LPA/AA11-K

DIAGNOSTIC TEST
MD-11-DRLPB-A

EP-DRLPB-A-DL

FICHE 1 OF 1

MAR 1978
digital
MADE IN USA

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DRLPB-A-D

PRODUCT NAME: LPA/AA11-K DIAGNOSTIC TEST

DATE: JANUARY 1978

MAINTAINER: DIAGNOSTIC GROUP

1. ABSTRACT

THIS DIAGNOSTIC EXERCISES THE "AA11-K" ANALOG CIRCUITRY. THE PROGRAM
WHEN STARTED WILL TYPE OUT THE PROGRAM TITLE.
A MESSAGE IS THEN PRINTED GIVING THE TWO LETTER DESIGNATOR
TO BE TYPED TO RUN ANY ONE OF THE FIVE (5) SEPERATE TESTS OF WHICH
THIS PROGRAM IS COMPRISED. THE PROGRAM THEN TYPES A 'CR.' AND
THEN WAITS IN A KEYBOARD MONITOR MODE FOR TWO LETTER'S TO BE TYPED.
ALTHOUGH THESE TESTS MAY BE RUN IN ANY ORDER IT IS IMPERATIVE
THAT TES↓ "AL" IS RUN FIRST AND VERIFY THAT THE AA11-K IS FULLY OPERATIONAL.
THE PROGRAM IS SET UP TO GIVE THE OPERATOR AS MUCH CONTROL
OVER THE PROGRAM AS POSSIBLE VIA THE TELETYPE.TYPING A '↑C'
(OBTAINED VIA TYPING THE 'CNTR' AND 'C' KEYS SIMULTANEOUSLY)
WHILE RUNNING ANY TEST WILL ENABLE THE PROGRAM TO RETURN TO
THE KEYBOARD MONITOR AND AWAIT A NEW LETTER DESIGNATOR TO BE
TYPED. TYPING A '↑G' WHILE RUNNING WILL ENABLE THE SOFTWARE SWITCH
REGISTER VALUE TO BE CHANGED.

THIS PROGRAM IS A MODIFIED VERSION OF "MD-11-DZAAC-B". IT WAS
MODIFIED TO ENABLE THE OPERATOR TO cHECK OUT THE AA11-K OPTION
WHEN IT IS ON THE LPA11KX I/O BUS. NO RECABLING IS NEEDED.
SOME TEST DONE IN THE ORIGINAL DIAGNOSTIC SUCH AS ARBITRATION
TEST, WERE DELETED AS THEY COULD NOT BE CHECKED. IF THIS
DIAGNOSTIC DOE↓N'T FIND A SUSPECTED PROBLEM, YOU MAY HAVE TO
RUN "MD-11-DZAAC-B". YOU SHOULD RUN "MD-11-DRLPA" BEFORE
RUNNING THIS DIAGNOSTIC. PLEASE READ SECTION 1D.

2. REQUIREMENTS (EQUIPMENT)

A. PDP-11 COMPUTER WITH 16K OF MEMORY AND A CONSOLE I/O TERMINAL
B. AA11-K QUAD OPTION MODULE INSTALLED
C. <OPTIONAL> VR14 OR STORAGE SCOPE

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING BINARY TAPES.

4. STARTING PROCEDURE

THE PROGRAM STARTING ADDRESS IS '200'.
THE RESTART ADDRESS IS '204'.

5. CONSOLE SWITCH SETTINGS

THIS PROGRAM HAS BEEN MODIFIED TO RUN WITH OR
WITHOUT A HARDWARE SWITCH REGISTER.
A. ALL SWITCHES SHOULD BE DOWn (0) WHEN THE PROGRAM IS STARTED.
B. REFER TO tHE INDIVIDUAL TEST DESCRIPTIONS FOR APPLICABLE
CONSOLE SWITCH SETTINGS
C. ALL SWITCHES SET TO A 1 WILL SELECT SOFTWARE SWITCH CONTROL.

6. ERRORS

ALL ERRORS ARE ACCOMPANIED WITH A ENGLISH DESCRIPTIVE COMMENT AS TO THE TYPE OF FAILURE. FURTHER QUALIFICATION OF THE ERROR CAN BE OBTAINED FROM THE COMMENT AT THE ERROR PC OR THE TEST ITSELF.

7.0 TEST PROCEDURE

7.1 AUTO LOGIC TEST

A. THIS TEST IS DESIGNED TO VERIFY THE DATA PATH THAT IS ADDRESSABLE FROM THE CPU. THIS ALSO INCLUDES ALL CONTROL, INTERRUPT AND INITILIZE SIGNALS. THE LOGIC TEST ALSO INCLUDES PROVISIONS FOR TESTING MULTIPLE AA11-K'S.

B. STARTING SEQUENCE

1. TYPE 'AL' TO RUN THE AUTO LOGIC TEST.
2. THE PROGRAM WILL THEN EXECUTE A LOGIC TEST ON ALL AVAILABLE UNITS

C. CONTROL SWITCHES

1. TYPING ↑C WILL ENABLE THE PROGRAM TO EXIT AND RETURN TO THE KEYBOARD MONITOR.
2. TYPING ↑G WHEN SOFTWARE SWITCH REGISTER IS ENABLED WILL REPORT LAST VALUE AND WAIT FOR NEW VALUE.

SWITCH	OCTAL	FUNCTION
SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON CURRENT TEST
SW13=1	020000	INHIBIT ERROR TYPEOUT
SW12=1	010000	STORAGE SCOPE CONNECTED
SW11=1	004000	INHIBIT TEST INTERATIONS
SW10=1	002000	EXTERNAL DELAY SIGNAL CONNECTED
SW09=1	001000	TWO'S COMPLIMENT MODE
SW08=1	0004XX	LOOP ON TEST IN SWR 7:0

D. ERRORS

REF. TO 6.

E. RESTRICTIONS

IF A STORAGE SCOPE IS CONNECTED, POWER MUST BE APPLIED TO IT.

F. EXECUTION TIME

IT TAKES APPROXIMATELY 25 SECONDS WITHOUT TEST INTERATIONS.

7.2 AUTO DISPLAY TEST

A. THIS TEST IS DESIGNED TO AID IN THE ADJUSTING AND ALIGNMENT OF THE VR14 OR STORAGE SCOPE SCOPE ON THE AA11-K DISPLAY CONTROL.

B. TYPE 'AD' TO RUN THE AUTO VISUAL DISPLAY TEST.
THE PROGRAM WILL THEN EXECUTE THE VISUAL DISPLAY TEST.

C. CONTROL SWITCHES

1. TYPING ↑C AT ANY TIME WILL ENABLE THE PROGRAM TO EXIT AND RETURN TO THE MONITOR.
2. TYPING ↑G WHEN SOFTWARE SWITCH REGISTER IS ENABLED WILL ENABLE THE PROGRAM TO CHANGE THE SOFTWARE SWITCH REGISTER VALUE.

CONSOLE SWITCHES	FUNCTION
CONSOLE SW10=1	SELECTS EXTERNAL DELAY MODE
CONSOLE SW09=1	SELECTS TWO'S COMPLEMENT MODE
CONSOLE SW08=0	CYCLE THRU ALL FOUR DISPLAY PATTERNS
CONSOLE SW08=1	SELECT PATTERNS IN SW 00-02
CONSOLE SW03=0	SELECT DAC 0 AND DAC 1
CONSOLE SW03=1	SELECT DAC 2 AND DAC 3
CONSOLE SW00-02=0	DISPLAY A HORIZONTAL LINE
CONSOLE SW00-02=1	DISPLAY A VERTICAL LINE
CONSOLE SW00-02=2	DISPLAY A SQUARE
CONSOLE SW00-02=3	DISPLAY AN "X"

D. ERRORS

THE ONLY ERRORS IN THIS TEST ARE DETECTED VISUALLY.

E. RESTRICTIONS

IF VR14, CHANNEL SWITCH MUST BE SET TO "1 & 2" POSITION.
IF STORAGE SCOPE, POWER MUST BE APPLIED.
THE "AUTO-CALIBRATION (AC)" MUST HAVE BEEN RUN PRIOR TO RUNNING THIS SECTION WHEN ON THE "AUTO HARDWARE TESTER".

F. EXECUTION TIME

S IT TAKES APPROXIMATELY 3 MINUTES TO COMPLETE THIS TEST BEFORE IT REPEATS.

7.4 MANUAL LOGIC LOOP

A. THIS LOOP IS PROVIDED TO ENABLE THE OPERATOR A SIMPLE PROGRAM LOOP TO AID IN REPAIR OF THE AA11-K. SWITCH REGISTER BITS 15:13 SELECT THE REGISTER TO BE LOADED AND BITS 12:00 CONTAINS THE DATA TO BE LOADED.

B. STARTING SEQUENCE

1. TYPE 'ML' TO RUN THE MANUAL LOGIC LOOP.
2. THE PROGRAM WILL NOW LOOP AND LOAD THE VALUE OF THE SWITCH REGISTER BITS 12:00 INTO THE SELECTED AA11-K REGISTER.

C. CONTROL SWITCHES

SW15:13	REGISTER SELECTED
000	DAC #0
001	DAC #1
010	DAC #2
011	DAC #3
1XX	STATUS REGISTER

D. ERRORS

NO PROVISIONS ARE MADE FOR LOGIC ERRORS.

E. RESTRICTIONS

NONE

F. EXECUTION TIME

THIS IS A NON-ENDING PROGRAM LOOP THAT CAN BE EXITED BY TYPING ↑C.

7.5 MANUAL DISPLAY LOOP

A. THIS LOOP IS PROVIDED FOR THE OPERATOR TO VERIFY OPERATION OF THE D/A CONVERTER AND MULTIPLEXER.

B. STARTING SEQUENCE

1. TYPE 'MD' TO RUN MANUAL DISPLAY LOOP
2. THE PROGRAM WILL NOW LOAD A "RAMP PATTERN" INTO EACH D/A CONVERTER.

C. CONTROL SWITCHES

1. TYPING ↑C WILL RETURN CONTROL TO tHE KEYBOARD MONITOR.
2. SW10=1 WILL SELECT EXTERNAL DELAY MODE.

D. ERRORS

NO PROVISIONS ARE MADE FOR LOGIC ERRORS.

E. RESTRICTIONS

NONE

7.6 MANUAL CALIBRATION LOOP

A. THIS LOOP IS PROVIDED TO eNABLE THE OPERATOR TO ADJUST THE D/A CONVERTER.

B. STARTING SEQUENCE

1. TYPE 'MC' TO RUN THE MANUAL CALIBRATION LOOP.
2. THE PROGRAM WILL NOW LOAD THE CONTENTS OF THE SWITCH REGISTER INTO EACH D/A REGISTER AND AFTER A DELAY CLEAR THE D/A REGISTER.

C. CONTROL SWITCHES

1. TYPING ↑C WILL EXIT THE LOOP AND RETURN TO tHE KEYBOARD MONITOR.
2. TYPING ↑G WHEN SOFTWARE SWITCH REGISTER IS ENABLED TO CHANGE THE SWITCH REGISTER VALUE
3. SWITCH REGISTER BITS 11:0 ARE LOADED IN TO ALL DAC'S.

D. ERRORS

NO PROVISIONS ARE MADE FOR LOGIC ERRORS.

E. RESTRICTIONS

NONE

8. AUTO DISPLAY TEST PATTERN DESCRIPTIONS

DISPLAY HORIZONTAL LINE

A HORIZONTAL LINE IS DISPLAYED ON THE SCOPE BY INITIALLY SETTING THE X AND Y DAC'S TO ZERO AND THEN INCREMENTING THE X VALUE WHILE HOLDING THE Y VALUE AT 4000. THE POINTS ARE DISPLAYED USING THE DISPLAY INTERRUPT ENABLED.

DISPLAY VERtICAL LINE

A VERTICAL LINE IS DISPLAYED ON THE SCOPE IN THE SAME MANNER AS FOR A HORIZONTAL LINE EXCEPT NOW THE Y VALUE IS INCREMENTED WHILE HOLDING THE X VALUE AT 1000.

DISPLAY SQUARE

A SQUARE IS DISPLAYED BY INITIALLY SETTING THE X AND Y VALUES TO NEGATIVE FULL SCALE, THEN X IS INCREMENTED TO POSITIVE SCALE (BOTTOM LINE) THEN Y IS INCREMENTED TO POSITIVE FULL SCALE (RIGHT LINE) THEN X IS DECrEMENTED TO NEGATIVE FULL SCALE (TOP LINE) AND FINALLY Y IS DECrEMENTED TO NEGATIVE FULL SCALE (LEFT LINE). MODE 01 (INTENSIFY ON LOADING X) AND MODE 10 (INTENSIFY ON LOADING Y) ARE USED.

DISPLAY X

AN X IS DISPLAYED BY INITIALLY SETTING THE X AND Y VALUES TO NEGATIVE FULL SCALE AND THEN INCREMENTING BOTH TO POSITIVE FULL SCALE (LOWER LEFT TO UPPER RIGHT DIAGONAL) THEN X IS RESET TO NEGATIVE FULL SCALE, Y REMAINS AT POSITIVE FULL SCALE AND THEN X IS INCREMENTED WHILE Y IS DECREMENTED UNTIL BOTH REACH FULL SCALE AGAIN (UPPER LEFT TO LOWER RIGHT DIAGONAL). MODE 01 (INTENSIFY ON LOADING X) IS USED.

9. MISCELLANEOUS

9.1 AA11-K BUS & VECTOR ADDRESS MODIFICATION

MODIFY LOCATION '$BASE' IF BASE ADDRESS IS NOT 170416.
MODIFY LOCATION '$VECT1' IF THE VECTOR AND PRIORITY IS NOT 100360.

*NOTE IF EITHER VALUE IS CHANGED, THE PROGRAM MUST BE RESTARTED AT 200.

9.2 XXDP/APT NOTES

THIS DIAGNOSTIC IS CHAINABLE UNDER XXDP.
THIS DIAGNOSTIC HAS THE "APT" HOOKS BUT HAS NOT BEEN TESTED.

9.3 POWER FAIL

A POWER FAIL WILL CAUSE A RESTART MESSAGE ON POWER UP AT
WHICH TIME THE PROGRAM IS RESTARTED.

9.4 MULTIPLE AA11-K INTERFACE TESTING

THE PROGRAM WILL "AUTO-SIZE" THE NUMBER OF AA11-K'S. THE PROGRAM WILL REPORT THE
NUMBER TO THE OPERATOR WHEN THE "AL" TEST IS SELECTED THE FIRST TIME.
IF THE OPERATOR WISHES TO INHIBIT "AUTO-SIZE" BIT 15 OF LOCATION "$ENV" MUST BE SET.

9.5 RESTRICTION

POWER MUST BE APPLIED TO A STORAGE SCOPE IF CONNECTED.

9.6 EXECUTION TIME

1. EXECUTION TIME OF THE AUTO LOGIC TEST IS:
   25 SECONDS WITHOUT INTERATIONS.
   50 SECONDS WITH INTERATIONS.
2. EXECUTION TIME OF THE AUTO DISPLAY TEST IS:
   180 SECONDS
3. EXECUTION TIME OF THE MANUAL LOGIC LOOP IS:
   OPERATOR DEPENDANT
4. EXECUTION TIME OF THE MANUAL DISPLAY LOOP IS:
   OPERATOR DEPENDANT
5. EXECUTION TIME OF THE MANUAL CALIBRATION LOOP IS:
   OPERATOR DEPENDANT

9.7 USER LINK TO I/O DEVICE

A SPECIAL USER LINK HAS BEEN PROVIDED IN ORDER FOR THE
OPERATOR TO eXAMINE OR MODIFY LOCATIONS ON THE LPA11-KX
I/O BUS. (NOTE: THIS CANNOT BE DONe DIRECTLY.)

PROCEDURE:

1) START THE PROCESSOR AT LOCATION $UTK:

2) THE DIALOG TO EXAMINE A LOCATION IS AS FOLLOWS:

   E OR D "E"

```
DEVICE ADDRS=    "OCTAL ADDRS"
XXXXXX
```

WHERE XXXXXX IS THE CONTENTS OF THE SPECIFIED LOC.

3) THE DIALOG TO MODIFY A LOCATION IS AS FOLLOWS:

```
E OR D              "D"
DATA=               "DATA TO BE DEPOSITED"
```

4) THE PROGRAM WILL STAY IN THIS LOOP UNTIL THE OPERATOR IS FINISHED. AT THIS TIME THE PROCESSOR SHOULD BE HALTED.

NOTE: THE OPERATORS RESPONSE IS ENCLOSE IN QUOTES.

10. LPA11 (SYSTEM) DIAGNOSTIC SUMMARY

DIAGNOSTICS FOR THE LPA11 ARE WRITTEN AT THREE LEVELS: (1) TOTAL PDP-11 SYSTEM, (2) LPA11 SYSTEM; AND, (3) LPA11 OPTIONS.

LEVEL 1, IS DESIGNED TO ISOLATE A FAILURE TO THE LPA11 SYSTEM. ALL OPTIONS ON THE PDP-11 ARE EXERCISED.

LEVEL 2 DIAGNOSTICS ISOLATE A FAILURE TO THE INDIVIDUAL OPTION WITHIN THE LPA11. THE LEVEL 2 DIAGNOSTIC IS MD-11-DRLPA. WHEN THE USER RUNS DRLPA HE CAN GENERALLY TELL WHICH OPTION DIAGNOSTIC (LEVEL 3) TO RUN NEXT. M8254 AND M8200-YC ERRORS MAY "LOOK" ALIKE AND DRLPA MAY NOT BE ABLE TO DISTINGUISH BETWEEN THEM. ARBITRATION ERRORS WILL NOT BE DETECTED BY THIS DIAGNOSTIC.

LEVEL THREE DIAGNOSTICS AID IN DETERMINING IF THE ERROR WAS IN FACT ON THE OPTION THE DRLPA SPECIFIED. THE USER MAY "LOOP" ON THE ERROR. WITHIN LEVEL THREE, THERE ARE TWO GROUPS OF DIAGNOSTICS. THE FIRST GROUP REQUIRES NO "EXTRA" WORK BY THE USER IN ORDER TO RUN. GROUP "A" DIAGNOSTICS DO NOT CHECK ARBITRATION, AND REQUIRE EXTRA TIME FOR EXECUTION. THE SECOND GROUP (GROUP "B") REQUIRES THAT THE USER RECONFIGURE THE PDP-11 SYSTEM. THIS RECONFIGURATION INVOLVES CABLING THE UNIBUS TO THE LPA'S I/O BUS.

THE DIAGNOSTIC FOR THE M8254 FALLS INTO THE GROUP "B" CATAGORY.

K01

SEQ 0010

THE LPA11-KX DIAGNOSTIC KIT WILL INCLUDE:

