314   826#
TP3B  = 176526   870#  3075*
TP3S  = 176524   869#  2008*  3074*  3090*
TP3SRV  012504  2003  3074#
TP3VEC= 000324   828#
TRACK   016670  3902*  3963*  3974#  4006
TRAPVE= 000034   799#  1540*  1541*
TRK     017705  3479  3840  4062#
TRTVEC= 000014   794#
TSERR   025770  1594*  2500  3270*  3355*  3441*  3644*  3729*  3807*  4468#
TYPDS = 104405  2495  2498  2501  2504  2507  3481  3842  4440  4650#
TYPE  = 104401  1567  1612  1742  1743  1775  1776  1809  1812  1814  1872  1873  1907  1908
                1950  1951  1984  1985  2018  2019  2493  2496  2499  2502  2505  2508  2514
                2548  2550  2567  2568  2585  2587  2608  2609  2680  2683  2699  2718  3478
                3479  3482  3492  3493  3494  3511  3512  3513  3839  3840  3843  3853  3854
                3855  3873  3874  3875  3896  3967  4106  4163  4164  4167  4178  4188  4199
                4218  4267  4272  4275  4279  4373  4381  4404  4421  4423  4426  4428  4443
                4448  4529  4598  4646#
TYPOC - 104402  2682  4166  4412  4436  4647#
TYPON = 104404  4649#
TYPOS - 104403  4648#
T1CHR   012400  1937*  1943  3022  3026  3030  3032*  3035*  3040#  4062
T1HLD   012402  1938*  3025*  3026  3041#  4062
T2CHR   012500  1971*  1977  3049  3053  3057  3059*  3062*  3067#  4062
T2HLD   012502  1972*  3052*  3053  3068#  4062
T3CHR   012600  2005*  2011  3075  3079  3083  3085*  3088*  3093#  4062
T3HLD   012602  2006*  3078*  3079  3094#  4062
ULITE - 040000   884#
USEL  = 000020  1031#  3194
WARNIN  017351  1612  4062#
WDDSEC= 000015  1041#  3240
WSEC  = 000005  1037#  3242  3564  3616  3924
$APTHD  001000  1093  1099#
$ASTAT= ****** U  4330  4345
$ATYC   025174  4301  4303#
$ATY1   025150  4299#
$ATY3   025156  4091  4300#
$ATY4   025166  4302#  4386
$AUTOB  001134  1130#  1577*  4160  4295
```

```
$BASE   001244          1192#
$BDADR  001122          1125#
$BDDAT  001126          1127#
$BELL   001160          1143#   4373    4396
$CHARC  024302          4108*   4118*   4125    4134*   4139#
$CKSWR  024306          4152#   4654
$CMTAG  001100          1113#   1531    1532    1542
$CM3  =  000000         1143#
$CNTLC  025106          4178    4267    4288#
$CNTLG  025120          4163    4290#
$CNTLU  025113          4188    4289#
$CPUOP  001216          1166#
$CRLF   001165          1145#   2699    4107    4142    4199    4288    4381    4396    4404    4423    4428    4448
$DBLK   026212          4495    4529    4537#
$DEVCT  001200          1157#   2759*   2784*   2812*   2848*   3010*   3036*   3063*   3089*   3116*   3136*   3140*   3163*
$DEVM   001246          1193#   1608    1610    1760    1782    1820    1833    1879    1882    1918    1923    1926    1957
                        1960    1991    1994    2032    2034    2128    2147    2166    2174    2225    2250    2275    2280
                        2299    2344    2373    2410    2447    2576    2605    2681    2697*
$DTBL   026202          4498    4533#
$ENV    001210          1162#   1526    1571    1601    1606    1824    2671    4086    4308    4332    4383
$ENVM   001211          1163#   1559    4088    4093    4310
$ERFLG  001103          1116#   4368*   4396
$ERMAX  001115          1122#
$ERROR  025416          1538    4366#
$ERRPC  001116          1123#   4062    4375*   4376*   4377    4396    4410
$ERRTB  001250          1210#   4418
$ERRTY  025556          4403#   4460
$ERTTL  001112          1120#   4374*   4396
$ETABL  001210          1161#
$ETEND  001250          1105    1194#
$FATAL  001172          1154#   4336*
$FFLG   025414          4299*   4302*   4330    4339*   4347#
$FILLC  001156          1141#   4111    4142
$FILLS  001155          1140#   4142
$GDADR  001120          1124#
$GDDAT  001124          1126#
$GTSWR  024356          4164#   4652
$HD  =  000000           680
$HIBTS  001000          1100#
$ICNT   001104          1117#
$INTAG  001135          1131#   4180    4200    4295
$ITEMB  001114          1121#   4377*   4385    4396    4407    4455
$LF     001166          1146#   4142    4279    4288    4396
$LFLG   025413          4340*   4346#
$LPADR  001106          1118#
$LPERR  001110          1119#
$LSTCN= 177777           669#
$LSTIN- 000000           669#
$LSTST= 177777           669#
$LSTTA= 000000           669#
$MADR1  001222          1179#
$MADR2  001226          1183#
$MADR3  001232          1186#
$MADR4  001236          1189#
$MAIL   001170          1101    1105    1152#   1558    1571    4086    4383
$MAMS1  001220          1173#
```