OPTION	GROUP	DIAG. #	DIAG. TITLE
LPA11-KX	LEVEL 2	MD-11-DRLPA	LPA11-K SYSTEM DIAG.
M8254	"B"	MD-11-DRLPN	M8254 (IPBM) DIAG.
AA11-K	A	MD-11-DRLPB	AA11-K DIAG.
	B	MD-11-DZAAC	AA11-K DIAG.
AR11	A	MD-11-DRLPC	LPA/AR11 DIAG. #1
	A	MD-11-DRLPD	LPA/AR11 DIAG. #2
	A	MD-11-DRLPE	LPA/AR11 DIAG. #3
	B	MD-11-DZARA	AR11 DIAG. #1
	B	MD-11-DZARB	AR11 DIAG. #2
	B	MD-11-DZARC	AR11 DIAG. #3
DR11-K	A	MD-11-DRLPF	LPA/DR11-K DIAG.
	B	MD-11-DZDRG	DR11-K DIAG.
KW11-K	A	MD-11-DRLPG	LPA/KW11-K DIAG.
	B	MD-11-DZKWK	KW11-K DIAG.
LPS11	A	MD-11-DRLPH	LPA/LPS11 DIAG. #1
	A	MD-11-DRLPI	LPA/LPS11 DIAG. #2
	A	MD-11-DRLPJ	LPA/LPS11 DIAG. #3
	B	MD-11-DZLPC	LPS11 DIAG. #1
	B	MD-11-DZLPD	LPS11 DIAG. #2
	B	MD-11-DZLPI	LPS11 DIAG. #3
AD11-K	A	MD-11-DRLPK	LPA/AD11-K DIAG.
	B	MD-11-DZADL	AD11-K DIAG.
M8200-YC	B	MD-11-DZLPL	LPA/M8200-YC BASIC MICRO-CPU R/W TEST
	B	MD-11-DZLPM	LPA/M8200-YC JMP+ROM READ TEST

```
                        .REM    [

                                LPA.MAC

WELCOME, THIS DIAGNOSTIC IS ONE IN A SERIES OF DIAGNOSTIC
DESiGNED IN ORDER TO AID YOU IN TESTING THE LPA-11XX OPTION.
I HOPE THAT YOU HAVE READ THE DOCUMENTATION SECTION OF THIS
DIAGNOSTIC.  IF YOU HAVE,YOU KNOW ABOUT ALL OF THE DIAGNOSTICS
THAT ARE AVAILIBLE FOR TESTING THE LPA SYSTEM.

                                GOOD LUCK !

[
.GLOBL  DRLPX2
```

```
                                .REM    !

                                THIS IS A LIST OF TESTS DELETED FROM THIS DIAGNOSTIC. THESE TESTS COULD NOT
                                BE DONE THROUGH THE LPA-11

                                TEST THAT THE LOW BYTE OF STATUS REG CAN BE CLEARED
                                TEST THAT THE HIGH BYTE OF STATUS REG CAN BE CLEARED
                                TEST THAT DISPLAY DOES INTR. AT LEVEL INDICATED
                                TEST THAT DISPLAY DOES NOT INTR. AT LEVEL INDICATED
                                TEST THE SCOPE SETTLING TIME

                                !
                            .TITLE  MAINDEC-11-DRLPB-A      AA11-K  DIAGNOSTIC
                            ;*COPYRIGHT (C) 1978
                            ;*DIGITAL EQUIPMENT CORP.
                            ;*MAYNARD, MASS. 01754
                            ;*
                            ;*PROGRAM BY EDWARD C. BADGER
                            ;*
                            ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                            ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                            ;*
                            .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

170416                                  ABASE=170416
100360                                  AVECT1=100360
000200                                  APRIOR=200

                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        12            STORAGE SCOPE CONNECTED
                                ;*        11            INHIBIT ITERATIONS
                                ;*        10            EXTERNAL DELAY SIGNAL CONNECTED
                                ;*         8            LOOP ON TEST IN SWR<7:0>
                                ;*         9            TWO'S COMPLEMENT MODE
                                .SBTTL  TRAP CATCHER

000000                                  .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
000174                                  .=174
000174  000000                 DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                 SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                .SBTTL  STARTING ADDRESS(ES)
000200  000137  001554                 JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  001572                 JMP     BEGIN1                  ;JUMP TO RESTART ADDRESS
000210  000137  014104                 JMP     DYNCAL                  ;JUMP TO DYNAMIC DAC CALIBRATION
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;**********************************************************
                                ;HOOKS REQUIRED BY ACT11
000214                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  007320                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO zERO
000214                                  .=$SVPC                 ;; RESTORE PC
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  001174                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000030                  $TSTM:  .WORD   30      ;;RUN TIM OF LONGEST TEST
001006  000060                  $PASTM: .WORD   60      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000120                  $UNITM: .WORD   120     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000031                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

                        001100                  .=1100
001100                  000000  $CMTAG:                         ;;START OF COMMON TAGS
                                        .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                  000000  $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                                ;;WHICH  ($REG0) WAS OBTAINED
001162                  000000  $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164                  000000  $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001166                  000000  $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001170                  077     $QUES:  .ASCII  /?/             ;;QUESTION MARK
001171                  015     $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001172                  000012  $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;*****************************************************************
                                .EVEN
001174                  $MAIL:                          ;;APT MAILBOX
001174                  000000  $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
001176                  000000  $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
001200                  000000  $TESTN: .WORD   ATESTN  ;;TEST NUMBER
```

```
                                                $PASS:  .WORD   APASS   ;;PASS COUNT
                                                $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
                                                $UNIT:  .WORD   AUNIT   ;;I/O UNIt NUMBER
                                                $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
                                                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
                                                $ETABLE:                ;;APT ENVIRONMENT TABLE
                                                $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
                                                $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
                                                $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
                                                $USWR:  .WORD   AUSWR   ;;USER SWITCHES
                                                $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                                                ;*                      BITS 15-11=CPU TYPE
                                                ;*                           11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                                ;*                           11/70=06,PDQ=07,Q=10
                                                ;*                      BIT 10=REAL TIME CLOCK
                                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                                ;*                      BIT  8=MEMORY MANAGEMENT
                                                $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
                                                $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                                ;*                           900 NSEC CORE=001
                                                ;*                           300 NSEC BIPOLAR=002
                                                ;*                           500 NSEC MOS=003
                                                $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
                                                $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
                                                $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
                                                $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
                                                $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
                                                $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
                                                $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
                                                $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
                                                $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
                                                $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
                                                $VECT1: .WORd   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
                                                $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
                                                $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TESt
                                                $DEVm:  .WORD   ADEVM   ;;DEVICE MAP
                                                $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
                                                $ETEND:
                                                .MEXIT
```

```
                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:      EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                ;*      DH              ;;POINTS TO THE DATA HEADER
                                ;*      DT              ;;POINTS TO THE DATA
                                ;*      DF              ;;POINTS TO THE DATA FORMAT


001256                          $ERRTB:


                                ;ITEM   1
001256  016547                          EM1                     ;AA11-K STATUS REGISTER IN ERROR
001260  017370                          DH1                     ;ERRPC  IBASE   GOOD    BAD
001262  021006                          DT1                     ;$ERRPC STAT    $GDDAT  $BDDAT
001264  021122                          DF0

                                ;ITEM   2
001266  016600                          EM2                     ; DAC0 REGISTER IN ERROR
001270  017370                          DH1                     ;ERRPC  IBASE   GOOD    BAD
001272  021020                          DT2                     ;$ERRPC DAC0    $GDDAT  $BDDAT
001274  021122                          DF0

                                ;ITEM   3
001276  016627                          EM3                     ; DAC1 REGISTER IN ERROR
001300  017370                          DH1                     ;ERRPC  IBASE   GOOD    BAD
001302  021032                          DT3                     ;$ERRPC DAC1    $GDDAT  $BDDAT
001304  021122                          DF0

                                ;ITEM   4
001306  016656                          EM4                     ;DAC #2 REGISTER IN ERROR
001310  017370                          DH1                     ;ERRPC  IBASE   GOOD    BAD
001312  021044                          DT4                     ;$ERRPC DAC2    $GDDAT  $BDDAT
001314  021122                          DF0

                                ;ITEM   5
001316  016705                          EM5                     ;DAC #3 REGISTER IN ERROR
001320  017370                          DH1                     ;ERRPC  IBASE   GOOD    BAD
001322  021056                          DT5                     ;$ERRPC DAC3    $GDDAT  $BDDAT
001324  021122                          DF0

                                ;ITEM   6
001326  016734                          EM6                     ;AA11-K FAILED TO INTERRUPT
001330  017424                          DH6                     ;ERRPC  IBASE
001332  021070                          DT6                     ;$ERRPC STAT
001334  021122                          DF0

                                ;ITEM   7
```

```
                                                EM7             ;AA11-K INTERRUPTED IN ERROR
001336  016767                                  DH6             ;ERRPC  IBASE
001340  017424                                  DT6             ;$ERRPC STAT
001342  021070                                  DF0
001344  021122
                                        ;ITEM   10
                                                EM10            ;BUS TIME-OUT ON AA11-K ADDRESSES
001346  017023                                  DH6             ;ERRPC  IBASE
001350  017424                                  DT6             ;$ERRPC STAT
001352  021070                                  DF0
001354  021122
                                        ;ITEM   11
                                                EM11            ;AA11-K READY BIT ERROR
001356  017074                                  DH1             ;ERRPC  IBASE   GOOD    BAD
001360  017370                                  DT1             ;$ERRPC STAT    $GDDAT  $BDDAT
001362  021006                                  DF0
001364  021122
                                        ;ITEM   12
                                                EM12            ;POWER SUPPLY VOLTAGE WAS INCORRECT
001366  017123                                  DH14            ;ERRPC  BASE    GOOD    BAD
001370  017472                                  DT14            ;$ERRPC $BASE   $GDDAT  $BDDAT
001372  021110                                  DF0
001374  021122
                                        ;ITEM   13
                                                EM13            ;A PREVIOUSLY EXISTING AA11-K DOES NOT RESPOND NOW
001376  017166                                  DH13            ;ERRPC  BASE    FIRST#  #NOW
001400  017440                                  DT13            ;$ERRPC $BASE   EVER    $UNIT
001402  021076                                  DF0
001404  021122
                                        ;ITEM   14
                                                EM14            ;AA11-K LOGIC SIGNAL HIGH OUTPUT TO LOW
001406  017207                                  DH14            ;ERRPC  BASE    GOOD    BAD
001410  017472                                  DT14            ;$ERRPC $BASE   $GDDAT  $BDDAT  SPREAD
001412  021110                                  DF0
001414  021122
                                        ;ITEM   15
                                                EM15            ;AA11-K LOGIC LOW OUTPUT TO HIGH
001416  017256                                  DH14            ;ERRPC  BASE    GOOD    BAD
001420  017472                                  DT14            ;$ERRPC $BASE   $GDDAT  $BDDAT  SPREAD
001422  021110                                  DF0
001424  021122
                                        ;ITEM   16
                                                EM16            ;SELECTED DAC HAS A LINEARITY ERROR
001426  017325                                  DH14            ;ERRPC  BASE    GOOD    BAD
001430  017472                                  DT14            ;$ERRPC $BASE   $GDDAT  $BDDAT
001432  021110                                  DF0
001434  021122
                                                ;
                                                ;ADDRESS OF KMC-11 OF LPA-11    THE ADDR FOR KMADO MAY BE
                                                ;                               CHANGED BY THE USER TO REFLECT
                                                ;                               A DIFFERENT KMC-11 ADDR. THE
                                                ;                               REST OF THE ADDRESSES WILL
                                                ;                               BE CHANGED BY THE PROGRAM.
                                                ;
```

```
                                LPCI:
001436                  KMAD0:  .WORD   170460          ;BASE KMC ADDR. MAY BE PATCHED BY USER.
001436  170460
                                LPMR:
001440                  KMAD1:  .WORD   170460+1                ;>DO NOT        <;KMC-CSR ADDR
001440  170461          LPCO:
001442                  KMAD2:  .WORD   170460+2                ;>PATCH         <;
001442  170462          LPSO:
001444                  KMAD3:  .WORD   170460+3                ;>THIS AREA     <
001444  170463          LPADL:
001446                  KMAD4:  .WORD   170460+4                ;
001446  170464          LPADH:
001450                  KMAD5:  .WORD   170460+5                ;>DO NOT        <
001450  170465          LPMS1:
001452                  KMAD6:  .WORD   170460+6                ;>PATCH         <
001452  170466          LPMS2:
001454                  KMAD7:  .WORD   170460+7                ;>THIS AREA     <
001454  170467
001456  000360  VECTOR: .WORD   AVECT1&777      ;BASE VECTOR OF KMC
001460  000364  VECTPS: .WORD   4+AVECT1&777    ;VECOTR ADDR.+2
001462  000004  VERSN:  .WORD   4                ;CURRENT VERSION NUMBER OF MICROCODE.
001464  000000  .DVLS:  .WORD   0                ;/DEVICE LIST OF I/O ADDR. DEFINED
001466  000020          .BLKW   16.             ;/BY INIT.
001526  000040  VADDR:  40              ;ADJUSTMENT TO NEXT AA11-K'S ADDRESS
001530  000040  VVECT:  40              ;ADJUSTMENT TO NEXT AA11-K'S VECTOR
001532  000012  DELAY:  10.             ;DELAY COUNTER FOR DELAY LOOPS
001534  000000  $TMDAT: 0
001536  170416  STAT:   170416
001540  170420  DAC0:   170420
001542  170422  DAC1:   170422
001544  170424  DAC2:   170424
001546  170426  DAC3:   170426
001550  000350  IV:     350
001552  000352  IVS:    352
001554  005037  021154         BEGIN:  CLR     TEMP
001560  005037  021164                 CLR     EVER            ;RESET POINTER
001564  005037  021166                 CLR     EVER1
001570  000403                         BR      BEG
001572  012737  000001  021154 BEGIN1: MOV     #1,TEMP
001600  000240                 BEG:    NOP
                               .SBTTL  INITIALIZE THE COMMON TAGS
                               ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001602  012706  001100                 MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001606  005026                         CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001610  022706  001140                 CMP     #SWR,R6 ;;DONE?
001614  001374                         BNE     .-6             ;;LOOP BACK IF NO
001616  012706  001100                 MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                               ;;INITIALIZE A FEW VECTORS
001622  012737  021422  000020         MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001630  012737  000340  000022         MOV     #340,@#IOTVEC+2 ;;LEVEL 7
```

```
                                        MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
                                        MOV     #340,@#EMTVEC+2 ;;LEVEL 7
                                        MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
                                        MOV     #340,@#TRAPVEC+2;LEVEL 7
                                        MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
                                        MOV     #340,@#PWRVEC+2 ;;LEVEL 7
                                        CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
                                        MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
                                        MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
                                        MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
                                        MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
                                        MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
                                        CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
                                        BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
                                        BR      65$             ;;BRANCH IF NO TIMEOUT
                                64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
                                        RTI
                                65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
                                        MOV     #DISPREG,DISPLAY
                                66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

                                        CLR     $PASS           ;;CLEAR PASS COUNT
                                        BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
                                        BEQ     67$             ;;YES,USE NON-APT SWITCH
                                        MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
                                67$:
                                        ;
                                        ;THIS SECTION OF CODE HANDLES INITIALIZING LPA-11 FUNCTIONS
                                        ;

                                        MOV     R0,-(SP)
                                        MOV     R1,-(SP)
                                        MOV     KMADD,R0        ;GET KMC-11 ADDRESS.
                                        MOV     #KMAD1,R1       ;GET ADDR. OF ADDR. LIST.

                                68$:    INC     R0              ;UPDATE ADDR.
                                        MOV     R0,(1)+         ;WRITE ADDR.
                                        CMP     R1,#KMAD7+2     ;DONE ALL ADDRESSES?
                                        BNE     68$             ;NO - DO NEXT ADDR.
                                        CLR     .DVLS           ;CLR ADDR. LIST.
                                        MOV     (SP)+,R1
                                        MOV     (SP)+,R0

                                        CLR     @#PS
                                        JMP     INIT1
                                .SBTTL  SUBROUTINE TO LOAD A TRAP CATCHER
                                LDTRAP: MOV     #242,R2                 ;LOAD R2
                                        MOV     #240,R1                 ;LOAD R1
                                5$:     MOV     R2,(R1)+                ;LOAD .+2
                                        CLR     (R1)+                   ;LOAD HALT
                                        MOV     R1,R2                   ;LOAD R2
```

```
                                        TST     (R2)+                   ;BUMP R2
                                        CMP     R2,#1002                ;TEST FOR LAST
                                        BNE     5$                      ;BR UNTIL DONE
                                .SBTTL  LOAD DEVICE ADDRESSES LOCATIONS
                                        MOV     #STAT,R0                ;LOAD POINTER
                                10$:    MOV     $BASE,(R0)+             ;LOAD BASE ADDRESS
                                        CMP     #IV,R0                  ;TEST FOR DONE
                                        BNE     10$                     ;BR
                                        MOV     #DACO,R0                ;LOAD 2ND ADDRESS
                                        MOV     #2,R1                   ;LOAD R1
                                12$:    ADD     R1,(R0)+                ;UPDATE REAL DEVICE WORD
                                        TST     (R1)+                   ;BUMP R1
                                        CMP     #12,R1                  ;TEST FOR DONE
                                        BNE     12$                     ;BR
                                        MOV     $VECT1,IV               ;LOAD INTR. VECTOR ADDRESS
                                        BIC     #160000,IV
                                        MOV     IV,IVS
                                        ADD     #2,IVS
                                        MOVB    $VECT1+1,BRLEV1         ;LOAD BR LEVEL
                                        BIC     #177437,BRLEV1
                                        MOV     BRLEV1,BRLEV2
                                        SUB     #40,BRLEV2
                                        RTS     PC                      ;EXIT

                                .SBTTL  INITIAL HEADER TYPEOUT AND WAIT FOR OPERATOR
                                AL:     .ASCII  /AL/
                                AD:     .ASCII  /AD/
                                ML:     .ASCII  /ML/
                                MD:     .ASCII  /MD/
                                MC:     .ASCII  /MC/

                                INIT1:  JSR     PC,LDTRAP
                                        TST     TEMP                    ;TEST IF START OR RESTART
                                        BNE     CTRLC                   ;RESTART
                                        TST     @#42                    ;TEST IF MONITOR
                                        BNE     MTEST                   ;BR IF NOT
                                        TYPE                            ;CALL MESSAGE PRINTER VIA 'EMT'
                                        TITLE                           ;TYPE PROGRAM HEADER.
                                WAITIN: MOV     #STACK,SP               ;RESET STACK
                                        CKSWR                           ;TEST FOR CTRL G
                                        RDLIN                           ;READ TWO CHARACTERS
                                        MOV     @(SP)+,OPRIN            ;READ THE CHARACTER
                                        BIC     #0,OPRIN                ;MASK OTHER CHARACTERS

                                        CMP     AL,OPRIN                ;TEST FOR "AL"
                                        BEQ     10$                     ;BR IF YES
                                        CMP     AD,OPRIN                ;TEST FOR "AD"
                                        BEQ     11$                     ;BR IF YES
                                        CMP     ML,OPRIN                ;TEST FOR "ML"
                                        BEQ     13$                     ;BR FI YES
                                        CMP     MD,OPRIN                ;TEST FOR "MD"
                                        BEQ     14$                     ;BR IF YES
                                        CMP     MC,OPRIN                ;TEST FOR "MC"
                                        BEQ     15$                     ;BR IF YES
```

```
                                                TYPE
                                                QMARK
                                                BR      WAITIN          ;WAIT FOR OPERATOR AGAIN

                                        10$:    JMP     MTEST           ;RUN THE LOGIC TEST
                                        11$:    JMP     VISUAL          ;RUN THE VISUAL PATTERN
                                        12$:    JMP     AUTCAL          ;RUN THE AUTO CALIBRATION (FACTORY ONLY)
                                        13$:    JMP     MANUL           ;RUN THE MANUAL LOGIC TEST
                                        14$:    JMP     FULRMP          ;RUN THE RAMP PATTERN ON FOUR DAC'S
                                        15$:    JMP     CALDAC          ;RUN THE MANUAL CAL OF THE DAC

                                CTRLC:  TYPE
                                        CONTC
                        001202          CLR     $PASS
                                        BR      WAITIN

                                        .SBTTL  DETERMINE THE NUMBER OF AA11-K ON THIS SYSTEM

                        001126  MTEST:  MOV     $BASE,$BDDAT    ;GET THE BASE ADDRESS
                                        CLR     .DVLS
                                        CLR     $UNIT           ;CLEAR UNIT #
                                1$:
                                ;*      MOV     @$BDDAT,$TMDAT  ;/READ DEVICE REG $BDDAT,PUT DATA IN $TMDAT.
                                        TST     $AERR
                                        BNE     2$
                        001126          ADD     VADDR,$BDDAT    ;UPDATE THE BUS ADDRESS
                                        INC     $UNIT           ;UPDATE UNIT COUNT
                                        BR      1$
                                2$:     NOP
                                        TST     $UNIT           ;TEST IF ANY EXIST
                                        BNE     3$              ;BR IF SOME ARE THERE
                                        ERROR   10              ;BASE ADDRESS CAUSED AN BUS TRAP
                                        BR      TST1            ;;
                                3$:     TST     EVER            ;TEST IF # HAS BEEN REPORTED
                                        BMI     4$              ;BR IF IT HAS
                                        TYPE
                                        FOUND1                  ;TELL OPERATOR THE # OF AA11-K'S
                        001206          MOV     $UNIT,-(SP)
                                        TYPOS
                                        .BYTE   2
                                        .BYTE   0
                                        TYPE
                                        FOUND2
                        021164          MOV     $UNIT,EVER      ;SAVE THE # OF AA11-K'S FOR LATER
                        021164          BIS     #BIT15,EVER     ;SET "REPORTED # FLAG"
                                        BR      5$              ;;
                        001206  4$:     CMPB    EVER,$UNIT      ;TEST IF ANY HAVE GONE AWAY
                                        BEQ     5$              ;;BR IF ALL ARE STILL HERE
                                        ERROR   13              ;EXISTING UNIT FAILED TO RESPOND NOW
                        001206  5$:     CLR     $UNIT           ;RESET UNIT POINTER
                        002072          JSR     PC,LDTRAP       ;LOAD TRAP CATCHER AND BUS ADDRESSES

                                LTEST:  NOP
```

```
                                ;;*****************************************************************
                                ;*TEST 1         TEST THAT THE AA11-K RESPONDS TO THE CPU
                                ;;*****************************************************************
002616  000004                  TST1:   SCOPE
002620  012737  002634  001106          MOV     #3$,$LPADR
002626  012737  002732  000004          MOV     #1$,ERRVEC      ;LOAD BUS TRAP RETURN
002634                          3$:
                                ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
002644  005737  001534                  TST     $TMDAT
                                ;*      MOV     @DAC0,$TMDAT    ;/READ DEVICE REG DAC0,PUT DATA IN $TMDAT.
002660  005737  001534                  TST     $TMDAT
                                ;*      MOV     @DAC1,$TMDAT    ;/READ DEVICE REG DAC1,PUT DATA IN $TMDAT.
002674  005737  001534                  TST     $TMDAT
                                ;*      MOV     @DAC2,$TMDAT    ;/READ DEVICE REG DAC2,PUT DATA IN $TMDAT.
002710  005737  001534                  TST     $TMDAT
                                ;*      MOV     @DAC3,$TMDAT    ;/READ DEVICE REG DAC3,PUT DATA IN $TMDAT.
002724  005737  001534                  TST     $TMDAT
002730  000402                        BR      2$              ;;BR AND RESTORE LOC. 4
002732  022626                  1$:     CMP     (SP)+,(SP)+     ;CLEAN THE STACK
002734  104010                          ERROR   10              ;ERROR, BUS TIMEOUT WHEN ADDRESSING THE AA11-K
002736  012737  000006  000004  2$:     MOV     #6,ERRVEC       ;LOAD RETURN
                                ;;*****************************************************************
                                ;*TEST 2         TEST THAT THE DAC0 REGISTER CAN BE CLEARED
                                ;;*****************************************************************
002744  000004                  TST2:   SCOPE
002746  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED
                                ;*      MOV     $GDDAT,@DAC0    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC0
                                ;*      MOV     @DAC0,$BDDAT    ;/READ DEVICE REG DAC0,PUT DATA IN $BDDAT.
002772  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003000  001401                        BEQ     TST3            ;;BR IF EQUAL
003002  104002                        ERROR   2                       ;ERROR, DAC0 REGISTER NOT = 0
                                ;;*****************************************************************
                                ;*TEST 3         TEST THAT THE DAC0 REGISTER CAN BE LOADED WITH #7777
                                ;;*****************************************************************
003004  000004                  TST3:   SCOPE
003006  012737  007777  001124          MOV     #7777,$GDDAT            ;LOAD EXPECTED
                                ;*      MOV     $GDDAT,@DAC0    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC0
                                ;*      MOV     @DAC0,$BDDAT    ;/READ DEVICE REG DAC0,PUT DATA IN $BDDAT.
003034  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003042  001401                        BEQ     TST4            ;;BR IF EQUAL
003044  104002                        ERROR   2                       ;ERROR,  DAC0 REGISTER NOT = 7777
                                ;;*****************************************************************
                                ;*TEST 4         TEST THAT THE DAC0 REGISTER CAN HOLD A FLOATING 1 PATTERN
```

```
                                ;;*****************************************************************
003046  000004                 TST4:   SCOPE
003050  012737  000100  001166         MOV     #100,$TIMES     ;;DO 100 ITERATIONS
003056  012737  004000  001124         MOV     #BIT11,$GDDAT           ;LOAD EXPECTED
003064                          1$:
                                ;*      MOV     $GDDAT,@DAC0    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC0
                                ;*      MOV     @DAC0,$BDDAT    ;/READ DEVICE REG DAC0,PUT DATA IN $BDDAT.
003104  023737  001124  001126         CMP     $GDDAT,$BDDAT           ;COMPARE THE DATA
003112  001401                         BEQ     2$              ;;BR IF SAME
003114  104002                         ERROR   2                       ;ERROR, DAC0 REGISTER FAILED TO HOLD A FLOATING
003116  006237  001124          2$:    ASR     $GDDAT                  ;CHANGE THE DATA
003122  001360                         BNE     1$                      ;BR AND TEST MORE DATA
                                ;;*****************************************************************
                                ;*TEST 5         TEST THAT THE DAC #1 REGISTER CAN BE CLEARED
                                ;;*****************************************************************
003124  000004                 TST5:   SCOPE
003126  005037  001124                 CLR     $GDDAT                  ;LOAD EXPECTED
                                ;*      MOV     $GDDAT,@DAC1    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC1
                                ;*      MOV     @DAC1,$BDDAT    ;/READ DEVICE REG DAC1,PUT DATA IN $BDDAT.
003152  023737  001124  001126         CMP     $GDDAT,$BDDAT           ;COMPARE
003160  001401                         BEQ     TST6            ;;BR IF EQUAL
003162  104003                         ERROR   3                       ;ERROR,  DAC1 REGISTER NOT = 0
                                ;;*****************************************************************
                                ;*TEST 6         TEST THAT THE Y REGISTER CAN BE LOADED WITH #7777
                                ;;*****************************************************************
003164  000004                 TST6:   SCOPE
003166  012737  007777  001124         MOV     #7777,$GDDAT            ;LOAD EXPECTED
                                ;*      MOV     $GDDAT,@DAC1    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC1
                                ;*      MOV     @DAC1,$BDDAT    ;/READ DEVICE REG DAC1,PUT DATA IN $BDDAT.
003214  023737  001124  001126         CMP     $GDDAT,$BDDAT           ;COMPARE
003222  001401                         BEQ     TST7            ;;BR IF EQUAL
003224  104003                         ERROR   3                       ;ERROR,  DAC1 REGISTER NOT = 7777
                                ;;*****************************************************************
                                ;*TEST 7         TEST THAT THE Y REGISTER CAN HOLD A FLOATING 1 PATTERN
                                ;;*****************************************************************
003226  000004                 TST7:   SCOPE
003230  012737  000100  001166         MOV     #100,$TIMES     ;;DO 100 ITERATIONS
003236  012737  004000  001124         MOV     #BIT11,$GDDAT           ;LOAD EXPECTED
003244                          1$:
                                ;*      MOV     $GDDAT,@DAC1    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC1
                                ;*      MOV     @DAC1,$BDDAT    ;/READ DEVICE REG DAC1,PUT DATA IN $BDDAT.
003264  023737  001124  001126         CMP     $GDDAT,$BDDAT           ;COMPARE THE DATA
003272  001401                         BEQ     2$              ;;BR IF DATA IS SAME
003274  104003                         ERROR   3                       ;ERROR, DAC1 REGISTER FAILED TO HOLD A FLOATING
```

```
                                ;;*****************************************************************
                                ;*TEST 10        TEST THAT THE DAC #2 REGISTER CAN BE CLEARED
                                ;;*****************************************************************
003304  000004          TST10:  SCOPE
003306  005037  001124          CLR     $GDDAT                  ;LOAD EXPECTED

                        ;*      MOV     $GDDAT,@DAC2    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC2

                        ;*      MOV     @DAC2,$BDDAT    ;/READ DEVICE REG DAC2,PUT DATA IN $BDDAT.
003332  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003340  001401                        BEQ     TST11           ;;BR IF EQUAL
003342  104004                        ERROR   4               ;ERROR, DAC #2 REGISTER NOT = 0

                                ;;*****************************************************************
                                ;*TEST 11        TEST THAT THE DAC #2 REGISTER CAN BE LOADED WITH #7777
                                ;;*****************************************************************
003344  000004          TST11:  SCOPE
003346  012737  007777  001124          MOV     #7777,$GDDAT            ;LOAD EXPECTED

                        ;*      MOV     $GDDAT,@DAC2    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC2

                        ;*      MOV     @DAC2,$BDDAT    ;/READ DEVICE REG DAC2,PUT DATA IN $BDDAT.
003374  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003402  001401                        BEQ     TST12           ;;BR IF EQUAL
003404  104004                        ERROR   4               ;ERROR, DAC #2 REGISTER NOT = 7777

                                ;;*****************************************************************
                                ;*TEST 12        TEST THAT THE DAC #2 REGISTER CAN HOLD A FLOATING 1 PATTERN
                                ;;*****************************************************************
003406  000004          TST12:  SCOPE
003410  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
003416  012737  004000  001124          MOV     #BIT11,$GDDAT           ;LOAD EXPECTED
003424                  1$:

                        ;*      MOV     $GDDAT,@DAC2    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC2

                        ;*      MOV     @DAC2,$BDDAT    ;/READ DEVICE REG DAC2,PUT DATA IN $BDDAT.
003444  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE DATA
003452  001401                        BEQ     2$              ;;BR IF SAME
003454  104004                        ERROR   4               ;ERROR, DAC #2 REGISTER FAILED TO HOLD A FLOATIN
003456  006237  001124          2$:   ASR     $GDDAT                  ;CHANGE THE DATA
003462  001360                        BNE     1$              ;BR AND TEST MORE DATA

                                ;;*****************************************************************
                                ;*TEST 13        TEST THAT THE DAC #3 REGISTER CAN BE CLEARED
                                ;;*****************************************************************
003464  000004          TST13:  SCOPE
003466  005037  001124          CLR     $GDDAT                  ;LOAD EXPECTED

                        ;*      MOV     $GDDAT,@DAC3    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC3
```

```
                                ;*      MOV     @DAC3,$BDDAT    ;/READ DEVICE REG DAC3,PUT DATA IN $BDDAT.
003512  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003520  001401                         BEQ     TST14           ;;BR IF EQUAL
003522  104005                         ERROR   5               ;ERROR, DAC #3 REGISTER NOT = 0

                                ;;**************************************************************
                                ;*TEST 14        TEST THAT THE DAC #3 REGISTER CAN BE LOADED WITH #7777
                                ;;**************************************************************
003524  000004                  TST14:  SCOPE
003526  012737  007777  001124          MOV     #7777,$GDDAT            ;LOAD EXPECTED

                                ;*      MOV     $GDDAT,@DAC3    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC3

                                ;*      MOV     @DAC3,$BDDAT    ;/READ DEVICE REG DAC3,PUT DATA IN $BDDAT.
003554  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003562  001401                         BEQ     TST15           ;;BR IF EQUAL
003564  104005                         ERROR   5               ;ERROR, DAC #3 REGISTER NOT = 7777

                                ;;**************************************************************
                                ;*TEST 15        TEST THAT THE DAC #3 REGISTER CAN HOLD A FLOATING 1 PATTERN
                                ;;**************************************************************
003566  000004                  TST15:  SCOPE
003570  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
003576  012737  004000  001124          MOV     #BIT11,$GDDAT           ;LOAD EXPECTED
003604                          1$:

                                ;*      MOV     $GDDAT,@DAC3    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC3

                                ;*      MOV     @DAC3,$BDDAT    ;/READ DEVICE REG DAC3,PUT DATA IN $BDDAT.
003624  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE DATA
003632  001401                         BEQ     2$              ;;BR IF SAME
003634  104005                         ERROR   5               ;ERROR, DAC #3 REGISTER FAILED TO HOLD A FLOATIN
003636  006237  001124          2$:     ASR     $GDDAT                  ;CHANGE THE DATA
003642  001360                         BNE     1$                      ;BR AND TEST MORE DATA

                                ;;**************************************************************
                                ;*TEST 16        TEST THAT THE FOUR DAC REGISTERS CAN HOLD DIFFERENT DATA
                                ;;**************************************************************
003644  000004                  TST16:  SCOPE
003646  012737  001111  001534          MOV     #1111,$TMDAT            ;LOAD DAC #0

                                ;*      MOV     $TMDAT,@DAC0    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC0
003664  012737  002222  001534          MOV     #2222,$TMDAT            ;LOAD DAC #1

                                ;*      MOV     $TMDAT,@DAC1    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC1
003702  012737  004444  001534          MOV     #4444,$TMDAT            ;LOAD DAC #2

                                ;*      MOV     $TMDAT,@DAC2    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC2
003720  012737  007777  001534          MOV     #7777,$TMDAT            ;LOAD DAC #3

                                ;*      MOV     $TMDAT,@DAC3    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC3
003736  012737  000000  001534          MOV     #0,$TMDAT               ;CLEAR STATUS

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
```

```
                                                ;LOAD EXPECTED
                                ;*      MOV     @DAC0,$BDDAT    ;/READ DEVICE REG DAC0,PUT DATA IN $BDDAT.
003772  023737  001124  001126         CMP     $GDDAT,$BDDAT           ;COMPARE
004000  001401                        BEQ     1$              ;;BR IF EQUAL
004002  104002                        ERROR   2                       ;ERROR, SELECTED DAC0 IN ERROR

004004  012737  002222  001124  1$:   MOV     #2222,$GDDAT            ;LOAD EXPECTED

                                ;*      MOV     @DAC1,$BDDAT    ;/READ DEVICE REG DAC1,PUT DATA IN $BDDAT.
004022  023737  001124  001126         CMP     $GDDAT,$BDDAT           ;COMPARE
004030  001401                        BEQ     2$              ;;BR IF EQUAL
004032  104003                        ERROR   3                       ;ERROR, SELECTED DAC1 IN ERROR

004034  012737  004444  001124  2$:   MOV     #4444,$GDDAT            ;LOAD EXPECTED

                                ;*      MOV     @DAC2,$BDDAT    ;/READ DEVICE REG DAC2,PUT DATA IN $BDDAT.
004052  023737  001124  001126         CMP     $GDDAT,$BDDAT           ;COMPARE
004060  001401                        BEQ     3$              ;;BR IF SAME
004062  104004                        ERROR   4                       ;ERROR, SELECTED DAC #2 IN ERROR

004064  012737  007777  001124  3$:   MOV     #7777,$GDDAT            ;LOAD EXPECTED

                                ;*      MOV     @DAC3,$BDDAT    ;/READ DEVICE REG DAC3,PUT DATA IN $BDDAT.
004102  023737  001124  001126         CMP     $GDDAT,$BDDAT           ;COMPARE
004110  001401                        BEQ     TST17           ;;BR IF SAME
004112  104005                        ERROR   5                       ;ERROR, SELECTED DAC #3 IN ERROR
                        ;;**************************************************************
                        ;*TEST 17       TEST THAT RESET SETS READY BIT
                        ;;**************************************************************
004114  000004          TST17:  SCOPE
004116  012737  000010  001166         MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004124  012737  000200  001124         MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
004132  004737  026474                JSR     PC,$RESET

                                ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
004146  023737  001124  001126         CMP     $GDDAT,$BDDAT           ;COMPARE
004154  001401                        BEQ     TST20           ;;BR IF SET
004156  104001                        ERROR   1                       ;RESET FAILED TO SET READY

                        ;;**************************************************************
                        ;*TEST 20       TEST THAT FAST INTENSIFY CAN BE SET AND CLEARED
                        ;;**************************************************************
004160  000004          TST20:  SCOPE
004162  012737  000002  001534         MOV     #BIT1,$TMDAT            ;LOAD BIT 1

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
004200  012737  000202  001124         MOV     #BIT7!BIT1,$GDDAT       ;LOAD EXPECTED

                                ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
004216  023737  001124  001126         CMP     $GDDAT,$BDDAT           ;COMAPRE
004224  001401                        BEQ     1$              ;;BR IF SAME
004226  104001                        ERROR   1                       ;ERROR, STATUS NOT = 202
004230  012737  000000  001534  1$:   MOV     #0,$TMDAT               ;CLEAR STATUS REGISTER
```

```
                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
004246  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED

                                ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
004264  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004272  001401                          BEQ     TST21           ;;BR IF SAME
004274  104001                          ERROR   1               ;ERROR, STATUS NOT = 200

                                ;;*****************************************************************
                                ;*TEST 21        TEST THAT MODE BIT 2 CAN BE SET AND CLEARED
                                ;;*****************************************************************
004276  000004                  TST21:  SCOPE
004300  012737  000004  001534          MOV     #BIT2,$TMDAT            ;LOAD DISPLAY STATUS

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
004316  012737  000204  001124          MOV     #BIT7!BIT2,$GDDAT       ;LOAD EXPECTED

                                ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
004334  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004342  001401                          BEQ     1$              ;;BR IF EQUAL
004344  104001                          ERROR   1               ;ERROR, STATUS NOT = 204
004346  012737  000000  001534  1$:     MOV     #0,$TMDAT               ;CLEAR STATUS

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
004364  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED

                                ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
004402  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004410  001401                          BEQ     TST22           ;;BR IF CLEARED
004412  104001                          ERROR   1               ;MODE FAILED TO CLEAR


                                ;;*****************************************************************
                                ;*TEST 22        TEST THAT MODE BIT 3 CAN BE SET
                                ;;*****************************************************************
004414  000004                  TST22:  SCOPE
004416  012737  000010  001534          MOV     #BIT3,$TMDAT            ;LOAD

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
004434  012737  000210  001124          MOV     #BIT7!BIT3,$GDDAT       ;LOAD EXPECTED

                                ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
004452  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004460  001401                          BEQ     1$              ;;BR IF EQUAL
004462  104001                          ERROR   1               ;ERROR, STATUS NOT = 210

004464  012737  000000  001534  1$:     MOV     #0,$TMDAT               ;CLEAR REG.

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
004502  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED

                                ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
004520  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
```

```
                                ;;*****************************************************************
                                ;*TEST 23        TEST THAT EXT. DELAY (BIT 4) CAN BE SET AND CLEARED
                                ;;*****************************************************************
000004                  TST23:  SCOPE
004534  012737  000020  001534          MOV     #BIT4,$TMDAT            ;LOAD STATUS

                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
004552  012737  000220  001124          MOV     #BIT7!BIT4,$GDDAT       ;LOAD EXPCTED

                        ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
004570  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004576  001401                          BEQ     1$              ;;BR IF SAME
004600  104001                          ERROR   1               ;ERROR, EXT. DELAY BIT FAILED TO SET
004602  012737  000000  001534  1$:     MOV     #0,$TMDAT               ;CLEAR BIT 4

                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
004620  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED

                        ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
004636  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004644  001401                          BEQ     TST24           ;;BR IF SAME
004646  104001                          ERROR   1               ;ERROR, EXT. DELAY BIT FAILED TO CLEAR
                                ;;*****************************************************************
                                ;*TEST 24        TEST THAT INTERRUPT ENABLE (BIT 6) CAN BE SET
                                ;;*****************************************************************
000004                  TST24:  SCOPE
004652  012737  000100  001534          MOV     #BIT6,$TMDAT            ;LOAD DISPLAY STATUS

                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
004670  012737  000300  001124          MOV     #BIT7!BIT6,$GDDAT       ;LOAD EXPECTED

                        ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
004706  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004714  001401                          BEQ     TST25           ;;BR IF EQUAL
004716  104001                          ERROR   1               ;ERROR, STATUS NOT = 300

                                ;;*****************************************************************
                                ;*TEST 25        TEST THAT CHANNEL (BIT 9) CAN BE SET
                                ;;*****************************************************************
000004                  TST25:  SCOPE
004722  012737  001000  001534          MOV     #BIT9,$TMDAT            ;LOAD DISPLAY STATUS

                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
004740  012737  001200  001124          MOV     #BIT9!BIT7,$GDDAT       ;LOAD EXPECTED

                        ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
004756  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004764  001401                          BEQ     TST26           ;;BR IF EQUAL
004766  104001                          ERROR   1               ;ERROR, STATUS NOT = 1200

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

```
                                        ;*TEST 26       TEST THAT STORE (BIT 10) CAN BE SET
                                        ;;*****************************************************************
004770  000004                          TST26:  SCOPE
004772  012737  002000  001534                  MOV     #BIT10,$TMDAT           ;LOAD DISPLAY STATUS

                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
005010  012737  002200  001124                  MOV     #BIT10!BIT7,$GDDAT      ;LOAD EXPECTED

                                        ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
005026  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;COMPARE
005034  001401                                  BEQ     TST27                   ;;BR IF EQUAL
005036  104001                                  ERROR   1                       ;ERROR, STATUS NOT = 2200

                                        ;;*****************************************************************
                                        ;*TEST 27       TEST THAT WRITE THRu (BIT 11) CAN BE SET
                                        ;;*****************************************************************
005040  000004                          TST27:  SCOPE
005042  012737  004000  001534                  MOV     #BIT11,$TMDAT           ;LOAD DISPLAY STATUS

                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
005060  012737  004200  001124                  MOV     #BIT11!BIT7,$GDDAT      ;LOAD EXPECTED

                                        ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
005076  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;COMPARE
005104  001401                                  BEQ     TST30                   ;;BR IF EQUAL
005106  104001                                  ERROR   1                       ;ERROR, STATUS NOT = 4200

                                        ;;*****************************************************************
                                        ;*TEST 30       TEST THAT  INTENSIFY BIT  SETS THAT THE READY BIT
                                        ;;*****************************************************************
005110  000004                          TST30:  SCOPE
                                        ; AND THEN SETS AFTER A DELAY
005112  012700  001000                          MOV     #1000,R0
005116  005037  001124                          CLR     $GDDAT                  ;LOAD EXPECTED
005122  012737  000001  001534                  MOV     #BIT0,$TMDAT            ;INTENSIFY

                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
005140  012737  000200  001124                  MOV     #BIT7,$GDDAT            ;LOAD EXPECTED

                                        ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
005156  023737  001124  001126                  CMP     $GDDAT,$BDDAT
005164  001401                                  BEQ     TST31                   ;;BR IF EQUAL
005166  104011                                  ERROR   11                      ;READY FAILED TO sET AFTER A DELAY
                                                                        ;IS STORAGE SCOPE CONNECTED BUT NOT TURNED ON ??
                                        ;;*****************************************************************
                                        ;*TEST 31       TESt THAT MODE 1 (INTENSIFY ON DACO) SETS THE READY FLAG
                                        ;;*****************************************************************
005170  000004                          TST31:  SCOPE
                                        ; AND THEN SETS IT
005172  012700  001000                          MOV     #1000,R0                ;SET UP DELAY
005176  012737  000204  001124                  MOV     #BIT7!BIT2,$GDDAT       ;LOAD EXPECTED
005204  012737  000004  001534                  MOV     #BIT2,$TMDAT            ;LOAD MODE 1

                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
```

```
1076
1077                                        ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
1078  005232  105737  001126                        TSTB    $BDDAT          ;TEST READY
1079  005236  100402                            BMI     2$              ;;BR IF READY STILL SET
1080  005240  104011                            ERROR   11                      ;ERROR, IN MODE 1 READY SHOULD NOT
1081                                                                       ; CLEAR UNTIL DAC0 IS LOADED
1082  005242  000425                            BR      TST32           ;;BR TO SCOPE
1083
1084  005244  012737  000004  001124  2$:       MOV     #BIT2,$GDDAT            ;LOAD EXPECTED
1085  005252  012737  000000  001534            MOV     #0,$TMDAT               ;ADDRESS DAC 0
1086
1087                                        ;*      MOV     $TMDAT,@DAC0    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC0
1088
1089                                        ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
1090
1091  005300  105737  001126                        TSTB    $BDDAT
1092  005304  100404                            BMI     TST32           ;;NEXT TEST
1093  005306  012737  000204  001124            MOV     #BIT7!BIT2,$GDDAT       ;LOAD EXPECTED
1094  005314  104011                            ERROR   11                      ;ERROR, READY FAILED TO sET
1095                                                                       ; AFTER MODE 1 OPERATION
1096
1097                                        ;;**************************************************************
1098                                        ;*TEST 32        TEST THAT MODE 2 (INTENSIFY ON DAC1) SETS THE READY FLAG
1099                                        ;;**************************************************************
1100  005316  000004                        TST32:  SCOPE
1101                                        ; AND THEN SETS IT
1102  005320  012700  001000                        MOV     #1000,R0                ;SET UP DELAY
1103  005324  012737  000210  001124            MOV     #BIT7!BIT3,$GDDAT       ;LOAD EXPECTED
1104  005332  012737  000010  001534            MOV     #BIT3,$TMDAT            ;LOAD MODE 2
1105
1106                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1107
1108                                        ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
1109  005360  105737  001126                        TSTB    $BDDAT          ;TEST READY
1110  005364  100402                            BMI     2$              ;;BR IF SET
1111  005366  104011                            ERROR   11                      ;ERROR, IN MODE 2 READY SHOULD NOT CLEAR
1112                                                                       ;UNTIL DAC1 IS LOADED
1113  005370  000425                            BR      TST33           ;;BR TO sCOPE
1114  005372  012737  000010  001124  2$:       MOV     #BIT3,$GDDAT            ;LOAD EXPECTED
1115  005400  012737  000000  001534            MOV     #0,$TMDAT               ;ADDRESS DAC1
1116
1117                                        ;*      MOV     $TMDAT,@DAC1    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC1
1118
1119                                        ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
1120  005426  105737  001126                        TSTB    $BDDAT
1121  005432  100404                            BMI     TST33           ;;NEXT TESt
1122  005434  012737  000210  001124            MOV     #BIT7!BIT3,$GDDAT       ;LOAD EXPECTED
1123  005442  104011                            ERROR   11                      ;ERROR, READY FAILED TO SET
1124                                                                       ; AFTER MODE 2 OPERATION
1125                                        ;;**************************************************************
1126                                        ;*TEST 33        TESt WHEN ERASE IS SET, READY BIT CLEARS AND SET AFTER DELAY    (SW12=1)
1127                                        ;;**************************************************************
1128  005444  000004                        TST33:  SCOPE
1129  005446  012737  000001  001166            MOV     #1,$TIMES       ;;DO 1 ITERATION
```

```
1130  005454  032777  010000  173456          BIT     #BIT12,@SWR             ;TEST BIT 12
1131  005462  001437                          BEQ     TST34           ;;BYPASS IF NO STORAGE SCOPE
1132  005464  012700  000010                  MOV     #10,R0
1133  005470  005037  021154                  CLR     TEMP                    ;CLEAR DELAY
1134  005474  012737  002000  001534          MOV     #BIT10,$TMDAT           ;SET STORE MODE
1135
1136                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1137  005512  052737  010000  001534          BIS     #BIT12,$TMDAT           ;SET ERASE BIT
1138
1139                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1140  005530                          1$:
1141
1142                                  ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
1143  005540  105737  001534                  TSTB    $TMDAT
1144  005544  100406                          BMI     TST34           ;;BR IF SET
1145  005546  005337  021154                  DEC     TEMP            ;DELAY
1146  005552  001366                          BNE     1$              ;;BR IF NOT READY
1147  005554  005300                          DEC     R0              ;DECREMENT COUNTER
1148  005556  001364                          BNE     1$              ;;BR IF NOT DONE
1149  005560  104011                          ERROR   11              ;ERROR, ERASE CLEARED READY AND FAILED
1150                                                                  ;TO SET READY AFTER A DELAY
1151                                                  ;SWR 12 = 1 AND NO STORAGE SCOPE CONNECTED ??
1152                                  ;;**************************************************************
1153                                  ;*TEST 34        TEST THAT RESET CLEARS MODE, EXT. DELAY AND FAST INTENISIFY BITS
1154                                  ;;**************************************************************
1155  005562  000004                  TST34:  SCOPE
1156  005564  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1157  005572  012737  000036  001534          MOV     #BIT4!BIT3!BIT2!BIT1,$TMDAT
1158
1159                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1160  005610  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
1161  005616  004737  026474                  JSR     PC,$RESET
1162
1163                                  ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
1164  005632  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1165  005640  001401                          BEQ     TST35           ;;BR IF EQUAL
1166  005642  104001                          ERROR   1               ;ERROR, RESET FAILED TO CLEAR STATUS REG
1167                                  ;;**************************************************************
1168                                  ;*TEST 35        TEST THAT RESET CLEARS INTERRUPT ENABLE, CHANNEL, STORE, WRITE THRU
1169                                  ;;**************************************************************
1170  005644  000004                  TST35:  SCOPE
1171  005646  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1172  005654  012737  007100  001534          MOV     #BIT11!BIT10!BIT9!BIT6,$TMDAT
1173
1174                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1175  005672  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
1176  005700  004737  026474                  JSR     PC,$RESET
1177
1178                                  ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
1179  005714  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1180  005722  001401                          BEQ     TST36           ;;BR IF EQUAL
1181  005724  104001                          ERROR   1               ;ERROR, RESET FAILED TO CLEAR STATUS
1182                                  ;;**************************************************************
1183                                  ;*TEST 36        TEST THAT RESET CLEARS DACO REGISTER    (SW09=0)
```