```
$MAMS2  001224            1181#
$MAMS3  001230            1184#
$MAMS4  001234            1187#
$MBADR  001002            1101#
$MFLG   025412            4300*  4306   4341*  4345#
$MNEW   025136            2683   4167   4293#
$MSGAD  001204            1159#  4316*  4319
$MSGLG  001206            1160#  4321*
$MSGTY  001170            1153#  4314   4322*  4334   4338*
$MSWR   025125            4164   4291#
$MTYP1  001221            1174#
$MTYP2  001225            1182#
$MTYP3  001231            1185#
$MTYP4  001235            1188#
$NULL   001154            1139#  4113   4142
$OCNT   026444            4570*  4599*  4612#
$OMODE  026446            4565*  4569*  4574   4577*  4588*  4614#
$PASS   001176            1156#  1558*  1826   2186   2492*  2494
$PASTM  001006            1103#
$QUES   001164            1144#  2718   4142   4218   4272   4288   4396
$RDCHR  024630            4231#  4655
$RDDEC= ****** U          4657
$RDLIN  024750            4259#  4656
$RDOCT= ****** U          4657
$RDSZ = 000010            4252#
$R2A  = ****** U          4657
$SAVRE= ****** U          4657
$SETUP= 000106            1498#  1537   1538   1540   1542   1567   1568   4147   4295   4367   4393   4395
$STUP - 177777            1498#
$SWR  = 132000             669#   680   1143   1542   4359   4360   4361   4362   4363   4371   4378   4390   4394
                          4396
$SWREG  001212            1164#  1561
$TESTN  001174            1155#  4456*
$TKB    001146            1136#  2690   4145   4156   4173   4235   4241
$TKS    001144            1135#  1520*  2688   4145   4154   4170   4182*  4202*  4233   4239
$TN   = 000001             669#   680
$TPB    001152            1138#  4131*  4142
$TPFLG  001157            1142#  4080   4142
$TPS    001150            1137#  4129   4142
$TRAP   026450            1540   4623#
$TRAP2  026472            4634#  4645
$TRP    000013            4638#  4647#  4648#  4649#  4650#  4651#  4652   4653#  4654   4655#  4656#  4657#  4658#
$TRPAD  026504            4628   4645#
$TSTM   001004            1102#
$TSTNM  001102            1115#  4370   4396
$TTYIN  025076            4260   4261   4283   4287#
$TYPBN- ****** U          4651
$TYPDS  025776            4483#  4650
$TYPE   024024            4080#  4327   4638   4646
$TYPEC  024236            2702   4110   4117   4124   4129#  4130   4204
$TYPEX  024304            4135   4137   4140#
$TYPOC  026246            4568#  4647
$TYPON  026262            4567   4570#  4649
$TYPOS  026222            4563#  4648
$UNIT   001202            1158#
$UNITM  001010            1104#
```

```
$USWR   001214            1165#
$VECT1  001240            1190#
$VECT2  001242            1191#
$OFILL  026445            4564*   4568*   4578    4613#
$40CAT= ****** U          4380
.       = 026532          1060#   1064#   1068#   1071#   1074#   1076#   1078#   1083#   1089    1090#   1092#   1094#   1112#
                          1147    1460#   1461#   1478#   1479#   1535    2436    2483    2521#   2762    2788    2816    2829
                          2851    3905    3917    3933    3946    4049    4062#   4142    4145    4287#   4288    4295#   4348#
                          4396    4451#   4537#
.$ASTA= ****** U          4300    4303
.$X     = 001000          1089#   1094
```