```
1184                                    ;;*****************************************************************
1185  005726  000004                    TST36:  SCOPE
1186  005730  012737  000040  001166            MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1187  005736  032777  001000  173174            BIT     #SW09,@SWR      ;TEST IF BIT 9 IS SET
1188  005744  001024                            BNE     TST37           ;;BR IF SET
1189  005746  012737  177777  001534            MOV     #-1,$TMDAT
1190
1191                                    ;*          MOV     $TMDAT,@DACO    ;/ PUT DATA FROM $TMdAT TO DEVICE REG DACO
1192  005764  005037  001124                    CLR     $GDDAT                  ;LOAD EXPECTED
1193  005770  004737  026474                    JSR     PC,$RESET
1194
1195                                    ;*          MOV     @DACO,$BDDAT    ;/READ DEVICE REG DACO,PUT DATA IN $BDDAT.
1196  006004  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1197  006012  001401                            BEQ     TST37           ;;BR IF EQUAL
1198  006014  104002                            ERROR   2                       ;ERROR, RESET FAILED TO CLEAR  DAC 0 REGISTER
1199                                    ;;*****************************************************************
1200                                    ;*TEST 37        TEST THAT RESET CLEARS DAC1 REGISTER    (SW09=0)
1201                                    ;;*****************************************************************
1202  006016  000004                    TST37:  SCOPE
1203  006020  012737  000040  001166            MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1204  006026  032777  001000  173104            BIT     #SW09,@SWR      ;TEST IF BIT 9 IS SET
1205  006034  001024                            BNE     TST40           ;;BR IF SET
1206  006036  012737  177777  001534            MOV     #-1,$TMDAT
1207
1208                                    ;*          MOV     $TMDAT,@DAC1    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC1
1209  006054  005037  001124                    CLR     $GDDAT                  ;LOAD EXPECTED
1210  006060  004737  026474                    JSR     PC,$RESET
1211
1212                                    ;*          MOV     @DAC1,$BDDAT    ;/READ DEVICE REG DAC1,PUT DATA IN $BDDAT.
1213  006074  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1214  006102  001401                            BEQ     TST40           ;;BR IF EQUAL
1215  006104  104003                            ERROR   3                       ;ERROR, RESET FAILED TO CLEAR DAC1 REGISTER
1216                                    ;;*****************************************************************
1217                                    ;*TEST 40        TEST THAT RESET CLEARS DAC #2 REGISTER  (SW09=0)
1218                                    ;;*****************************************************************
1219  006106  000004                    TST40:  SCOPE
1220  006110  012737  000040  001166            MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1221  006116  032777  001000  173014            BIT     #SW09,@SWR      ;TEST IF BIT 9 IS SET
1222  006124  001023                            BNE     TST41           ;;BR IF SET
1223  006126  012737  177777  001534            MOV     #-1,$TMDAT              ;LOAD THE REGISTER
1224
1225                                    ;*          MOV     $TMDAT,@DAC2    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC2
1226  006144  005037  001124                    CLR     $GDDAT                  ;CLEAR EXPECTED
1227  006150  004737  026474                    JSR     PC,$RESET
1228
1229                                    ;*          MOV     @DAC2,$BDDAT    ;/READ DEVICE REG DAC2,PUT DATA IN $BDDAT.
1230  006164  005737  001126                    TST     $BDDAT
1231  006170  001401                            BEQ     TST41           ;;BR IF CLEARED
1232  006172  104004                            ERROR   4                       ;ERROR, RESET FAILED TO CLEAR DAC #2
1233                                    ;;*****************************************************************
1234                                    ;*TEST 41        TEST THAT RESET CLEARS DAC #3 REGISTER  (SW09=0)
1235                                    ;;*****************************************************************
1236  006174  000004                    TST41:  SCOPE
1237  006176  012737  000040  001166            MOV     #40,$TIMES      ;;DO 40 ITERATIONS
```

```
1238  006204  032777  001000  172726          BIT     #SW09,@SWR      ;TEST IF BIT 9 IS SET
1239  006212  001023                          BNE     TST42           ;;BR IF SET
1240  006214  012737  177777  001534          MOV     #-1,$TMDAT              ;LOAD THE REGISTER
1241
1242                                  ;*      MOV     $TMDAT,@DAC3    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC3
1243  006232  005037  001124                  CLR     $GDDAT                  ;CLEAR THE EXPECTED
1244  006236  004737  026474                  JSR     PC,$RESET
1245
1246                                  ;*      MOV     @DAC3,$BDDAT    ;/READ DEVICE REG DAC3,PUT DATA IN $BDDAT.
1247  006252  005737  001126                  TST     $BDDAT
1248  006256  001401                          BEQ     TST42           ;;BR IF CLEARED
1249  006260  104005                          ERROR   5               ;ERROR, RESET FAILED TO CLEAR DAC #3
1250                                  ;;**************************************************************
1251                                  ;*TEST 42        TEST THAT RESET SET DAC0 TO 4000        (SW09=1)
1252                                  ;;**************************************************************
1253  006262  000004                  TST42:  SCOPE
1254  006264  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1255  006272  032777  001000  172640          BIT     #SW09,@SWR              ;TEST BIT 9
1256  006300  001425                          BEQ     TST43           ;;BR IF CLEARED
1257  006302  012737  003777  001534          MOV     #3777,$TMDAT            ;LOAD DAC0
1258
1259                                  ;*      MOV     $TMDAT,@DAC0    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC0
1260  006320  012737  004000  001124          MOV     #4000,$GDDAT            ;LOAD EXPTECTED
1261  006326  004737  026474                  JSR     PC,$RESET
1262
1263                                  ;*      MOV     @DAC0,$BDDAT    ;/READ DEVICE REG DAC0,PUT DATA IN $BDDAT.
1264  006342  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1265  006350  001401                          BEQ     TST43           ;;BR IF SAME
1266  006352  104002                          ERROR   2               ;RESET FAILED TO SET DAC0 TO 4000
1267                                  ;;**************************************************************
1268                                  ;*TEST 43        TEST THAT RESET SET DAC1 TO 4000        (SW09=1)
1269                                  ;;**************************************************************
1270  006354  000004                  TST43:  SCOPE
1271  006356  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1272  006364  032777  001000  172546          BIT     #SW09,@SWR              ;TEST BIT 9
1273  006372  001425                          BEQ     TST44           ;;BR IF CLEARED
1274  006374  012737  003777  001534          MOV     #3777,$TMDAT            ;LOAD DAC1
1275
1276                                  ;*      MOV     $TMDAT,@DAC1    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC1
1277  006412  012737  004000  001124          MOV     #4000,$GDDAT            ;LOAD EXPTECTED
1278  006420  004737  026474                  JSR     PC,$RESET
1279
1280                                  ;*      MOV     @DAC1,$BDDAT    ;/READ DEVICE REG DAC1,PUT DATA IN $BDDAT.
1281  006434  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1282  006442  001401                          BEQ     TST44           ;;BR IF SAME
1283  006444  104003                          ERROR   3               ;RESET FAILED TO sET DAC1 TO 4000
1284                                  ;;**************************************************************
1285                                  ;*TEST 44        TEST THAT RESET SET DAC2 TO 4000        (SW09=1)
1286                                  ;;**************************************************************
1287  006446  000004                  TST44:  SCOPE
1288  006450  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1289  006456  032777  001000  172454          BIT     #SW09,@SWR              ;TEST BIT 9
1290  006464  001425                          BEQ     TST45           ;;BR IF CLEARED
1291  006466  012737  003777  001534          MOV     #3777,$TMDAT            ;LOAD DAC2
```

```
1292
1293                                    ;*      MOV     $TMDAT,@DAC2    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC2
1294  006504  012737  004000  001124            MOV     #4000,$GDDAT            ;LOAD EXPTECTED
1295  006512  004737  026474                    JSR     PC,$RESET
1296
1297                                    ;*      MOV     @DAC2,$BDDAT    ;/READ DEVICE REG DAC2,PUT DATA IN $BDDAT.
1298  006526  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1299  006534  001401                            BEQ     TST45           ;;BR IF SAME
1300  006536  104004                            ERROR   4                       ;RESET FAILED TO sET DAC2 TO 4000
1301                                    ;;**************************************************************
1302                                    ;*TEST 45       TEST THAT RESET SET DAC3 TO 4000        (SW09=1)
1303                                    ;;**************************************************************
1304  006540  000004                    TST45:  SCOPE
1305  006542  012737  000040  001166            MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1306  006550  032777  001000  172362            BIT     #SW09,@SWR              ;TEST BIT 9
1307  006556  001425                            BEQ     TST46           ;;BR IF CLEARED
1308  006560  012737  003777  001534            MOV     #3777,$TMDAT            ;LOAD DAC3
1309
1310                                    ;*      MOV     $TMDAT,@DAC3    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC3
1311  006576  012737  004000  001124            MOV     #4000,$GDDAT            ;LOAD EXPTECTED
1312  006604  004737  026474                    JSR     PC,$RESET
1313
1314                                    ;*      MOV     @DAC3,$BDDAT    ;/READ DEVICE REG DAC3,PUT DATA IN $BDDAT.
1315  006620  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1316  006626  001401                            BEQ     TST46           ;;BR IF SAME
1317  006630  104005                            ERROR   5                       ;RESET FAILED TO SET DAC3 TO 4000
1318                                    ;;**************************************************************
1319                                    ;*TEST 46       TEST THAT EXTERNAL DELAY DOES NOT SET DISPLAY READY     SW10=0
1320                                    ;;**************************************************************
1321  006632  000004                    TST46:  SCOPE
1322  006634  012737  000010  001166            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1323  006642  032777  002000  172270            BIT     #SW10,@SWR              ;TEST IF SW 10 IS SET
1324  006650  001051                            BNE     TST47           ;;BR IF EXT. DELAY SWITCH IS SET
1325  006652  012737  000020  001534            MOV     #BIT4,$TMDAT            ;LOAD EXT. DELAY ENABLE
1326
1327                                    ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1328
1329                                    ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
1330  006700  105237  001534                    INCB    $TMDAT
1331
1332                                    ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEViCE REG STAT
1333  006714  012700  000000                    MOV     #0,R0                   ;LOAD DELAY
1334  006720                            1$:
1335
1336                                    ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
1337  006730  105737  001534                    TSTB    $TMDAT
1338  006734  100403                            BMI     2$                      ;BR IF SET
1339  006736  005300                            DEC     R0                      ;DELAY
1340  006740  001367                            BNE     1$                      ;BR IF NOT FINISHED
1341  006742  000414                            BR      TST47           ;;
1342  006744  012737  000020  001124    2$:     MOV     #BIT4,$GDDAT            ;LOAD EXPECTED
1343
1344                                    ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
1345  006762  023737  001124  001126            CMP     $GDDAT,$BDDAT
```

```
1346  006770  001401                          BEQ     TST47           ;;BRANCH IF EQ
1347  006772  104011                          ERROR   11                      ;EXT. DELAY CAUSED READY TO SET
1348                                                                  ;WHEN THE OPERATOR (VIA SW10) SAID NO EXT. DELAY WAS CON
1349
1350                                  ;;*****************************************************************
1351                                  ;*TEST 47        TEST THE EXTERNAL DELAY DOES SET DISPLAY READY  SW10=1
1352                                  ;;*****************************************************************
1353  006774  000004                  TST47:  SCOPE
1354  006776  012737  000010  001166          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1355  007004  032777  002000  172126          BIT     #SW10,@SWR              ;TEST SR BIT 10
1356  007012  001442                          BEQ     TST50           ;;BR IF CLEARED
1357  007014  012737  000020  001534          MOV     #BIT4,$TMDAT            ;ENABLE EXT. DELAY
1358
1359                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1360  007032  012700  000000                  MOV     #0,R0                   ;LOAD COUNTER
1361  007036  005237  001534                  INC     $TMDAT                  ;ENABLE DISPLAY
1362
1363                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1364  007052                          1$:
1365
1366                                  ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
1367  007062  105737  001536                  TSTB    STAT
1368  007066  100412                          BMI     2$                      ;BR IF DONe
1369  007070  005300                          DEC     R0                      ;DELAY
1370  007072  001367                          BNE     1$                      ;BR IF NOT DONE
1371  007074  012737  000220  001124          MOV     #BIT7!BIT4,$GDDAT       ;LOAD EXPECTED VALUE
1372
1373                                  ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
1374  007112  104011                          ERROR   11                      ;EXT. DELAY FAILED TO SET READY
1375                                                                  ;WHEN THE OPERATOR (VIA SW 10=1) SAID AN EXT. DELAY WAS
1376  007114  004737  026474          2$:     JSR     PC,$RESET                       ;ENSURE CLEARED AA11-K
1377
1378                                  ;;*****************************************************************
1379                                  ;*TEST 50        DETERMINE IF MORE AA11-K'S REMAIN TO BE TESTED
1380                                  ;;*****************************************************************
1381  007120  000004                  TST50:  SCOPE
1382  007122  012737  000001  001166          MOV     #1,$TIMES       ;;DO 1 ITERATION
1383  007130  004737  026474                  JSR     PC,$RESET
1384  007134  005237  001206                  INC     $UNIT                   ;MORE UNITS?
1385  007140  123737  001206  021164          CMPB    $UNIT,EVER
1386  007146  001431                          BEQ     $EOP            ;;BR IF NONE
1387  007150  063737  001526  001536          ADD     VADDR,STAT              ;UPDATE ADDRESS
1388  007156  063737  001526  001540          ADD     VADDR,DAC0
1389  007164  063737  001526  001542          ADD     VADDR,DAC1
1390  007172  063737  001526  001544          ADD     VADDR,DAC2
1391  007200  063737  001526  001546          ADD     VADDR,DAC3
1392  007206  063737  001530  001550          ADD     VVECT,IV                ;UPDATE VECTOR
1393  007214  063737  001530  001552          ADD     VVECT,IVS
1394  007222  005037  001102                  CLR     $TSTNM
1395  007226  000137  002614                  JMP     LTEST                   ;TEST ANOTHER AA11-K
1396
1397                                  .SBTTL  END OF PASS ROUTINE
1398
1399                                  ;;*****************************************************************
```

```
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO MTEST

007232                                  $EOP:
007232  000004                                  SCOPE
007234  005037  001102                          CLR     $TSTNM          ;;ZERO THE TEST NUMBER
007240  005037  001166                          CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
007244  005237  001202                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
007250  042737  100000  001202                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
007256  005327                                  DEC     (PC)+           ;;LOOP?
007260  000001                        $EOPCT: .WORD   1
007262  003022                                  BGT     $DOAGN          ;;YES
007264  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
007266  000001                        $ENDCT: .WORD   1
007270  007260                                  $EOPCT
007272  104401  007337                          TYPE    ,$ENDMG         ;;TYPE "END PASS #"
007276  013746  001202                          MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
007302  104405                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
007304  104401  007334                          TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
007310  013700  000042                $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
007314  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
007316  000005                                  RESET                   ;;CLEAR THE WORLD
007320  004710                        $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
007322  000240                                  NOP                     ;;SAVE ROOM
007324  000240                                  NOP                     ;;FOR
007326  000240                                  NOP                     ;;ACT11
007330                                  $DOAGN:
007330  000137                                  JMP     @(PC)+          ;;RETURN
007332  002442                        $RTNAD: .WORD   MTEST
007334      377       377       000   $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
007337      015  042412  042116       $ENDMG: .ASCIZ  <15><12>/END PASS #/
007344  050040  051501  020123
007352  000043

007354  012706  001100                VISUAL: MOV     #STACK,SP               ;LOAD SP
007360  005737  021166                        TST     EVER1                   ;TEST IF TYPED ONCE
007364  001006                                BNE     PIC0                    ;BR IF YES
007366  104401                                TYPE
007370  016202                                MES6                    ;HEADER ABOUT SCOPE ADJ.
007372  104401                                TYPE
007374  016232                                MES15                   ;TEXT ABOUT SWR
007376  005237  021166                        INC     EVER1                   ;SET FLAG
                                            .SBTTL  DISPLAY HORIZONTAL LINE
007402  013737  001540  007470        PIC0:   MOV     DAC0,R0X
007410  013737  001542  007472                MOV     DAC1,R1Y
007416  032777  000010  171514                BIT     #SW03,@SWR              ;TEST SR BIT 3
007424  001406                                BEQ     2$                      ;BR IF DAC #0 AND #1
007426  013737  001544  007470                MOV     DAC2,R0X                ;USE DAC #2 AND 3
007434  013737  001546  007472                MOV     DAC3,R1Y
007442  012737  000005  021152        2$:     MOV     #5,TICKS                ;LOAD TIMER FOR CP TYPE
007450  004737  015064                        JSR     PC,CHTIME               ;CHANGE TIMER FOR CP TYPE
007454  004737  007570                1$:     JSR     PC,PBB
007460  004737  014756                        JSR     PC,TIMER                ;DONE ?
```

```
1454  007464  000773                          BR      1$
1455  007466  000405                          BR      PIC1
1456  007470  000000                  ROX:    .WORD   0
1457  007472  000000                  R1Y:    .WORD   0
1458  007474  000000                  R0Y:    .WORD   0
1459  007476  000000                  R1X:    .WORD   0
1460  007500  000000                  R2Y:    .WORD   0
1461                                  .SBTTL  DISPLAY A VERTICAL LINE
1462  007502  013737  001542  007470  PIC1:   MOV     DAC1,ROX
1463  007510  013737  001540  007472          MOV     DAC0,R1Y
1464  007516  032777  000010  171414          BIT     #SW03,@SWR              ;TEST SR BIT 3
1465  007524  001406                          BEQ     2$                      ;BR IF DAC #0 AND 1
1466  007526  013737  001546  007470          MOV     DAC3,ROX                ;USE DAC #2 AND 3
1467  007534  013737  001544  007472          MOV     DAC2,R1Y
1468  007542  012737  000005  021152  2$:     MOV     #5,TICKS                ;LOAD TIME
1469  007550  004737  015064                  JSR     PC,CHTIME               ;CHANGE TIMER FOR CP TYPE
1470  007554  004737  007570          1$:     JSR     PC,PBB
1471  007560  004737  014756                  JSR     PC,TIMER                ;DONE ?
1472  007564  000773                          BR      1$
1473  007566  000516                          BR      PIC3
1474  007570  012737  000000  001534  PBB:    MOV     #0,$TMDAT
1475
1476                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1477  007606  032777  002000  171324          BIT     #SW10,@SWR              ;TEST IS SW10 =1
1478  007614  001413                          BEQ     10$                     ;BR IF NOT
1479
1480                                  ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
1481  007626  052737  000020  001534          BIS     #BIT4,$TMDAT
1482
1483                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1484  007644  013704  001536          10$:    MOV     STAT,R4
1485  007650  012703  007777                  MOV     #7777,R3        ;SET HIGH LIMIT
1486  007654  012702  000001                  MOV     #1,R2           ;INITIALIZE INCREMENTS BETWEEN POINTS
1487  007660  032777  001000  171252          BIT     #SW09,@SWR              ;TEST SW 9 = 1
1488  007666  001042                          BNE     2$                      ;BR IF YES
1489  007670  012737  000400  001534          MOV     #400,$TMDAT
1490
1491                                  ;*      MOV     $TMDAT,@R1Y     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R1Y
1492  007706  012737  000000  001534          MOV     #0,$TMDAT
1493
1494                                  ;*      MOV     $TMDAT,@ROX     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ROX
1495  007724  060237  001534          1$:     ADD     R2,$TMDAT               ;INCREMENT
1496
1497                                  ;*      MOV     $TMDAT,@ROX     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ROX
1498
1499                                  ;*      MOV     @STAT,$GDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $GDDAT.
1500  007750  005237  001124                  INC     $GDDAT
1501
1502                                  ;*      MOV     $GDDAT,@STAT    ;/ PUT DATA FROM $GDDAT TO DEVICE REG STAT
1503  007764  023703  001534                  CMP     $TMDAT,R3               ;DONE ALL POINTS?
1504  007770  001355                          BNE     1$              ;NO
1505  007772  000207                          RTS     PC
1506  007774  012737  000000  001534  2$:     MOV     #0,$TMDAT                       ;LOAD TWO'S COMP. ORIGIN
1507
```

```
1508                                        ;*      MOV     $TMDAT,@R1Y     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R1Y
1509
1510                                        ;*      MOV     $TMDAT,@R0X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R0X
1511  010022  000740                               BR      1$
1512                                                .SBTTL  PINCUSHION TEST (DISPLAY SQUARE)
1513
1514  010024  012737  000000  001534  PIC3:       MOV     #0,$TMDAT
1515
1516                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1517  010042  012737  000030  021152              MOV     #30,TICKS
1518  010050  004737  015064                      JSR     PC,CHTIME
1519  010054  013737  001540  007476              MOV     DAC0,R1X
1520  010062  013737  001542  007500              MOV     DAC1,R2Y
1521  010070  032777  000010  171042              BIT     #SW03,@SWR      ;TEST SR BIT 3
1522  010076  001406                              BEQ     1$              ;BR IF DAC #0 AND 1
1523  010100  013737  001544  007476              MOV     DAC2,R1X                ;USE DAC #2 AND 3
1524  010106  013737  001546  007500              MOV     DAC3,R2Y
1525  010114  013703  001536          1$:         MOV     STAT,R3
1526  010120  012704  000020                      MOV     #20,R4
1527  010124  032777  001000  171006  P3:         BIT     #SW09,@SWR      ;TEST BIT 9
1528  010132  001414                              BEQ     1$
1529  010134  012737  004000  001534              MOV     #4000,$TMDAT    ;LOAD TWO'S COMP ORIGIN
1530
1531                                        ;*      MOV     $TMDAT,@R1X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R1X
1532
1533                                        ;*      MOV     $TMDAT,@R2Y     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R2Y
1534  010162  000412                               BR      2$
1535  010164  005037  001534          1$:         CLR     $TMDAT          ;CLEAR AXIS
1536
1537                                        ;*      MOV     $TMDAT,@R1X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R1X
1538
1539                                        ;*      MOV     $TMDAT,@R2Y     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R2Y
1540                                                ;DRAW BOTTOM LINE
1541  010210  012700  000377          2$:         MOV     #377,R0
1542  010214  032777  002000  170716              BIT     #SW10,@SWR              ;TEST IF SW10 =1
1543  010222  001413                              BEQ     P3A                     ;BR IF NOT
1544
1545                                        ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
1546  010234  052737  040000  001534              BIS     #BIT14,$TMDAT
1547
1548                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1549  010252                                P3A:
1550
1551                                        ;*      MOV     @R1X,$TMDAT     ;/READ DEVICE REG R1X,PUT DATA IN $TMDAT.
1552  010262  060437  001534                      ADD     R4,$TMDAT
1553
1554                                        ;*      MOV     $TMDAT,@R1X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R1X
1555
1556                                        ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
1557  010306  005237  001534                      INC     $TMDAT
1558
1559                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1560                                                                        ;WAIT FOR READY
1561  010322  005300                               DEC     R0
```

```
1562  010324  001352                        BNE     P3A             ;NO
1563                              ;DRAW RIGHT LINE
1564  010326  012700  000377                MOV     #377,R0
1565  010332                    P3B:
1566
1567                              ;*      MOV     @R2Y,$TMDAT     ;/READ DEVICE REG R2Y,PUT DATA IN $TMDAT.
1568  010342  060437  001534                ADD     R4,$TMDAT
1569
1570                              ;*      MOV     $TMDAT,@R2Y     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R2Y
1571
1572                              ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
1573  010366  005237  001534                INC     $TMDAT
1574
1575                              ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1576                                                              ;WAIT FOR READY
1577  010402  005300                        DEC     R0
1578  010404  001352                        BNE     P3B             ;NO
1579                              ;DRAW TOP LINE
1580  010406  012700  000377                MOV     #377,R0
1581  010412                    P3C:
1582
1583                              ;*      MOV     @R1X,$TMDAT     ;/READ DEVICE REG R1X,PUT DATA IN $TMDAT.
1584  010422  160437  001534                SUB     R4,$TMDAT
1585
1586                              ;*      MOV     $TMDAT,@R1X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R1X
1587
1588                              ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
1589  010446  005237  001534                INC     $TMDAT
1590
1591                              ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1592                                                              ;WAIT FOR READY
1593  010462  005300                        DEC     R0
1594  010464  001352                        BNE     P3C             ;NO
1595                              ;DRAW LEFT LINE
1596  010466  012700  000377                MOV     #377,R0
1597  010472                    P3D:
1598
1599                              ;*      MOV     @R2Y,$TMDAT     ;/READ DEVICE REG R2Y,PUT DATA IN $TMDAT.
1600  010502  160437  001534                SUB     R4,$TMDAT
1601
1602                              ;*      MOV     $TMDAT,@R2Y     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R2Y
1603
1604                              ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
1605  010526  005237  001534                INC     $TMDAT
1606
1607                              ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMdAT TO DEVICE REG STAT
1608
1609                                                              ;WAIT FOR READY
1610  010542  005300                        DEC     R0
1611  010544  001352                        BNE     P3D             ;NO
1612  010546  004737  014756                JSR     PC,TIMER
1613  010552  000401                        BR      1$
1614  010554  000402                        BR      PIC4
1615  010556  000137  010124        1$:     JMP     P3
```