```
COMMEN    803#
ENDCOM    803#
ERROR     697#  1674  1688  1693  1721  1726  1770  1800  1845  1851  1866  1902  1945  1979  2013
          2040  2048  2062  2076  2083  2088  2144  2163  2247  2272  2312  2317  2321  2334  2353
          2358  2370  2381  2386  2397  2423  2432  2437  2462  2467  2472  2891  2939  2991  3028
          3055  3081  3108  3154  3184  3260  3265  3274  3276  3337  3346  3350  3359  3361  3368
          3421  3428  3431  3436  3443  3634  3639  3648  3650  3711  3720  3724  3733  3735  3794
          3798  3803  3809  3915  3931  3944  4014  4033  4047
ESCAPE    803#
GETPRI    803#
GETSWR    803#  1568#
MULT      803#
NEWTST    803#
POP       803#  4342  4343  4524
PUSH      803#  4303  4305  4326  4483
REPORT    669#   803#
SCOPE     698#
SETPRI    803#
SETTRA   4638#  4647  4648  4649  4650  4652  4654  4655  4656  4657
SETUP     803#  1529
SKIP      803#
SLASH     803#
SPACE     803#
STARS     803#   807   809   830   832   872   874   926   928   959   961  1021  1023  1056  1058
         1086  1088  1095  1108  1147  1150  1446  1448  1464  1466  1651  1658  1756  1758  1778
         1780  1816  1818  1875  1877  1910  1912  1953  1955  1987  1989  2021  2028  2100  2111
         2204  2212  2290  2297  2337  2342  2402  2408  2439  2445  2488  2490  2637  2639  2676
         2678  2721  2728  2743  2747  2770  2774  2798  2802  2823  2825  2832  2836  2860  2862
         2876  2878  2899  2902  2924  2926  2947  2950  2974  2977  2984  2986  3017  3019  3044
         3046  3070  3072  3097  3099  3123  3125  3170  3177  3209  3212  3229  3231  3245  3251
         3301  3303  3315  3321  3377  3380  3392  3397  3458  3463  3499  3501  3528  3530  3539
         3541  3551  3553  3559  3561  3568  3571  3583  3586  3607  3609  3619  3625  3674  3676
         3689  3695  3745  3748  3759  3764  3819  3824  3861  3863  3887  3893  3977  3979  4001
         4004  4019  4022  4038  4040  4058  4060  4065  4144  4147  4223  4252  4298  4355  4398
         4473  4540  4617
SWRSU     803#  1542#
TRMTPP   4638#
TYPBIN    803#
TYPDEC    803#  2494  2497  2500  2503  2506  3480  3841  4438
TYPNAM    803#  1563
TYPNUM    803#
TYPOCS    803#
TYPOCT    803#  2681  4165  4410  4435
TYPTXT    803#
$$CMRE   1106#
$$CMTM   1106#
$$ESCA    803#
$$NEWT    803#
$$SET    4638#  4647  4648  4649  4650  4652  4654  4655  4656  4657
$$SETM   1558#
$$SKIP    803#
.EQUAT    669#   693
.HEADE    669#   670
.SETUP    669#  1498
.SWRHI    669#
.$ACT1    669#
```

```
.$APTB    669#   1148#
.$APTH    669#   1084
.$APTY    669#   4296
.$CATC    669#
.$CMTA    669#   1106
.$EOP     669#
.$ERRO    669#   4353
.$ERRT    669#   4396
.$READ    669#   4142
.$SCOP    669#
.$TRAP    669#   4615
.$TYPD    669#   4471
.$TYPE    669#   4063
.$TYPO    669#   4538


. ABS.  026532     000     CON.   RO    ABS   GBL    I


 ERRORS DETECTED:  0

 Z:CVKDAB/I,Z:CVKDAB.SEQ/CRF/SOL=CVKDAB.P11
 RUN-TIME:  22 13 1 SECONDS
 RUN-TIME RATIO: 129/37=3.4
 CORE USED:  24K  (47 PAGES)
```