```
                                                .SBTTL  PLOT AN X
                                        PIC4:   MOV     #0,$TMDAT
                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
                                                MOV     #30,TICKS
                                                JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
                                        PIC4B:  MOV     DAC0,R1X
                                                MOV     DAC1,R2Y
                                                BIT     #SW03,@SWR      ;TEST SWR BIT 3
                                                BEQ     1$              ;BR IF DAC #0 AND 1
                                                MOV     DAC2,R1X                ;USE DAC #2 AND 3
                                                MOV     DAC3,R2Y
                                        1$:     MOV     STAT,R3
                                                MOV     #4,R4
                                        P4:     BIT     #SW09,@SWR      ;TEST BIT 9
                                                BEQ     1$
                                                MOV     #4000,$TMDAT
                                        ;*      MOV     $TMDAT,@R2Y     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R2Y
                                        ;*      MOV     $TMDAT,@R1X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R1X
                                                BR      2$
                                        1$:     CLR     $TMDAT
                                        ;*      MOV     $TMDAT,@R1X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R1X
                                        ;*      MOV     $TMDAT,@R2Y     ;/ PUT DATA FROM $TMdAT TO DEVICE REG R2Y
                                        ;PLOT LINE BEGINNING IN LOWER LEFT CORNER
                                        2$:     MOV     #1777,R0
                                                BIT     #SW10,@SWR              ;TEST IF SW10 =1
                                                BEQ     P4A                     ;BR IF NOT
                                        ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMdAT.
                                                BIS     #BIT4,$TMDAT
                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
                                        P4A:
                                        ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
                                                INC     $TMDAT
                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
                                        ;*      MOV     @R2Y,$TMDAT     ;/READ DEVICE REG R2Y,PUT DATA IN $TMDAT.
                                                ADD     R4,$TMDAT
                                        ;*      MOV     $TMDaT,@R2Y     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R2Y
                                        ;*      MOV     $TMDAT,@R1X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R1X
                                                                        ;+4 TO X
                                                DEC     R0
                                                BNE     P4A             ;NO
```

```
                                        ;PLOT LINE BEGINNING IN UPPER LEFT CORNER
011074  012737  007774  001534            MOV     #7774,$TMDAT

                                ;*      MOV     $TMDAT,@R2Y     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R2Y
011112  005037  001534                  CLR     $TMDAT

                                ;*      MOV     $TMDAT,@R1X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R1X
011126  012700  001777                  MOV     #1777,R0
011132                          P4B:

                                ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
011142  005237  001534                  INC     $TMDAT

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMdAT TO DEVICE REG STAT

                                ;*      MOV     @R2Y,$TMDAT     ;/READ DEVICE REG R2Y,PUT DATA IN $TMDAT.
011166  160437  001534                  SUB     R4,$TMDAT

                                ;*      MOV     $TMDAT,@R2Y     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R2Y

                                ;*      MOV     @R1X,$TMDAT     ;/READ DEVICE REG R1X,PUT DATA IN $TMDAT.
011212  060437  001534                  ADD     R4,$TMDAT

                                ;*      MOV     $TMDAT,@R1X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R1X
011226  005300                          DEC     R0
011230  001340                          BNE     P4B             ;NO
011232  004737  014756                  JSR     PC,TIMER
011236  000402                          BR      1$
011240  000137  007402                  JMP     PICO
011244  000137  010662          1$:     JMP     P4
```

```
1701
1702  011250  000000                  VCXTMP: 0
1703  011252  000000                  VCYTMP: 0
1704  011254  000000                  YPOS:   0                ;CONTAINS Y POSITION AT ANY GIVEN TIME
1705  011256  000000                  XPOS:   0                ;CONTAINS X POSITION AT ANY GIVEN TIME
1706  011260  000000                  CHRCOL: 0
1707  011262  000000                  YPT:    0
1708
1709
1710                                  .SBTTL  MANUAL DISPLAY ROUTINE
1711
1712  011264                          PIC6:
1713  011264  012706  001100          FULRMP: MOV     #STACK,SP               ;LOAD POINTER
1714  011270  004737  002072                  JSR     PC,LDTRAP               ;LOAD BUS ADDRESS
1715  011274  032777  002000  167636          BIT     #SW10,@SWR      ;TEST SR 10=1
1716  011302  001413                          BEQ     1$              ;BR IF NOT
1717
1718                                  ;*      MOV     @STAT,$TMDAT    ;/READ DEViCE REG STAT,PUT DATA IN $TMDAT.
1719  011314  052737  000020  001534          BIS     #BIT4,$TMDAT
1720
1721                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1722  011332  013737  001540  007470  1$:     MOV     DAC0,R0X                ;GET BUS ADDRESS
1723  011340  004737  011404                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #1
1724  011344  013737  001542  007470          MOV     DAC1,R0X                ;GET BUS ADDRESS
1725  011352  004737  011404                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #1
1726  011356  013737  001544  007470          MOV     DAC2,R0X                ;GET BUS ADDRESS
1727  011364  004737  011404                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #2
1728  011370  013737  001546  007470          MOV     DAC3,R0X                ;GET THE BUS ADDRESS
1729  011376  004737  011404                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #3
1730  011402  000753                          BR      1$                      ;BR BACK
1731
1732  011404  005037  001534          10$:    CLR     $TMDAT
1733
1734                                  ;*      MOV     $TMDAT,@R0X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R0X
1735  011420  062737  000010  001534  11$:    ADD     #10,$TMDAT
1736
1737                                  ;*      MOV     $TMDAT,@R0X     ;/ PUT DATA FROM $TMDAT TO DEVICE REG R0X
1738  011436  005737  001534                  TST     $TMDAT                  ;TEST IF DONe
1739  011442  001366                          BNE     11$                     ;BR IF NOT
1740  011444  004737  013672                  JSR     PC,ACTRLC               ;TEST FOR CTRL C
1741  011450  000207                          RTS     PC                      ;EXIT
1742  011452  000240                          NOP
1743  011454  000240                          NOP
1744
1745                                  .SBTTL  AUTO CALIBRATION
1746  011456  012706  001100          AUTCAL: MOV     #STACK,SP               ;LOAD STACK POINTER
1747  011462  004737  002072                  JSR     PC,LDTRAP               ;LOAD TRAN AND ADDRESSES
1748  011466  104401                          TYPE
1749  011470  020076                          AUTOCL                          ;TELL OPERATOR
1750  011472  104401                          TYPE
1751  011474  017604                          SELD01
1752
1753                                  ;LOAD BUS TRAP FOR OPERTOR FALSE SELECTION
1754
```

```
1755  011476  012737  013460  000004          MOV     #AUTRAP,ERRVEC
1756  011504  012737  000050  001102          MOV     #$TN-1,$TSTNM           ;LOAD TEST NUMBER
1757  011512  012737  011536  001110          MOV     #PSUPLY,$LPERR          ;LOAD LOOP RETURN
1758  011520  012737  011536  001106          MOV     #PSUPLY,$LPADR          ;LOAD LOOP ADDRESS
1759                                           ;@@@@@@@@@@@@@@@@@@@@@@@@@@@@
1760  011526  004537  014704                  JSR     R5,CROSS                ;CHANGE RELAYS
1761  011532  020000                          BIT13                           ;USING BIT 13 FOR CRISS-CROSS WRAP MODE
1762                                           ;@@@@@@@@@@@@@@@@@@@@@@@@@@@@
1763                                  ;;*****************************************************************
1764                                  ;*TEST 51        TEST THAT CH 3 IS AT +2.5 BIPOLAR
1765                                  ;;*****************************************************************
1766  011534  000004                  TST51:  SCOPE
1767  011536  013737  015400  001124  PSUPLY: MOV     V1754,$GDDAT            ;LOAD EXPECTED VALUE
1768  011544  004537  015204                  JSR     R5,CONVRT               ;CONVERT
1769  011550  000003                          CH3                             ;CH 3
1770  011552  013737  015412  001126          MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1771  011560  013737  015402  015420          MOV     V24,SPREAD              ;LOAD + OR - COUNT SPREAD
1772  011566  004737  015422                  JSR     PC,COMPAR               ;COMPARE $GDDAT AND $BDDAT
1773  011572  000402                          BR      NSUPLY          ;;BR IF WITHIN TOLERANCE
1774  011574  104012                          ERROR   12                      ;CH 3 FAILED TO EQUAL EXPECTED VALUE
1775
1776                                  ;;*****************************************************************
1777                                  ;*TEST 52        TEST THAT CH 4 IS AT -2.5 BIPOLAR
1778                                  ;;*****************************************************************
1779  011576  000004                  TST52:  SCOPE
1780  011600  013737  015402  001124  NSUPLY: MOV     V24,$GDDAT              ;LOAD EXPECTED VALUE
1781  011606  004537  015204                  JSR     R5,CONVRT               ;CONVERT
1782  011612  000004                          CH4                             ;CH 4
1783  011614  013737  015412  001126          MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1784  011622  013737  015402  015420          MOV     V24,SPREAD              ;LOAD + OR - COUNT SPREAD
1785  011630  004737  015422                  JSR     PC,COMPAR               ;COMPARE $GDDAT AND $BDDAT
1786  011634  000402                          BR      SUPOK           ;;BR IF WITHIN TOLERANCE
1787  011636  104012                          ERROR   12                      ;CH 4 FAILED TO EQUAL EXPECTED VALUE
1788  011640  000422                          BR      TST53           ;;
1789  011642                          SUPOK:
1790  011642  004537  014704                  JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
1791  011646  020000                          BIT13
1792  011650  104401  011656                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1793  011654  000414                          BR      64$             ;;GET OVER THE ASCIZ
1794                                  ;;65$:   .ASCIZ  <15><12>/ANALOG SUPPLIES OK/<15><12>
1795  011706                          64$:
1796                                  ;;*****************************************************************
1797                                  ;*TEST 53        TEST THAT ERASE L (CH 53) IS GREATER THAN +3 VOLTS WHEN HIGH
1798                                  ;;*****************************************************************
1799  011706  000004                  TST53:  SCOPE
1800  011710  012737  000000  001534          MOV     #0,$TMDAT               ;LOAD STATUS
1801
1802                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1803  011726  012737  001146  001124          MOV     #1146,$GDDAT            ;LOAD EXPECTED VALUE
1804  011734  004537  015204                  JSR     R5,CONVRT               ;CONVERT
1805  011740  000053                          CH53                            ;CH 53
1806  011742  013737  015412  001126          MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1807  011750  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1808  011756  003401                          BLE     TST54           ;;BR IF LESS THAN OR EQUAL
```

```
                                        ERROR   14                      ;CH 53 FAILED TO EQUAL EXPECTED VALUE

                                ;;*********************************************************************
                                ;*TEST 54        TEST THAT ERASE L (CH 53) IS LESS THAN +400 MVOLTS WHEN LOW
                                ;;*********************************************************************
                                TST54:  SCOPE

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
                                        MOV     #BIT12!BIT10,$TMDAT     ;SET ERASE

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
                                        MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
                                        JSR     R5,CONVRT               ;CONVERT
                                        CH53                            ;CH 53
                                        MOV     ADEND,$BDDAT            ;LOAD VALUE READ
                                        CMP     $GDDAT,$BDDAT           ;COMPARE
                                        BGE     TST55           ;;BR IF GREATER THAN OR EQUAL
                                        ERROR   15                      ;CH 53 FAILED TO EQUAL EXPECTED VALUE

                                ;;*********************************************************************
                                ;*TEST 55        TEST THAT WRITE-THRU (CH 53) IS LESS THAN +400 MVOLTS WHEN LOW
                                ;;*********************************************************************
                                TST55:  SCOPE
                                        MOV     #BIT11!BIT10,$TMDAT     ;SET WRITE-THRU

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
                                        MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
                                        JSR     R5,CONVRT               ;CONVERT
                                        CH53                            ;CH 54
                                        MOV     ADEND,$BDDAT            ;LOAD VALUE READ
                                        CMP     $GDDAT,$BDDAT           ;COMPARE
                                        BGE     TST56           ;;BR IF GREATER THAN OR EQUAL
                                        ERROR   15                      ;CH 54 FAILED TO EQUAL EXPECTED VALUE
                                ;;*********************************************************************
                                ;*TEST 56        TEST THAT NON-STORE (CH 55) IS LESS THAN +400 MVOLTS WHEN LOW
                                ;;*********************************************************************
                                TST56:  SCOPE
                                        MOV     30,$TMDAT               ;CLEAR STORE MODE

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
                                        MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
                                        JSR     R5,CONVRT               ;CONVERT
                                        CH55                            ;CH 55
                                        MOV     ADEND,$BDDAT            ;LOAD VALUE READ
                                        CMP     $GDDAT,$BDDAT           ;COMPARE
                                        BGE     TST57           ;;BR IF GREATER THAN OR EQUAL
                                        ERROR   15                      ;CH 53 FAILED TO EQUAL EXPECTED VALUE
                                ;;*********************************************************************
                                ;*TEST 57        TEST THAT NON-STORE L (CH 55) IS GREATER THAN +3 VOLTS WHEN HIGH
                                ;;*********************************************************************
                                TST57:  SCOPE
                                        MOV     #BIT12!BIT11!BIT10!BIT9,$TMDAT          ;SET STORE MODE

                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEViCE REG STAT
```

```
1863  012216  012737  001146  001124          MOV     #1146,$GDDAT            ;LOAD EXPECTED VALUE
1864  012224  004537  015204                  JSR     R5,CONVRT               ;CONVERT
1865  012230  000055                          CH55                            ;CH 55
1866  012232  013737  015412  001126          MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1867  012240  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1868  012246  003401                          BLE     TST60           ;;BR IF LESS THAN OR EQUAL
1869  012250  104014                          ERROR   14                      ;CH 55 FAILED TO EQUAL EXPECTED VALUE
1870                                  ;;*****************************************************************
1871                                  ;*TEST 60        TEST THAT DELAY RETURN SETS READY
1872                                  ;;*****************************************************************
1873  012252  000004                  TST60:  SCOPE
1874  012254  012737  000020  001534          MOV     #BIT4,$TMDAT            ;LOAD EXT DELAY BIT
1875
1876                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1877  012272  012737  000020  001124          MOV     #BIT4,$GDDAT            ;LOAD EXPECTED
1878  012300  005237  001534                  INC     $TMDAT                  ;MAKE READY GO AWAY
1879
1880                                  ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
1881  012314  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1882  012322  001401                          BEQ     1$              ;;BR IF EXPECTED
1883  012324  104001                          ERROR   1                       ;READY FAILED TO cLEAR
1884  012326                          1$:
1885
1886                                  ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
1887  012336  052737  002000  001534          BIS     #BIT10,$TMDAT
1888
1889                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1890  012354  012737  002220  001124          MOV     #BIT10!BIT7!BIT4,$GDDAT ;LOAD EXPECTED
1891
1892                                  ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
1893  012372  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1894  012400  001401                          BEQ     2$              ;;BR IF SAME
1895  012402  104001                          ERROR   1                       ;EXT. DELAY RETURN FAILED TO SET READY
1896  012404  012737  000000  001534  2$:     MOV     #0,$TMDAT
1897
1898                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1899                                  ;;*****************************************************************
1900                                  ;*TEST 61        TEST THAT CHANNEL 02 L (CH 56) IS GREATER THAN +3 VOLTS WHEN HIGH
1901                                  ;;*****************************************************************
1902  012422  000004                  TST61:  SCOPE
1903  012424  012737  014000  001534          MOV     #BIT12!BIT11,$TMDAT     ;LOAD STATUS
1904
1905                                  ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
1906  012442  012737  001100  001124          MOV     #1100,$GDDAT            ;LOAD EXPECTED VALUE
1907  012450  004537  015204                  JSR     R5,CONVRT               ;CONVERT
1908  012454  000056                          CH56                            ;CH 56
1909  012456  013737  015412  001126          MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1910  012464  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1911  012472  003401                          BLE     TST62           ;;BR IF LESS THAN OR EQUAL
1912  012474  104014                          ERROR   14                      ;CH 56 FAILED TO EQUAL EXPECTED
1913                                                                          ; VALUE
1914
1915                                  ;;*****************************************************************
1916                                  ;*TEST 62        TEST THAT CHANNEL 02 L (CH 56) IS LESS THAN +400 MVOLTS WHEN LOW
```

```
                                ;;*****************************************************************
012476  000004                  TST62:  SCOPE
012500  012737  003000  001534          MOV     #BIT9!BIT10,$TMDAT      ;SET CHANNEL 2
                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
012516  012737  000120  001124          MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
012524  004537  015204                  JSR     R5,CONVRT               ;CONVERT
012530  000056                          CH56                            ;CH 56
012532  013737  015412  001126          MOV     ADEND,$BDDAT            ;LOAD VALUE READ
012540  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
012546  002001                          BGE     1$              ;;BR IF GREATER THAN
012550  104015                          ERROR   15                      ;CH 56 FAILED TO EQUAL EXPECTED
                                                                        ; VALUE
012552  012737  000000  001534  1$:     MOV     #0,$TMDAT               ;CLEAR BIT 9
                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
                                ;;*****************************************************************
                                ;*TEST 63        TEST THAT "ERASE" AND DONE CAN BE CLEARED AND SET
                                ;;*****************************************************************
012570  000004                  TST63:  SCOPE
012572  012737  000000  001534          MOV     #0,$TMDAT               ;CLEAR
                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
012610  005037  001124                  CLR     $GDDAT                  ;CLEAR EXPECTED
012614  012737  010000  001534          MOV     #BIT12,$TMDAT           ;LOAD EXPECTED
                                ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
                                ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
012642  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
012650  001401                          BEQ     1$              ;;BR IF SET
012652  104001                          ERROR   1                       ;ERASE FAILED TO CLEAR READY, CHECK G5036-
                                                                        ;BCO8R CONNECTION: A-VV, VV-A
012654                          1$:
                                ;*      MOV     @GSTAT,$TMDAT   ;/READ DEVICE REG GSTAT,PUT DATA IN $TMDAT.
012664  053737  001000  001534          BIS     BIT9,$TMDAT
                                ;*      MOV     $TMDAT,@GSTAT   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GSTAT
012702  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED
                                ;*      MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
012716  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
012724  001401                          BEQ     2$              ;;BR IF CLEARED
012726  104001                          ERROR   1                       ;ERASE BIT FAILED TO CLEAR BY ERASE
                                                                        ;RETURN (SETTING OF THE CH 2 BIT)
012730                          2$:
                                ;*      MOV     @GSTAT,$TMDAT   ;/READ DEVICE REG GSTAT,PUT DATA IN $TMDAT.
012740  042737  001000  001534          BIC     #BIT9,$TMDAT
                                ;*      MOV     $TMDAT,@GSTAT   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GSTAT
```

```
1971  012756  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
1972
1973                                  ;*        MOV     @STAT,$BDDAT    ;/READ DEVICE REG STAT,PUT DATA IN $BDDAT.
1974  012774  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1975  013002  001401                          BEQ     3$              ;;BR IF EQUAL
1976  013004  104001                          ERROR   1                       ;READY FAILED TO SET ON THE END
1977                                                                  ; OF ERASE RETURN (CLEARING CH 2)
1978
1979  013006                          3$:
1980  013006  004537  014704                  JSR     R5,CROSS                ;LOAD THE RELAYS WITH 0
1981  013012  000000                          0
1982                                  ;;*********************************************************************
1983                                  ;*TEST 64        ADJUSTMENT ROUTINES FOR DAC 0 - 1
1984                                  ;;*********************************************************************
1985  013014  000004                  TST64:  SCOPE
1986  013016  012737  000001  001166          MOV     #1,$TIMES       ;;DO 1 ITERATION
1987                                  ;OFFSET ADJUST FOR DAC 0
1988  013024  004537  014532                  JSR     R5,SNDVLT               ;LOAD THE VOLTAGE
1989  013030  020760                          N50000
1990  013032  012737  000000  001534          MOV     #0,$TMDAT               ;LOAD DAC REGISTER
1991
1992                                  ;*        MOV     $TMDAT,@DAC0    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC0
1993  013050  004537  014704                  JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
1994  013054  060000                          BIT14!BIT13
1995  013056  104401                          TYPE
1996  013060  020310                          R38                             ;TELL OPR. TO ADJUST R38
1997  013062  004737  013734                  JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1998
1999                                  ;OFFSET ADJUSTMENT FOR DAC 1
2000  013066  004537  014704                  JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
2001  013072  020000                          BIT13
2002
2003                                  ;*        MOV     $TMDAT,@DAC1    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC1
2004  013104  104401                          TYPE
2005  013106  020355                          R40                             ;TELL OPR. TO ADJUST R40
2006  013110  004737  013734                  JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
2007
2008                                  ;GAIN ADJUST FOR DAC 0
2009  013114  004537  014532                  JSR     R5,SNDVLT
2010  013120  020773                          P49976
2011  013122  012737  007777  001534          MOV     #7777,$TMDAT            ;LOAD DAC 0
2012
2013                                  ;*        MOV     $TMDAT,@DAC0    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC0
2014  013140  004537  014704                  JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
2015  013144  060000                          BIT14!BIT13
2016  013146  104401                          TYPE
2017  013150  020534                          R37                             ;TELL OPR. TO ADJUST R37
2018  013152  004737  013734                  JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
2019
2020                                  ;GAIN ADJUST FOR DAC 1
2021  013156  004537  014704                  JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
2022  013162  020000                          BIT13
2023
2024                                  ;*        MOV     $TMDAT,@DAC1    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC1
```

```
2025  013174  104401                          TYPE
2026  013176  020601                          R39                             ;TELL OPR. TO ADJUST R39
2027  013200  004737  013734                  JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
2028
2029                                  ;DAC 0 D/A FUNCTION CHECK
2030  013204  004537  015112                  JSR     R5,FUNDAC               ;GO TO DAC FUNCTION SUB.
2031  013210  001540                          DAC0
2032  013212  000051                          CH51
2033
2034                                  ;DAC 1 D/A FUNCTION CHECK
2035  013214  004537  015112                  JSR     R5,FUNDAC
2036  013220  001542                          DAC1
2037  013222  000052                          CH52
2038
2039                                  ;;*****************************************************************
2040                                  ;*TEST 65        ADJUSTMENT ROUTINES FOR DAC 2 - 3
2041                                  ;;*****************************************************************
2042  013224  000004                  TST65:  SCOPE
2043  013226  012737  000001  001166          MOV     #1,$TIMES               ;;DO 1 ITERATION
2044                                  ;NOW DO  DAC 2 AND 3
2045  013234  104401                          TYPE
2046  013236  017651                          SELD23
2047                                  ;OFFSET ADJUST FOR DAC 2
2048  013240  004537  014532                  JSR     R5,SNDVLT               ;LOAD E.D.C
2049  013244  020760                          N50000
2050  013246  012737  000000  001534          MOV     #0,$TMDAT               ;LOAD DAC VALUE
2051
2052                                  ;*      MOV     $TMDAT,@DAC2    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC2
2053  013264  004537  014704                  JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
2054  013270  060000                          BIT14!BIT13
2055
2056  013272  104401                          TYPE
2057  013274  000042                          42                              ;TELL OPR. TO ADJUST R42
2058  013276  004737  013734                  JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
2059
2060                                  ;OFFSET ADJUST FOR DAC 3
2061
2062                                  ;*      MOV     $TMDAT,@DAC3    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC3
2063  013312  004537  014704                  JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
2064  013316  020000                          BIT13
2065
2066                                  ;*      MOV     $TMDAT,@DAC3    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC3
2067  013330  104401                          TYPE
2068  013332  020467                          R44                             ;TELL OPR. TO ADJUST R44
2069  013334  004737  013734                  JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
2070
2071                                  ;GAIN ADJUST FOR DAC 2
2072  013340  004537  014532                  JSR     R5,SNDVLT
2073  013344  020773                          P49976
2074  013346  012737  007777  001534          MOV     #7777,$TMDAT            ;LOAD DAC 2
2075
2076                                  ;*      MOV     $TMDAT,@DAC2    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC2
2077  013364  004537  014704                  JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
2078  013370  060000                          BIT14!BIT13
```

```
2079
2080  013372  104401                    TYPE
2081  013374  020646                    R41                     ;TELL OPR. TO ADJUST R41
2082  013376  004737  013734            JSR     PC,CSPACE       ;WAIT FOR OPR. TO TYPE "SPACE"
2083
2084                                ;GAIN ADJUST FOR DAC 3
2085
2086                                ;*      MOV     $TMDAT,@DAC3    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC3
2087  013412  004537  014704            JSR     R5,CROSS        ;LOAD THE RELAYS WITH BIT13
2088  013416  020000                    BIT13
2089
2090  013420  104401                    TYPE
2091  013422  020713                    R43                     ;TELL OPR. TO ADJUST R43
2092  013424  004737  013734            JSR     PC,CSPACE       ;WAIT FOR OPR. TO TYPE "SPACE"
2093
2094                                ;DAC 2 D/A FUNCTION CHECK
2095
2096  013430  004537  015112            JSR     R5,FUNDAC
2097  013434  001544                    DAC2
2098  013436  000051                    CH51
2099
2100                                ;DAC 3 D/A FUNCTION CHECK
2101
2102  013440  004537  015112            JSR     R5,FUNDAC
2103  013444  001546                    DAC3
2104  013446  000052                    CH52
2105
2106  013450  104401                    TYPE
2107  013452  017716                    CALDON
2108  013454  000137  011456            JMP     AUTCAL
2109
2110  013460  022626              AUTRAP: CMP     (SP)+,(SP)+
2111  013462  012737  000006  000004    MOV     #6,ERRVEC
2112  013470  104401                    TYPE
2113  013472  020162                    AUTOER
2114  013474  000137  002430            JMP     CTRLC
2115                                    .SBTTL  MANUAL LOGIC TEST
2116
2117  013500  012706  001100    MANUL:  MOV     #STACK,SP       ;LOAD STACK
2118  013504  004737  002072            JSR     PC,LDTRAP       ;LOAD BUS ADDRESS
2119  013510  104401                    TYPE
2120  013512  017755                    MANHED
2121  013514  004737  013672    1$:     JSR     PC,ACTRLC       ;TEST FOR CTRL C
2122  013520  104407                    CKSWR
2123  013522  017700  165412            MOV     @SWR,R0         ;READ THE SWITCHES
2124  013526  010001                    MOV     R0,R1           ;COPY
2125  013530  042700  017777            BIC     #17777,R0       ;MASK OFF BITS
2126  013534  001003                    BNE     2$              ;BR IF NOT DAC 0
2127  013536  013702  001540            MOV     DAC0,R2         ;LOAD DAC 0 BUS ADDRESS
2128  013542  000424                    BR      10$
2129  013544  022700  020000    2$:     CMP     #BIT13,R0       ;TEST FOR DAC 1
2130  013550  001003                    BNE     3$              ;BR IF NOT
2131  013552  013702  001542            MOV     DAC1,R2         ;LOAD DAC 1 BUS ADDRESS
2132  013556  000416                    BR      10$
```

```
2133  013560  022700  040000          3$:     CMP     #BIT14,R0               ;TEST FOR DAC 2
2134  013564  001003                          BNE     4$                      ;BR IF NOT
2135  013566  013702  001544                  MOV     DAC2,R2                 ;LOAD DAC 2 BUS ADDRESS
2136  013572  000410                          BR      10$
2137  013574  022700  060000          4$:     CMP     #BIT14!BIT13,R0         ;TEST FOR DAC 3
2138  013600  001003                          BNE     5$                      ;BR IF NOT
2139  013602  013702  001546                  MOV     DAC3,R2                 ;LOAD DAC 3 BUS ADDRESS
2140  013606  000402                          BR      10$
2141  013610  013702  001536          5$:     MOV     STAT,R2                 ;LOAD STATUS REG. BUS ADDRESS
2142
2143  013614  010137  001124          10$:    MOV     R1,@#GDDAT
2144  013620  010237  001126                  MOV     R2,@#BDDAT
2145
2146                                  ;*      MOV     @#GDDAT,@#BDDAT ;/ PUT DATA FROM @#GDDAT TO DEVICE REG @#BDDAT
2147  013634  013703  001532                  MOV     DELAY,R3                ;PRESET THE DELAY
2148  013640  005303                  11$:    DEC     R3                      ;DELAY
2149  013642  001376                          BNE     11$
2150  013644  005037  001124                  CLR     @#GDDAT                 ;CLEAR THE REGISTER
2151
2152                                  ;*      MOV     @#GDDAT,@#BDDAT ;/ PUT DATA FROM @#GDDAT TO DEVICE REG @#BDDAT
2153  013660  013703  001532                  MOV     DELAY,R3                ;LOAD THE PRESET
2154  013664  005303                  12$:    DEC     R3                      ;DELAY
2155  013666  001376                          BNE     12$
2156  013670  000711                          BR      1$
2157
2158                                  ;TEST FOR CTRL C  -- DONT WAIT
2159
2160  013672  105777  165246          ACTRLC: TSTB    @#TKS                   ;INPUT FLAG ?
2161  013676  100014                          BPL     1$                      ;NO
2162  013700  017737  165242  013732          MOV     @#TKB,10$               ;READ THE CHARACTER
2163  013706  042737  177600  013732          BIC     #177600,10$
2164  013714  022737  000003  013732          CMP     #3,10$                  ;TEST FOR CTRL C
2165  013722  001002                          BNE     1$                      ;BR IF NOT
2166  013724  000137  002430                  JMP     CTRLC
2167  013730  000207                  1$:     RTS     PC                      ;EXIT
2168  013732  000000                  10$:    0
2169
2170                                  ;WAIT FOR A "SPACE" CHARACTER
2171
2172  013734  105777  165204          CSPACE: TSTB    @#TKS                   ;WAIT FOR CHAR
2173  013740  100375                          BPL     CSPACE
2174  013742  017737  165200  014012          MOV     @#TKB,10$               ;READ CHAR
2175  013750  042737  177600  014012          BIC     #177600,10$             ;MASK
2176  013756  022737  000003  014012          CMP     #3,10$                  ;TEST FOR CTRL C
2177  013764  001002                          BNE     1$
2178  013766  000137  002430                  JMP     CTRLC
2179  013772  022737  000040  014012  1$:     CMP     #40,10$                 ;TEST FOR SPACE
2180  014000  001001                          BNE     2$
2181  014002  000207                          RTS     PC                      ;EXIT
2182  014004  104401                  2$:     TYPE
2183  014006  017517                          QMARK
2184  014010  000751                          BR      CSPACE
2185
2186  014012  000000                  10$:    0
```

```
                                                .SBTTL  MANUAL DAC CALIBRATION
                                        CALDAC: MOV     #STACK,SP               ;LOAD STACK POINTER
                                                JSR     PC,LDTRAP               ;LOAD BUS ADDRESSES
                                        1$:     CKSWR                           ;TEST FOR CTRL G
                                                JSR     PC,ACTRLC               ;TEST FOR CTRL C
                                                MOV     @SWR,R0                 ;READ SWITCHES
                                                MOV     R0,$TMDAT               ;LOAD DAC #0

                                        ;*      MOV     $TMDAT,@DAC0    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC0

                                        ;*      MOV     $TMDAT,@DAC1    ;/ PUT DATA FROM $TMdAT TO DEVICE REG DAC1

                                        ;*      MOV     $TMDAT,@DAC2    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC2

                                        ;*      MOV     $TMDAT,@DAC3    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC3
                                                BR      1$

                                                .SBTTL  DYNAMIC DAC CALIBRATION

                                        DYNCAL: MOV     #STACK,SP               ;LOAD STACK POINTER
                                                JSR     PC,LDTRAP               ;LOAD BUS ADDRESSES
                                        1$:     CKSWR                           ;TEST FOR CTRL G
                                                JSR     PC,ACTRLC               ;TEST FOR CTRL C
                                                MOV     @SWR,R0                 ;READ SWR
                                                JSR     PC,10$                  ;LOAD THE SWR VALUE TO ALL DACS
                                                CLR     R0                      ;CLEAR R0
                                                JSR     PC,10$                  ;LOAD ALL DAC'S WITH 0
                                                BR      1$

                                        10$:

                                        ;*      MOV     $TMDAT,@DAC0    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC0

                                        ;*      MOV     $TMDAT,@DAC1    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC1

                                        ;*      MOV     $TMDAT,@DAC2    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC2

                                        ;*      MOV     $TMDAT,@DAC3    ;/ PUT DATA FROM $TMDAT TO DEVICE REG DAC3
                                                MOV     #20,R0                  ;LOAD DELAY COUNTER
                                        11$:    DEC     R0                      ;DELAY
                                                BPL     11$                     ;WAIT
                                                RTS     PC                      ;EXIT


                                                .SBTTL  SUBROUTINE TO ERASE STORAGE SCOPE SCREEN
                                        CLRVC:  CKSWR
                                                BIT     #BIT5,@SWR              ;TEST SR BIT 5
                                                BEQ     3$                      ;BR IF NOT A STORAGE SCOPE
                                                MOV     #BIT10,$TMDAT           ;ERASE THE SCREEN

                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
```

```
2241                                        ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
2242  014254  052737  010000  001534                BIS     #BIT12,$TMDAT
2243
2244                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
2245  014272  012700  000020                        MOV     #20,R0          ;SET UP DELAY
2246  014276  005001                                CLR     R1
2247  014300  032777  000040  164632  1$:    BIT     #BIT5,@SWR      ;TEST IF NOT STORAGE SCOPE
2248  014306  001430                                BEQ     3$
2249
2250                                        ;*      MOV     @STAT,$TMDAT    ;/READ DEVICE REG STAT,PUT DATA IN $TMDAT.
2251  014320  105737  001534                        TSTB    $TMDAT
2252  014324  100421                                BMI     3$              ;BRANCH IF SET
2253  014326  005301                                DEC     R1              ;DELAY
2254  014330  001363                                BNE     1$
2255  014332  004737  013672                        JSR     PC,ACTRLC       ;TEST FOR CTRL C
2256  014336  005300                                DEC     R0              ;DELAY
2257  014340  001357                                BNE     1$
2258  014342  037727  164572  010000                BIT     @SWR,#SW12      ;TEST INHIBIT PRINTOUT
2259  014350  001002                                BNE     2$
2260  014352  104401                                TYPE
2261  014354  016135                                MES3
2262  014356  104407                        2$:     CKSWR                   ;TEST FOR CTRL C
2263  014360  005777  164554                        TST     @SWR            ;TEST @SWR
2264  014364  100001                                BPL     3$
2265  014366  000000                                HALT                    ;ERASE RETURN FAILED TO SET READY
2266  014370  000207                        3$:     RTS     PC
2267
2268                                        .SBTTL  SUBROUTINE TO DRAW A HORIZONTAL LINE
2269  014372  012737  000000  001534  LOADVC: MOV     #0,$TMDAT       ;CLEAR STATUS
2270
2271                                        ;*      MOV     $TMDAT,@STAT    ;/ PUT DATA FROM $TMDAT TO DEVICE REG STAT
2272  014410  012737  007777  021156                MOV     #7777,TEMP1
2273  014416  013700  001536                        MOV     STAT,R0
2274  014422  032777  000010  164510                BIT     #SW03,@SWR      ;TEST SR BIT 3
2275  014430  001405                                BEQ     4$              ;BR IF DAC #0 AND 1
2276  014432  013701  001544                        MOV     DAC2,R1 ;LOAD X AXIS
2277  014436  013702  001546                        MOV     DAC3,R2 ;LOAD Y AXIS
2278  014442  000404                                BR      5$
2279  014444  013701  001540                4$:     MOV     DAC0,R1
2280  014450  013702  001542                        MOV     DAC1,R2
2281  014454  012710  002000                5$:     MOV     #BIT10,(0)      ;SET STORE MODE
2282  014460  013712  021156                        MOV     TEMP1,(2)
2283  014464  012711  007777                1$:     MOV     #7777,(1)
2284  014470  000402                                BR      3$
2285  014472  162711  000010                2$:     SUB     #10,(1)
2286  014476  005210                        3$:     INC     (0)
2287  014500  105710                                TSTB    (0)
2288  014502  100376                                BPL     .-2
2289  014504  022711  000007                        CMP     #7,(1)
2290  014510  001370                                BNE     2$
2291  014512  104407                                CKSWR                   ;TEST FOR CTRL G
2292  014514  004737  013672                        JSR     PC,ACTRLC               ;TEST FOR CTRL C
2293  014520  162712  000010                        SUB     #10,(2)
2294  014524  100357                                BPL     1$
```

```
2295  014526  000207                          RTS     PC
2296
2297  014530  167772                FILZ:     167772                  ;DR11-C OUTPUT CONTROL REGISTER
2298
2299                                ;SUBROUTINE TO LOAD THE EDC VOLTAGE
2300                                ;DATA FORMAT IS:        STX
2301                                ;                       P OR N
2302                                ;                       N VOLTS
2303                                ;                       O TENTHS VOLTS
2304                                ;                       O
2305                                ;                       O
2306                                ;                       O
2307                                ;                       O
2308                                ;                       V       VOLTS
2309                                ;                       ETX
2310
2311  014532  012500                SNDVLT: MOV     (R5)+,R0                ;LOAD POINTER
2312  014534  112001                2$:     MOVB    (R0)+,R1                ;GET A BYTE
2313  014536  001446                        BEQ     3$                      ;BR IF TERM.
2314  014540  005101                        COM     R1                      ;CONVERT IT
2315  014542  110137  001534               MOVB    R1,$TMDAT               ;LOAD FUNNY DATA BYTE
2316
2317                                ;*      MOV     $TMDAT,@FILZ    ;/ PUT DATA FROM $TMDAT TO DEVICE REG FILZ
2318  014556  012701  001000               MOV     #1000,R1                ;LOAD DELAY
2319  014562  005301                5$:     DEC     R1                      ;DELAY
2320  014564  001376                        BNE     5$
2321
2322                                ;*      MOV     @FILZ,$TMDAT    ;/READ DEVICE REG FILZ,PUT DATA IN $TMDAT.
2323  014576  042737  000200  001534        BIC     #BIT7,$TMDAT
2324
2325                                ;*      MOV     $TMDAT,@FILZ    ;/ PUT DATA FROM $TMDAT TO DEVICE REG FILZ
2326  014614  012701  001000               MOV     #1000,R1                ;LOAD DELAY COUNT
2327  014620  005301                1$:     DEC     R1                      ;DELAY
2328  014622  001376                        BNE     1$
2329
2330                                ;*      MOV     @FILZ,$TMDAT    ;/READ DEVICE REG FILZ,PUT DATA IN $TMDAT.
2331  014634  052737  000200  001534        BIS     #BIT7,$TMDAT
2332
2333                                ;*      MOV     $TMDAT,@FILZ    ;/ PUT DATA FROM $TMDAT TO DEVICE REG FILZ
2334  014652  000730                        BR      2$                      ;DO MORE DATA
2335
2336  014654  012701  000000        3$:     MOV     #0,R1                   ;LOAD DELAY
2337  014660  012737  000177  001534        MOV     #177,$TMDAT             ;SET BITS 0-6
2338
2339                                ;*      MOV     $TMDAT,@FILZ    ;/ PUT DATA FROM $TMDAT TO DEVICE REG FILZ
2340  014676  005301                4$:     DEC     R1                      ;DELAY
2341  014700  001376                        BNE     4$
2342  014702  000205                        RTS     R5                      ;EXIT
2343
2344  014704  012537  014754        CROSS:  MOV     (R5)+,10$       ;READ REG.
2345  014710  052737  000177  014754        BIS     #177,10$                ;LOAD LOW 7 BITS
2346  014716  013737  014754  001534        MOV     10$,$TMDAT              ;LOAD RELAYS
2347
2348                                ;*      MOV     $TMDAT,@FILZ    ;/ PUT DATA FROM $TMDAT TO DEVICE REG FILZ
```

```
2349  014734  012700  000001              MOV     #1,R0                   ;LOAD DELAY
2350  014740  005001                      CLR     R1
2351  014742  005301              1$:     DEC     R1                      ;DELAY
2352  014744  001376                      BNE     1$
2353  014746  005300                      DEC     R0                      ;DELAY,
2354  014750  001374                      BNE     1$                      ;DONE ?
2355  014752  000205                      RTS     R5                      ;EXIT
2356  014754  000000              10$:    0
2357                                      .SBTTL  TIMER ROUTINE FOR VISUAL TEST PATTERNS
2358                              ; ENTER VIA JSR PC,TIMER
2359
2360  014756  104407              TIMER:  CKSWR
2361  014760  004737  013672              JSR     PC,ACTRLC               ;TEST FOR CTRL C
2362  014764  017737  164150  021150      MOV     @SWR,TIMSV
2363
2364  014772  032737  000400  021150  TIMERA: BIT     #BIT8,TIMSV
2365  015000  001006                      BNE     TIMER2          ;BIT 8 SET ?
2366  015002  005337  021152              DEC     TICKS           ;NO, DECREMENT TICKS
2367  015006  001002                      BNE     TIMER1
2368  015010  062716  000002              ADD     #2,(6)          ;ADD 2 TO STACK POINTER
2369  015014  000207              TIMER1: RTS     PC              ;RETURN
2370
2371                              ; SWR 8=1 SELECT TEST TO LOCK ON
2372                              ; SWR 2-0= TEST NUMBER
2373
2374  015016  042737  177770  021150  TIMER2: BIC     #177770,TIMSV
2375  015024  006337  021150              ASL     TIMSV
2376  015030  062737  015052  021150      ADD     #ROUTPT,TIMSV
2377  015036  017737  004106  021150      MOV     @TIMSV,TIMSV
2378  015044  022600                      CMP     (SP)+,R0
2379  015046  000177  004076      TIMER4: JMP     @TIMSV
2380
2381  015052  007402              ROUTPT: PIC0                    ;DISPLAY A HORIZONTAL LINE
2382  015054  007502                      PIC1                    ;DISPLAY A VERTICAL LINE
2383  015056  010024                      PIC3                    ;DISPLAY A SQUARE
2384  015060  010562                      PIC4                    ;DISPALY A "X"
2385  015062  011264                      PIC6                    ;DISPLAY CHARACTER SET
2386
2387  015064  013737  015110  021160  CHTIME: MOV     CPTYPE,BRLEV1
2388  015072  005337  021160      1$:     DEC     BRLEV1
2389  015076  001403                      BEQ     2$
2390  015100  006337  021152              ASL     TICKS
2391  015104  000772                      BR      1$
2392  015106  000207              2$:     RTS     PC
2393
2394  015110  000001              CPTYPE: 1
2395
2396                              ;DO 32 CONVERSIONS ON EACH PRESET VALUE OF THE DAC'S
2397                              ;AND SUBTRACT THE FIRST RESULT FROM EACH OF THE RESULTS
2398
2399  015112  013537  015202      FUNDAC: MOV     @(R5)+,XDACUT           ;GET BUS ADDRESS OF ARUT DAC
2400  015116  012537  015142              MOV     (R5)+,60$               ;GET CHANNEL INPUT FOR ARKG
2401  015122  012777  007600  000052      MOV     #7600,@XDACUT           ;PRESET THE DAC
2402  015130  012737  007600  001124      MOV     #7600,$GDDAT            ;LOAD EXPECTED
```

```
2403
2404  015136  004537  015204          1$:     JSR     R5,CONVRT                       ;CONVERT USING ARKG
2405  015142  000051                  60$:    CH51                                    ;
2406
2407  015144  012737  000010  015420          MOV     #10,SPREAD                      ;LOAD LIMIT
2408  015152  004737  015422                  JSR     PC,COMPAR                       ;TEST IF WITHIN LIMIT
2409  015156  000401                          BR      2$              ;;BR IF WITHIN
2410  015160  104016                          ERROR   16              ;SELECTED DAC HAS A LINEARITY ERROR
2411
2412  015162  162777  000400  000012  2$:     SUB     #400,@XDACUT                    ;UPDATE DAC
2413  015170  162737  000400  001124          SUB     #400,$GDDAT                     ;UPDATE GOOd VALUE
2414  015176  001357                          BNE     1$              ;;BR IF NOT FINISHED
2415
2416  015200  000000                  10$:    0
2417  015202  000000                  XDACUT: 0
2418                                  ;SUBROUTINE TO GET AVERAGE OF 32 CONVERSIONS FROM THE KNOWN GOOd AR 11
2419
2420  015204  012537  015374          CONVRT: MOV     (R5)+,10$
2421  015210  013737  015374  015416          MOV     10$,CHANL
2422  015216  000337  015374                  SWAB    10$
2423  015222  012737  000020  015376          MOV     #16.,11$
2424  015230  005037  015412                  CLR     ADEND
2425  015234  013737  015374  001534          MOV     10$,$TMDAT
2426
2427                                  ;*      MOV     $TMDAT,@GADCS   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GADCS
2428  015252  005237  001534          2$:     INC     $TMDAT
2429
2430                                  ;*      MOV     $TMDAT,@GADCS   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GADCS
2431  015266                          1$:
2432
2433                                  ;*      MOV     @GADCS,$TMDAT   ;/READ DEVICE REG GADCS,PUT DATA IN $TMDAT.
2434  015276  105737  001534                  TSTB    $TMDAT
2435  015302  100371                          BPL     1$
2436
2437                                  ;*      MOV     @GADDBR,$TMDAT  ;/READ DEVICE REG GADDBR,PUT DATA IN $TMDAT.
2438  015314  063737  001534  015412          ADD     $TMDAT,ADEND
2439
2440                                  ;*      MOV     $TMDAT,@GADDBR  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GADDBR
2441  015332  005337  015376                  DEC     11$
2442  015336  001345                          BNE     2$
2443  015340  006237  015412                  ASR     ADEND
2444  015344  006237  015412                  ASR     ADEND
2445  015350  006237  015412                  ASR     ADEND
2446  015354  006237  015412                  ASR     ADEND
2447  015360  005537  015412                  ADC     ADEND                           ;ROUND UP
2448  015364  013737  015412  001126          MOV     ADEND,$BDDAT
2449  015372  000205                          RTS     R5
2450
2451  015374  000000                  10$:    0
2452  015376  000000                  11$:    0
2453  015400  001754                  V1754:  1754
2454  015402  000024                  V24:    24
2455  015404  170460                  GADCS:  170460
2456  015406  170462                  GADDBR: 170462
```

```
2457  015410  170470                  GSTAT:  170470
2458  015412  000000                  ADEND:  0
2459  015414  000000                  FCHANL: 0
2460  015416  000000                  CHANL:  0
2461  015420  000000                  SPREAD: 0
2462
2463  015422  010046                  COMPAR: MOV     RO,-(SP)
2464  015424  010146                          MOV     R1,-(SP)
2465  015426  013700  001124                  MOV     $GDDAT,RO               ;LOAD RO
2466  015432  013701  001126                  MOV     $BDDAT,R1               ;LOAD R1
2467  015436  160100                          SUB     R1,RO                   ;SUBTRACT
2468  015440  100001                          BPL     8$                      ;
2469  015442  005400                          NEG     RO
2470  015444  020037  015420          8$:     CMP     RO,SPREAD               ;MAGNITUDE OF DIFF. IN RO
2471  015450  003405                          BLE     10$
2472  015452  012601                  9$:     MOV     (SP)+,R1
2473  015454  012600                          MOV     (SP)+,RO
2474  015456  062716  000002                  ADD     #2,(SP)
2475  015462  000207                          RTS     PC
2476
2477  015464  012601                  10$:    MOV     (SP)+,R1
2478  015466  012600                          MOV     (SP)+,RO
2479  015470  000207                          RTS     PC
2480
2481                                          .SBTTL  ASCII MESSAGES
2482
2483  015472  005015  040412  030501  TITLE:  .ASCII  <15><12><12>'AA11-K DIAGNOSTIC TEST, (MAINDEC-11-DRLPB-A0)'<15><12>
2484  015500  026461  020113  044504
2485  015506  043501  047516  052123
2486  015514  041511  052040  051505
2487  015522  026124  024040  040515
2488  015530  047111  042504  026503
2489  015536  030461  042055  046122
2490  015544  041120  040455  024460
2491  015552  005015
2492  015554  042523  042514  052103          .ASCII  /SELECT TESTS BY TYPING A TWO LETTER I.D./<15><12>
2493  015562  052040  051505  051524
2494  015570  041040  020131  054524
2495  015576  044520  043516  040440
2496  015604  052040  047527  046040
2497  015612  052105  042524  020122
2498  015620  027111  027104  005015
2499  015626  046101  035011  052501          .ASCII  /AL     :AUTO LOGIC TEST/<15><12>
2500  015634  047524  046040  043517
2501  015642  041511  052040  051505
2502  015650  006524     012
2503  015653     101  004504  040472          .ASCII  /AD     :AUTO DISPLAY TEST (DISPLAY SCOPE CONNECTED)/<15><12>
2504  015660  052125  020117  044504
2505  015666  050123  040514  020131
2506  015674  042524  052123  024040
2507  015702  044504  050123  040514
2508  015710  020131  041523  050117
2509  015716  020105  047503  047116
2510  015724  041505  042524  024504
```

```
2511  015732  005015
2512  015734  046115  035011  040515          .ASCII  /ML     :MANUAL LOGIC TEST (SWR 15-13 = REGISTER  12-0 = DATA)/<15><12>
2513  015742  052516  046101  046040
2514  015750  043517  041511  052040
2515  015756  051505  020124  051450
2516  015764  051127  030440  026465
2517  015772  031461  036440  051040
2518  016000  043505  051511  042524
2519  016006  020122  030440  026462
2520  016014  020060  020075  040504
2521  016022  040524  006451     012
2522  016027     115  004504  046472          .ASCII  /MD     :MANUAL DISPLAY (NO DISPLAY SCOPE)/<15><12>
2523  016034  047101  040525  020114
2524  016042  044504  050123  040514
2525  016050  020131  047050  020117
2526  016056  044504  050123  040514
2527  016064  020131  041523  050117
2528  016072  024505  005015
2529  016076  041515  035011  040515          .ASCII  /MC     :MANUAL CALIBRATION LOOP/<15><12>
2530  016104  052516  046101  041440
2531  016112  046101  041111  040522
2532  016120  044524  047117  046040
2533  016126  047517  006520     012
2534  016133     056     000          .ASCIZ  /./
2535  016135     015  042412  040522  MES3:   .ASCIZ  <15><12>"ERASE RETURN FAILED TO SET READY"<15><12>
2536  016142  042523  051040  052105
2537  016150  051125  020116  040506
2538  016156  046111  042105  052040
2539  016164  020117  042523  020124
2540  016172  042522  042101  006531
2541  016200  000012
2542  016202  005015  041523  050117  MES6:   .ASCIZ  <15><12>'SCOPE ADJUSTMENT TEST'
2543  016210  020105  042101  052512
2544  016216  052123  042515  052116
2545  016224  052040  051505  000124
2546  016232  005015  053523  034122  MES15:  .ASCII  <15><12>/SWR8 AND 0 THRU 2 CONTROL PATTERN/<15><12>
2547  016240  040440  042116  030040
2548  016246  052040  051110  020125
2549  016254  020062  047503  052116
2550  016262  047522  020114  040520
2551  016270  052124  051105  006516
2552  016276     012
2553  016277     123  051127  041040          .ASCII  'SWR BIT 3 SELECTS DAC 0+1 OR DAC 2+3'<15><12>
2554  016304  052111  031440  051440
2555  016312  046105  041505  051524
2556  016320  042040  041501  030040
2557  016326  030453  047440  020122
2558  016334  040504  020103  025462
2559  016342  006463     012
2560  016345     123  051127  041040          .ASCII  /SWR BIT 5 SELECTS STORAGE SCOPE MODE/<15><12>
2561  016352  052111  032440  051440
2562  016360  046105  041505  051524
2563  016366  051440  047524  040522
2564  016374  042507  051440  047503
```

```
2565   016402  042520  046440  042117
2566   016410  006505     012
2567   016413     123  051127  041040          .ASCII  /SWR BIT 7 ENABLES VERTICAL SETTLING TEST/<15><12>
2568   016420  052111  033440  042440
2569   016426  040516  046102  051505
2570   016434  053040  051105  044524
2571   016442  040503  020114  042523
2572   016450  052124  044514  043516
2573   016456  052040  051505  006524
2574   016464     012
2575   016465     123  051127  041040          .ASCIZ  /SWR BIT 9 SELECTS TWO'S COMPLEMENT DISPLAY MODE/<15><12>
2576   016472  052111  034440  051440
2577   016500  046105  041505  051524
2578   016506  052040  047527  051447
2579   016514  041440  046517  046120
2580   016522  046505  047105  020124
2581   016530  044504  050123  040514
2582   016536  020131  047515  042504
2583   016544  005015     000
2584   016547     123  040524  052524  EM1:    .ASCIZ  /STATUS REGISTER IN ERROR/
2585   016554  020123  042522  044507
2586   016562  052123  051105  044440
2587   016570  020116  051105  047522
2588   016576  000122
2589   016600  040504  030103  051040  EM2:    .ASCIZ  /DAC0 REGISTER IN ERROR/
2590   016606  043505  051511  042524
2591   016614  020122  047111  042440
2592   016622  051122  051117     000
2593   016627     104  041501  020061  EM3:    .ASCIZ  /DAC1 REGISTER IN ERROR/
2594   016634  042522  044507  052123
2595   016642  051105  044440  020116
2596   016650  051105  047522  000122
2597   016656  040504  031103  051040  EM4:    .ASCIZ  /DAC2 REGISTER IN ERROR/
2598   016664  043505  051511  042524
2599   016672  020122  047111  042440
2600   016700  051122  051117     000
2601   016705     104  041501  020063  EM5:    .ASCIZ  /DAC3 REGISTER IN ERROR/
2602   016712  042522  044507  052123
2603   016720  051105  044440  020116
2604   016726  051105  047522  000122
2605   016734  040501  030461  045455  EM6:    .ASCIZ  /AA11-K FAILED TO INTERRUPT/
2606   016742  043040  044501  042514
2607   016750  020104  047524  044440
2608   016756  052116  051105  052522
2609   016764  052120     000
2610   016767     101  030501  026461  EM7:    .ASCIZ  /AA11-K INTERRUPTED IN ERROR/
2611   016774  020113  047111  042524
2612   017002  051122  050125  042524
2613   017010  020104  047111  042440
2614   017016  051122  051117     000
2615   017023     102  051525  052040  EM10:   .ASCIZ  /BUS TIME-OUT WHEN REFERENCING THE AA11-K/
2616   017030  046511  026505  052517
2617   017036  020124  044127  047105
2618   017044  051040  043105  051105
```

```
2619  017052  047105  044503  043516
2620  017060  052040  042510  040440
2621  017066  030501  026461  000113
2622  017074  040501  030461  045455  EM11:   .ASCIZ  /AA11-K READY BIT ERROR/
2623  017102  051040  040505  054504
2624  017110  041040  052111  042440
2625  017116  051122  051117     000
2626  017123     120  053517  051105  EM12:   .ASCIZ  /POWER SUPPLY VOLTAGE WAS INCORRECT/
2627  017130  051440  050125  046120
2628  017136  020131  047526  052114
2629  017144  043501  020105  040527
2630  017152  020123  047111  047503
2631  017160  051122  041505  000124
2632  017166  047117  020105  044502  EM13:   .ASCIZ  /ONE BIT THE DUST/
2633  017174  020124  044124  020105
2634  017202  052504  052123     000
2635  017207     101  030501  026461  EM14:   .ASCIZ  /AA11-K LOGIC SIGNAL HIGH OUTPUT TO LOW/
2636  017214  020113  047514  044507
2637  017222  020103  044523  047107
2638  017230  046101  044040  043511
2639  017236  020110  052517  050124
2640  017244  052125  052040  020117
2641  017252  047514  000127
2642  017256  040501  030461  045455  EM15:   .ASCIZ  /AA11-K LOGIC SIGNAL LOW OUTPUT TO HIGH/
2643  017264  046040  043517  041511
2644  017272  051440  043511  040516
2645  017300  020114  047514  020127
2646  017306  052517  050124  052125
2647  017314  052040  020117  044510
2648  017322  044107     000
2649  017325     123  046105  041505  EM16:   .ASCIZ  /SELECTED DAC HAS A LINEARITY ERROR/
2650  017332  042524  020104  040504
2651  017340  020103  040510  020123
2652  017346  020101  044514  042516
2653  017354  051101  052111  020131
2654  017362  051105  047522  000122
2655  017370  051105  050122  004503  DH1:    .ASCIZ  /ERRPC  IBASE   EXPECT    BAD/
2656  017376  041111  051501  020105
2657  017404  020040  054105  042520
2658  017412  052103  020040  020040
2659  017420  040502  000104
2660  017424  051105  050122  004503  DH6:    .ASCIZ  /ERRPC  IBASE/
2661  017432  041111  051501  000105
2662  017440  051105  050122  004503  DH13:   .ASCIZ  /ERRPC  IBASE   # FIRST # NOW/
2663  017446  041111  051501  004505
2664  017454  020043  044506  051522
2665  017462  004524  020043  047516
2666  017470  000127
2667  017472  051105  050122  004503  DH14:   .ASCIZ  /ERRPC  IBASE   GOOD    BAD/
2668  017500  041111  051501  004505
2669  017506  047507  042117  041011
2670  017514  042101     000
2671
2672  017517     015     012     077  QMARK:  .BYTE   15,12,77,15,12,0
```

```
2673  017522      015      012      000
2674  017525      136      103      015  CONTC:  .BYTE   136,103,15,12,56,0
2675  017530      012      056      000
2676  017533      015      012           FOUND1: .BYTE   15,12
2677  017535      120   047522   051107           .ASCIZ  /PROGRAM DETECTED /
2678  017542   046501   042040   052105
2679  017550   041505   042524   020104
2680  017556      000
2681  017557      050   024470   020040  FOUND2: .ASCIZ  /(8)   AA11-K(S)  /
2682  017564   040440   030501   026461
2683  017572   024113   024523   020040
2684  017600      000
2685  017601      015      012      000           .BYTE   15,12,0
2686  017604      015      012           SELD01: .BYTE   15,12
2687  017606   042523   020124   053523           .ASCIZ  /SET SWITCH TO SELECT DAC 0 AND 1/<15><12>
2688  017614   052111   044103   052040
2689  017622   020117   042523   042514
2690  017630   052103   042040   041501
2691  017636   030040   040440   042116
2692  017644   030440   005015      000
2693  017651      015      012           SELD23: .BYTE   15,12
2694  017653      123   052105   051440           .ASCIZ  /SET SWITCH TO SELECT DAC 2 AND 3/<15><12>
2695  017660   044527   041524   020110
2696  017666   047524   051440   046105
2697  017674   041505   020124   040504
2698  017702   020103   020062   047101
2699  017710   020104   006463   000012
2700  017716      015      012           CALDON: .BYTE   15,12
2701  017720   052501   047524   041440           .ASCIZ  /AUTO CALIBRATION COMPLETED/<15><12>
2702  017726   046101   041111   040522
2703  017734   044524   047117   041440
2704  017742   046517   046120   052105
2705  017750   042105   005015      000
2706  017755      015      012           MANHED: .BYTE   15,12
2707  017757      123   020127   032461           .ASCII  /SW 15-13 SELECT THE REGISTER/<15><12>
2708  017764   030455   020063   042523
2709  017772   042514   052103   052040
2710  020000   042510   051040   043505
2711  020006   051511   042524   006522
2712  020014      012
2713  020015      123   020127   031061           .ASCII  /SW 12-00 ARE LOADED INTO THE SELECTED REGISTER/<15><12>
2714  020022   030055   020060   051101
2715  020030   020105   047514   042101
2716  020036   042105   044440   052116
2717  020044   020117   044124   020105
2718  020052   042523   042514   052103
2719  020060   042105   051040   043505
2720  020066   051511   042524   006522
2721  020074      012
2722  020075      000           .BYTE   0
2723  020076      015      012  AUTOCL: .BYTE   15,12
2724  020100   052501   047524   041440           .ASCII  /AUTO CALIBRATION (AA11-K OPTION TEST AREA ONLY)/
2725  020106   046101   041111   040522
2726  020114   044524   047117   024040
```

```
2727  020122  040501  030461  045455
2728  020130  047440  052120  047511
2729  020136  020116  042524  052123
2730  020144  040440  042522  020101
2731  020152  047117  054514    051
2732  020157     015     012    000          .BYTE   15,12,0
2733  020162  005015  052502  020123  AUTOER: .ASCII  <15><12>/BUS TRAP WHEN LOADING VOLTAGE OR CHANGING A RELAY/<15><12>
2734  020170  051124  050101  053440
2735  020176  042510  020116  047514
2736  020204  042101  047111  020107
2737  020212  047526  052114  043501
2738  020220  020105  051117  041440
2739  020226  040510  043516  047111
2740  020234  020107  020101  042522
2741  020242  040514  006531    012
2742  020247     077  004477  040501          .ASCIZ  /??     AA11-K OPTION TEST AREA ??      /<15><12>
2743  020254  030461  045455  047440
2744  020262  052120  047511  020116
2745  020270  042524  052123  040440
2746  020276  042522  020101  037477
2747  020304  006411  000012
2748
2749  020310  005015  042101  052512  R38:    .ASCIZ  <15><12>/ADJUST R38 FOR A NULL ON THE METER/
2750  020316  052123  051040  034063
2751  020324  043040  051117  040440
2752  020332  047040  046125  020114
2753  020340  047117  052040  042510
2754  020346  046440  052105  051105
2755  020354     000
2756  020355     015  040412  045104  R40:    .ASCIZ  <15><12>/ADJUST R40 FOR A NULL ON THE METER/
2757  020362  051525  020124  032122
2758  020370  020060  047506  020122
2759  020376  020101  052516  046114
2760  020404  047440  020116  044124
2761  020412  020105  042515  042524
2762  020420  000122
2763  020422  005015  042101  052512  R22:    .ASCIZ  <15><12>/ADJUST R22 FOR A NULL ON THE METER/
2764  020430  052123  051040  031062
2765  020436  043040  051117  040440
2766  020444  047040  046125  020114
2767  020452  047117  052040  042510
2768  020460  046440  052105  051105
2769  020466     000
2770  020467     015  040412  045104  R44:    .ASCIZ  <15><12>/ADJUST R44 FOR A NULL ON THE METER/
2771  020474  051525  020124  032122
2772  020502  020064  047506  020122
2773  020510  020101  052516  046114
2774  020516  047440  020116  044124
2775  020524  020105  042515  042524
2776  020532  000122
2777  020534  005015  042101  052512  R37:    .ASCIZ  <15><12>/ADJUST R37 FOR A NULL ON THE METER/
2778  020542  052123  051040  033463
2779  020550  043040  051117  040440
2780  020556  047040  046125  020114
```

```
2781   020564  047117  052040  042510
2782   020572  046440  052105  051105
2783   020600     000
2784   020601     015  040412  045104  R39:    .ASCIZ  <15><12>/ADJUST R39 FOR A NULL ON THE METER/
2785   020606  051525  020124  031522
2786   020614  020071  047506  020122
2787   020622  020101  052516  046114
2788   020630  047440  020116  044124
2789   020636  020105  042515  042524
2790   020644  000122
2791   020646  005015  042101  052512  R41:    .ASCIZ  <15><12>/ADJUST R41 FOR A NULL ON THE METER/
2792   020654  052123  051040  030464
2793   020662  043040  051117  040440
2794   020670  047040  046125  020114
2795   020676  047117  052040  042510
2796   020704  046440  052105  051105
2797   020712     000
2798   020713     015  040412  045104  R43:    .ASCIZ  <15><12>/ADJUST R43 FOR A NULL ON THE METER/
2799   020720  051525  020124  032122
2800   020726  020063  047506  020122
2801   020734  020101  052516  046114
2802   020742  047440  020116  044124
2803   020750  020105  042515  042524
2804   020756  000122
2805           000001                  STX=1
2806           000003                  ETX=3
2807   020760     001          N50000: .BYTE   STX
2808   020761     116  030065  030060          .ASCII  /N500000V/
2809   020766  030060     126
2810   020771     003     000          .BYTE   ETX,0
2811   020773     001          P49976: .BYTE   STX
2812   020774  032120  034471  033067          .ASCII  /P499760V/
2813   021002  053060
2814   021004     003     000          .BYTE   ETX,0
2815                                   .EVEN
2816   021006  001116  001536  001124  DT1:    $ERRPC,STAT,$GDDAT,$BDDAT,0
2817   021014  001126  000000
2818   021020  001116  001540  001124  DT2:    $ERRPC,DAC0,$GDDAT,$BDDAT,0
2819   021026  001126  000000
2820   021032  001116  001542  001124  DT3:    $ERRPC,DAC1,$GDDAT,$BDDAT,0
2821   021040  001126  000000
2822   021044  001116  001544  001124  DT4:    $ERRPC,DAC2,$GDDAT,$BDDAT,0
2823   021052  001126  000000
2824   021056  001116  001546  001124  DT5:    $ERRPC,DAC3,$GDDAT,$BDDAT,0
2825   021064  001126  000000
2826   021070  001116  001536  000000  DT6:    $ERRPC,STAT,0
2827   021076  001116  001536  021164  DT13:   $ERRPC,STAT,EVER,$UNIT,0
2828   021104  001206  000000
2829   021110  001116  001536  001124  DT14:   $ERRPC,STAT,$GDDAT,$BDDAT,0
2830   021116  001126  000000
2831   021122  000000  000000  000000  DF0:    0,0,0,0,0,0,0,0
2832   021130  000000  000000  000000
2833   021136  000000  000000
2834   021142  000000          LOW:    0
```

```
 2835  021144  000000                HIGH:   0
 2836  021146  000010                INCR:   10
 2837  021150  000000                TIMSV:  0
 2838  021152  000000                TICKS:  0
 2839  021154  000000                TEMp:   0
```

```
2840  021156  000000                  TEMP1:  0                       ;TEMPORARY STORAGE
2841  021160  000000                  BRLEV1: 0
2842  021162  000000                  BRLEV2: 0
2843  021164  000000                  EVER:   0
2844  021166  000000                  EVER1:  0
2845  021170  000000                  OPRIN:  0
2846  021172  000000                  P7CNT:  0
2847  021174  000000                  P7PNT:  0
2848
2849                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
2850
2851                                  ;;**********************************************************************
2852                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
2853                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
2854                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
2855                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
2856                                  ;*REPLACED WITH SPACES.
2857                                  ;*CALL:
2858                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
2859                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
2860
2861  021176                          $TYPDS:
2862  021176  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2863  021200  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2864  021202  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2865  021204  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2866  021206  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2867  021210  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
2868  021214  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
2869  021220  100004                          BPL     1$              ;;BR IF INPUT IS POS.
2870  021222  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
2871  021224  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
2872  021232  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
2873  021234  012703  021412                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
2874  021240  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
2875  021244  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
2876  021246  016001  021402                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
2877  021252  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
2878  021254  002402                          BLT     4$              ;;BR IF DONe
2879  021256  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
2880  021260  000774                          BR      3$
2881  021262  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
2882  021264  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
2883  021266  001002                          BNE     5$              ;;FALL THROUGH IF 0
2884  021270  105716                          TSTB    (SP)            ;;STILL DOInG LEADING 0'S?
2885  021272  100407                          BMI     7$              ;;BR IF YES
2886  021274  106316                  5$:     ASLB    (SP)            ;;MSD?
2887  021276  103003                          BCC     6$              ;;BR IF NO
2888  021300  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
2889  021306  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
2890  021312  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
2891  021316  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
2892  021320  005720                          TST     (R0)+           ;;JUST INCREMENTING
2893  021322  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
```

```
2894   021326  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
2895   021330  003002                          BGT     8$              ;;GO TO EXIT
2896   021332  010502                          MOV     R5,R2           ;;GET THE LSD
2897   021334  000764                          BR      6$              ;;GO CHANGE TO ASCII
2898   021336  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
2899   021340  100003                          BPL     9$              ;;BR IF NO
2900   021342  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
2901   021350  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
2902   021352  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
2903   021354  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
2904   021356  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2905   021360  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2906   021362  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2907   021364  104401  021412                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
2908   021370  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
2909   021376  012616                          MOV     (SP)+,(SP)
2910   021400  000002                          RTI                     ;;RETURN TO USER
2911   021402  023420                  $DTBL:  10000.
2912   021404  001750                          1000.
2913   021406  000144                          100.
2914   021410  000012                          10.
2915   021412  000004                  $DBLK:  .BLKW   4
2916                                   .SBTTL  SCOPE HANDLER ROUTINE
2917
2918                                   ;;**************************************************************
2919                                   ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
2920                                   ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
2921                                   ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
2922                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2923                                   ;*SW14=1         LOOP ON TEST
2924                                   ;*SW11=1         INHIBIT ITERATIONS
2925                                   ;*SW08=1         LOOP ON TEST IN SWR<7:0>
2926                                   ;*CALL
2927                                   ;*      SCOPE           ;;SCOPE=IOT
2928
2929   021422                          $SCOPE:
2930   021422  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
2931   021424  004737  013672                  JSR     PC,ACTRLC
2932   021430  032777  040000  157502  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
2933   021436  001070                          BNE     $OVER           ;;YES IF SW14=1
2934                                   ;#####START OF CODE FOR THE XOR TESTER#####
2935   021440  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
2936                                                                   ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
2937   021442  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
2938   021446  012737  021466  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
2939   021454  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
2940   021460  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
2941   021464  000446                          BR      $SVLAD          ;;GO TO THE NEXT TEST
2942   021466  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
2943   021470  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
2944   021474  000451                          BR      $OVER           ;;LOOP ON THE PRESENT TEST
2945   021476                          6$:;#####END OF CODE FOR THE XOR TESTER#####
2946   021476  032777  000400  157434          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
2947   021504  001404                          BEQ     2$              ;;BR IF NO
```

```
2948  021506  127737  157426  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
2949  021514  001441                          BEQ     $OVER           ;;BR IF YES
2950  021516  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
2951  021522  001404                          BEQ     3$              ;;BR IF NO
2952  021524  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
2953  021530  005037  001166                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
2954  021534  032777  004000  157376  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
2955  021542  001011                          BNE     1$              ;;BR IF YES
2956  021544  005737  001202                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
2957  021550  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
2958  021552  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
2959  021556  023727  001166  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
2960  021564  002015                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
2961  021566  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
2962  021574  013737  021634  001166          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
2963  021602  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
2964  021606  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
2965  021614  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
2966  021620  013777  001102  157314  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
2967  021626  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
2968  021632  000002                          RTI                     ;;FIXES PS
2969  021634  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
2970                                  .SBTTL  ERROR HANDLER ROUTINE
2971
2972                                  ;;**************************************************************
2973                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2974                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2975                                  ;*AND GO TO $ERRTYP ON ERROR
2976                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2977                                  ;*SW15=1         HALT ON ERROR
2978                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
2979                                  ;*CALL
2980                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
2981
2982  021636                          $ERROR:
2983  021636  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
2984  021640  004737  013672                  JSR     PC,@CTRLC
2985  021644  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
2986  021650  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
2987  021652  013777  001102  157262          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
2988  021660  005237  001112                  INC     $ERTTL          ;;INC THE ERROR COUNT
2989  021664  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
2990  021670  162737  000002  001116          SUB     #2,$ERRPC
2991  021676  117737  157214  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
2992  021704  032777  020000  157226          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
2993  021712  001004                          BNE     20$             ;;SKIP TYPEOUTS
2994  021714  004737  021766                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
2995  021720  104401  001171                  TYPE    ,$CRLF
2996  021724                          20$:
2997  021724  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
2998  021732  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
2999  021734  113737  001114  021746          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
3000  021742  004737  023746                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
3001  021746     000                  21$:    .BYTE   0
```

```
3002  021747  000                       .BYTE   0
3003  021750  000777          22$:      BR      22$             ;;APT ERROR LOOP
3004  021752  005777  157162  2$:       TST     @SWR            ;;HALT ON ERROR
3005  021756  100002                    BPL     3$              ;;SKIP IF CONTINUE
3006  021760  000000                    HALT                    ;;HALT ON ERROR!
3007  021762  104407                    CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
3008  021764                    3$:
3009  021764  000002                    RTI                     ;;RETURN
3010                            .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
3011
3012                            ;;**************************************************************
3013                            ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
3014                            ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
3015                            ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
3016
3017  021766                    $ERRTYP:
3018  021766  104401  001171            TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
3019  021772  010046                    MOV     R0,-(SP)        ;;SAVE R0
3020  021774  005000                    CLR     R0              ;;PICKUP THE ITEM INDEX
3021  021776  153700  001114            BISB    @#$ITEMB,R0
3022  022002  001004                    BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
3023                                                            ;;TYPE THE PC OF THE ERROR
3024  022004  013746  001116            MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
3025                                                            ;;ERROR ADDRESS
3026  022010  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3027  022012  000426                    BR      6$              ;;GET OUT
3028  022014  005300          1$:       DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
3029  022016  006300                    ASL     R0              ;;      WORK FOR THE ERROR TABLE
3030  022020  006300                    ASL     R0
3031  022022  006300                    ASL     R0
3032  022024  062700  001256            ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
3033  022030  012037  022040            MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
3034  022034  001404                    BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
3035  022036  104401                    TYPE                    ;;TYPE THE "ERROR MESSAGE"
3036  022040  000000          2$:       .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
3037  022042  104401  001171            TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
3038  022046  012037  022056  3$:       MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
3039  022052  001404                    BEQ     5$              ;;SKIP TYPEOUT IF 0
3040  022054  104401                    TYPE                    ;;TYPE THE "DATA HEADER"
3041  022056  000000          4$:       .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
3042  022060  104401  001171            TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
3043  022064  011000          5$:       MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
3044  022066  001004                    BNE     7$              ;;GO TYPE THE DATA
3045  022070  012600          6$:       MOV     (SP)+,R0        ;;RESTORE R0
3046  022072  104401  001171            TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
3047  022076  000207                    RTS     PC              ;;RETURN
3048  022100                    7$:
3049  022100  013046                    MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
3050  022102  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3051  022104  005710                    TST     (R0)            ;;IS THERE ANOTHER NUMBER?
3052  022106  001770                    BEQ     6$              ;;BR IF NO
3053  022110  104401  022116            TYPE    ,8$             ;;TYPE TWO(2) SPACES
3054  022114  000771                    BR      7$              ;;LOOP
3055  022116  020040  000       8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
```

```
                022122                          .EVEN
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
022122  012737  022266  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
022130  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
022136  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
022140  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
022142  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
022144  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
022146  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
022150  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
022152  017746  156762                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
022156  010637  022272                  MOV     SP,$SAVR6       ;;SAVE SP
022162  012737  022174  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
022170  000000                          HALT
022172  000776                          BR      .-2             ;;HANG UP

                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
022174  012737  022266  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWn
022202  013706  022272                  MOV     $SAVR6,SP       ;;GET SP
022206  005037  022272                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
022212  005237  022272          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
022216  001375                          BNE     1$              ;;OF  WORD
022220  012677  156714                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
022224  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
022226  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
022230  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
022232  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
022234  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
022236  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
022240  012737  022122  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
022246  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
022254  104401                          TYPE                    ;;REPORT THE POWER FAILURE
022256  022274          $PWRMG: .WORD   PWRMSG                  ;;POWER FAIL MESSAGE POINTER
022260  012716                  MOV     (PC)+,(SP)              ;;RESTART AT BEGIN
022262  001554          $PWRAD: .WORD   BEGIN                   ;;RESTART ADDRESS
                                RTI
022266  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
022270  000776                  BR      .-2                     ;; BEFORE THE POWER DOWN WAS COMPLETE
022272  000000          $SAVR6: 0                               ;;PUT THE SP HERE
```

```
3099  022274  005015  042522  052123  PWRMSG: .ASCIZ  <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12><12>
3100  022302  051101  044524  043516
3101  022310  040440  052106  051105
3102  022316  040440  050040  053517
3103  022324  051105  043040  044501
3104  022332  052514  042522  005015
3105  022340  000012
3106                                  .EVEN
3107
3108                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
3109
3110                                  ;;*****************************************************************
3111                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3112                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
3113                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3114                                  ;*CALL:
3115                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3116                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
3117                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3118                                  ;*      .BYTE   M               ;;M=1 OR 0
3119                                  ;*                                      ;;1=TYPE LEADING ZEROS
3120                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
3121                                  ;*
3122                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3123                                  ;*$TYPOS OR $TYPOC
3124                                  ;*CALL:
3125                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3126                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
3127                                  ;*
3128                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3129                                  ;*CALL:
3130                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3131                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
3132
3133  022342  017646  000000          $TYpOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
3134  022346  116637  000001  022565          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
3135  022354  112637  022567                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
3136  022360  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
3137  022364  000406                          BR      $TYpON
3138  022366  112737  000001  022565  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
3139  022374  112737  000006  022567          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
3140  022402  112737  000005  022564  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
3141  022410  010346                          MOV     R3,-(SP)        ;;SAVE R3
3142  022412  010446                          MOV     R4,-(SP)        ;;SAVE R4
3143  022414  010546                          MOV     R5,-(SP)        ;;SAVE R5
3144  022416  113704  022567                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
3145  022422  005404                          NEG     R4
3146  022424  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
3147  022430  110437  022566                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
3148  022434  113704  022565                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
3149  022440  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
3150  022444  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
3151  022446  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
3152  022450  000404                          BR      3$              ;;GO DO MSB
```

```
3153   022452   006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
3154   022454   006105                          ROL     R5
3155   022456   006105                          ROL     R5
3156   022460   010503                          MOV     R5,R3
3157   022462   006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
3158   022464   105337   022566                 DECB    $OMODE          ;;TYPE THIS DIGIT?
3159   022470   100016                          BPL     7$              ;;BR IF NO
3160   022472   042703   177770                 BIC     #177770,R3      ;;GET RID OF JUNK
3161   022476   001002                          BNE     4$              ;;TEST FOR 0
3162   022500   005704                          TST     R4              ;;SUPPRESS THIS 0?
3163   022502   001403                          BEQ     5$              ;;BR IF YES
3164   022504   005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
3165   022506   052703   000060                 BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
3166   022512   052703   000040         5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
3167   022516   110337   022562                 MOVB    R3,8$           ;;SAVE FOR TYPING
3168   022522   104401   022562                 TYPE    ,8$             ;;GO TYPE THIS DIGIT
3169   022526   105337   022564         7$:     DECB    $OCNT           ;;COUNT BY 1
3170   022532   003347                          BGT     2$              ;;BR IF MORE TO DO
3171   022534   002402                          BLT     6$              ;;BR IF DONE
3172   022536   005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
3173   022540   000744                          BR      2$              ;;GO DO THE LAST DIGIT
3174   022542   012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
3175   022544   012604                          MOV     (SP)+,R4        ;;RESTORE R4
3176   022546   012603                          MOV     (SP)+,R3        ;;RESTORE R3
3177   022550   016666   000002  000004         MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
3178   022556   012616                          MOV     (SP)+,(SP)
3179   022560   000002                          RTI                     ;;RETURN
3180   022562      000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
3181   022563      000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
3182   022564      000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
3183   022565      000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
3184   022566   000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
3185                                   .SBTTL  TYPE ROUTINE
3186
3187                                   ;;**************************************************************
3188                                   ;*ROUTINE TO tYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
3189                                   ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
3190                                   ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
3191                                   ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
3192                                   ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
3193                                   ;*
3194                                   ;*CALL:
3195                                   ;*1) USING A TRAP INSTRUCTION
3196                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
3197                                   ;*OR
3198                                   ;*      TYPE
3199                                   ;*      MESADR
3200                                   ;*
3201
3202   022570   105737   001157         $TYpE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
3203   022574   100002                          BPL     1$              ;;BR IF YES
3204   022576   000000                          HALT                    ;;HALT HERE IF NO TERMINAL
3205   022600   000430                          BR      3$              ;;LEAVE
3206   022602   010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
```

```
3207  022604  017600  000002            MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
3208  022610  122737  000001  001214    CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
3209  022616  001011                    BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
3210  022620  132737  000100  001215    BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
3211  022626  001405                    BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
3212  022630  010037  022640            MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
3213  022634  004737  023736            JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
3214  022640  000000            61$:    .WORD   0               ;;MESSAGE ADDRESS
3215  022642  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
3216  022650  001003                    BNE     60$             ;;YES,SKIP TYPE OUT
3217  022652  112046                2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
3218  022654  001005                    BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
3219  022656  005726                    TST     (SP)+           ;;IF TERmINATOR POP IT OFF THE STACK
3220  022660  012600                60$:    MOV     (SP)+,R0        ;;RESTORE R0
3221  022662  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
3222  022666  000002                        RTI                     ;;RETURN
3223  022670  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
3224  022674  001430                    BEQ     8$
3225  022676  122716  000200            CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
3226  022702  001006                    BNE     5$
3227  022704  005726                    TST     (SP)+           ;;POP  <CR><LF> EQUIV
3228  022706  104401                    TYPE                    ;;TYPE A CR AND LF
3229  022710  001171                    $CRLF
3230  022712  105037  023046            CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
3231  022716  000755                    BR      2$              ;;GET NEXT CHARACTER
3232  022720  004737  023002        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
3233  022724  123726  001156        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
3234  022730  001350                    BNE     2$              ;;IF NO GO GET NEXT CHAR.
3235  022732  013746  001154            MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
3236                                                            ;;AND THE NULL CHAR.
3237  022736  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
3238  022742  002770                    BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
3239  022744  004737  023002            JSR     PC,$TYPEC       ;;GO TYPE A NULL
3240  022750  105337  023046            DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
3241  022754  000770                    BR      7$              ;;LOOP
3242
3243                                ;HORIZONTAL TAB PROCESSOR
3244
3245  022756  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
3246  022762  004737  023002        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
3247  022766  132737  000007  023046    BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
3248  022774  001372                    BNE     9$              ;;TAB STOP
3249  022776  005726                    TST     (SP)+           ;;POP SPACE OFF STACK
3250  023000  000724                    BR      2$              ;;GET NEXT CHARACTER
3251  023002  105777  156142    $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
3252  023006  100375                    BPL     $TYPEC
3253  023010  116677  000002  156134    MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3254  023016  122766  000015  000002    CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
3255  023024  001003                    BNE     1$              ;;BRANCH IF NO
3256  023026  105037  023046            CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
3257  023032  000406                    BR      $TYPEX          ;;EXIT
3258  023034  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
3259  023042  001402                    BEQ     $TYPEX          ;;BRANCH IF YES
3260  023044  105227                    INCB    (PC)+           ;;COUNT THE CHARACTER
```

```
3261  023046  000000                    $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
3262  023050  000207                    $TYpEX: RTS     PC
3263
3264                                    .SBTTL  TTY INPUT ROUTINE
3265
3266                                    ;;**********************************************************************
3267                                    .ENABL  LSB
3268
3269                                    ;;**********************************************************************
3270                                    ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
3271                                    ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
3272                                    ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
3273                                    ;*WHEN OPERATING IN TTY FLAG MODE.
3274  023052  022737  000176  001140    $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
3275  023060  001074                            BNE     15$             ;;BRANCH IF NO
3276  023062  105777  156056                    TSTB    @$TKS           ;;CHAR THERE?
3277  023066  100071                            BPL     15$             ;;IF NO, DON'T WAIT AROUND
3278  023070  117746  156052                    MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
3279  023074  042716  177600                    BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
3280  023100  022726  000007                    CMP     #7,(SP)+        ;;IS IT A CONTROL G?
3281  023104  001062                            BNE     15$             ;;NO, RETURN TO USER
3282  023106  123727  001134  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
3283  023114  001456                            BEQ     15$             ;;BRANCH IF YES
3284
3285  023116  104401  023577                    TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
3286  023122  104401  023604            $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
3287  023126  013746  000176                    MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
3288  023132  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3289  023134  104401  023615                    TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
3290  023140  005046                    19$:    CLR     -(SP)           ;;CLEAR COUNTER
3291  023142  005046                            CLR     -(SP)           ;;THE NEW SWR
3292  023144  105777  155774            7$:     TSTB    @$TKS           ;;CHAR THERE?
3293  023150  100375                            BPL     7$              ;;IF NOT TRY AGAIN
3294
3295  023152  117746  155770                    MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
3296  023156  042716  177600                    BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
3297
3298
3299
3300  023162  021627  000025            9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
3301  023166  001005                            BNE     10$             ;;BRANCH IF NOT
3302  023170  104401  023572                    TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
3303  023174  062706  000006            20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
3304  023200  000757                            BR      19$             ;;LET'S TRY IT AGAIN
3305
3306
3307  023202  021627  000015            10$:    CMP     (SP),#15        ;;IS IT A <CR>?
3308  023206  001022                            BNE     16$             ;;BRANCH IF NO
3309  023210  005766  000004                    TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
3310  023214  001403                            BEQ     11$             ;;BRANCH IF YES
3311  023216  016677  000002  155714          MOV     2(SP),@SWR      ;;SAVE NEW SWR
3312  023224  062706  000006            11$:    ADD     #6,SP           ;;CLEAR UP STACK
3313  023230  104401  001171            14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
3314  023234  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
```

```
3315  023242  001003                        BNE     15$             ;;BRANCH IF NOT
3316  023244  012777  000100  155672        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
3317  023252  000002                15$:    RTI                     ;;RETURN
3318  023254  004737  023002        16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
3319  023260  021627  000060                CMP     (SP),#60        ;;CHAR < 0?
3320  023264  002420                        BLT     18$             ;;BRANCH IF YES
3321  023266  021627  000067                CMP     (SP),#67        ;;CHAR > 7?
3322  023272  003015                        BGT     18$             ;;BRANCH IF YES
3323  023274  042726  000060                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
3324  023300  005766  000002                TST     2(SP)           ;;IS THIS THE FIRST CHAR
3325  023304  001403                        BEQ     17$             ;;BRANCH IF YES
3326  023306  006316                        ASL     (SP)            ;;NO, SHIFT PRESENT
3327  023310  006316                        ASL     (SP)            ;;   CHAR OVER TO MAKE
3328  023312  006316                        ASL     (SP)            ;;   ROOM FOR NEW ONE.
3329  023314  005266  000002        17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
3330  023320  056616  177776                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
3331  023324  000707                        BR      7$              ;;GET THE NEXT ONE
3332  023326  104401  001170        18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
3333  023332  000720                        BR      20$             ;;SIMULATE CONTROL-U
3334                                .DSABL  LSB
3335
3336
3337                                ;;**************************************************************
3338                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
3339                                ;*CALL:
3340                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
3341                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
3342                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
3343                                ;
3344
3345  023334  011646        $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
3346  023336  016666  000004  000002        MOV     4(SP),2(SP)     ;;SAVE THE PS
3347  023344  105777  155574        1$:     TSTB    @$TKS           ;;WAIT FOR
3348  023350  100375                        BPL     1$              ;;A CHARACTER
3349  023352  117766  155570  000004        MOVB    @$TKB,4(SP)     ;;READ THE TTY
3350  023360  042766  177600  000004        BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
3351  023366  026627  000004  000023        CMP     4(SP),#23       ;;IS IT A CONTROL-S?
3352  023374  001013                        BNE     3$              ;;BRANCH IF NO
3353  023376  105777  155542        2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
3354  023402  100375                        BPL     2$              ;;LOOP UNTIL ITS THERE
3355  023404  117746  155536                MOVB    @$TKB,-(SP)     ;;GET CHARACTER
3356  023410  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
3357  023414  022627  000021                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
3358  023420  001366                        BNE     2$              ;;IF NOT DISCARD IT
3359  023422  000750                        BR      1$              ;;YES, RESUME
3360  023424  026627  000004  000140  3$:   CMP     4(SP),#140      ;;IS IT UPPER CASE?
3361  023432  002407                        BLT     4$              ;;BRANCH IF YES
3362  023434  026627  000004  000175        CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
3363  023442  003003                        BGT     4$              ;;BRANCH IF YES
3364  023444  042766  000040  000004        BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
3365  023452  000002                4$:     RTI                     ;;GO BACK TO uSER
3366                                ;;**************************************************************
3367                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
3368                                ;*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

023454  010346                  $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
023456  012703  023562            1$:   MOV     #$TTYIN,R3              ;;GET ADDRESS
023462  022703  023572            2$:   CMP     #$TTYIN+8.,R3           ;;BUFFER FULL?
023466  101405                          BLOS    4$                      ;;BR IF YES
023470  104410                          RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
023472  112613                          MOVB    (SP)+,(R3)              ;;GET CHARACTER
023474  122713  000177           10$:   CMPB    #177,(R3)               ;;IS IT A RUBOUT
023500  001003                          BNE     3$                      ;;SKIP IF NOT
023502  104401  001170            4$:   TYPE    $QUES                   ;;TYPE A '?'
023506  000763                          BR      1$                      ;;CLEAR THE BUFFER AND LOOP
023510  111337  023560            3$:   MOVB    (R3),9$                 ;;ECHO THE CHARACTER
023514  104401  023560                  TYPE    9$
023520  122723  000015                  CMPB    #15,(R3)+               ;;CHECK FOR RETURN
023524  001356                          BNE     2$                      ;;LOOP IF NOT RETURN
023526  105063  177777                  CLRB    -1(R3)                  ;;CLEAR RETURN (THE 15)
023532  104401  001172                  TYPE    $LF                     ;;TYPE A LINE FEED
023536  012603                          MOV     (SP)+,R3                ;;RESTORE R3
023540  011646                          MOV     (SP),-(SP)              ;;ADJUST THE STACK AND PUT ADDRESS OF THE
023542  016666  000004  000002          MOV     4(SP),2(SP)             ;;      FIRST ASCII CHARACTER ON IT
023550  012766  023562  000004          MOV     #$TTYIN,4(SP)
023556  000002                          RTI                             ;;RETURN
023560     000                  9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
023561     000                          .BYTE   0                       ;;TERMINATOR
023562  000010               $TTYIN:    .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
023572  052536  005015  000  $CNTLU:    .ASCIZ  /^U/<15><12>            ;;CONTROL "U"
023577     136  006507  000012 $CNTLG:  .ASCIZ  /^G/<15><12>            ;;CONTROL "G"
023604  005015  053523  020122 $MSWR:   .ASCIZ  <15><12>/SWR = /
023612  020075     000
023615     040  047040  053505 $MNEW:   .ASCIZ  /  NEW = /
023622  036440  000040

                                        .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

                                        ;;*****************************************************************
                                        ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                        ;*CHANGE IT TO BINARY.
                                        ;*CALL:
                                        ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                        ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                        ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

023626  011646                  $RDOCT: MOV     (SP),-(SP)              ;;PROVIDE SPACE FOR THE
023630  016666  000004  000002          MOV     4(SP),2(SP)             ;;INPUT NUMBER
023636  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
023640  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
023642  010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
023644  104411                    1$:   RDLIN                           ;;READ AN ASCIZ LINE
023646  012600                          MOV     (SP)+,R0                ;;GET ADDRESS OF 1ST CHARACTER
023650  005001                          CLR     R1                      ;;CLEAR DATA WORD
023652  005002                          CLR     R2
023654  112046                    2$:   MOVB    (R0)+,-(SP)             ;;PICKUP THIS CHARACTER
```

```
3423  023656  001412                          BEQ     3$              ;;IF ZERO GET OUT
3424  023660  006301                          ASL     R1              ;;*2
3425  023662  006102                          ROL     R2
3426  023664  006301                          ASL     R1              ;;*4
3427  023666  006102                          ROL     R2
3428  023670  006301                          ASL     R1              ;;*8
3429  023672  006102                          ROL     R2
3430  023674  042716  177770                 BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
3431  023700  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
3432  023702  000764                          BR      2$              ;;LOOP
3433  023704  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
3434  023706  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
3435  023712  010237  023726                  MOV     R2,$HIOCT
3436  023716  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
3437  023720  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
3438  023722  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
3439  023724  000002                          RTI                     ;;RETURN
3440  023726  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
3441                                  .SBTTL  APT COMMUNICATIONS ROUTINE
3442
3443                                  ;;**************************************************************************
3444  023730  112737  000001  024174  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
3445  023736  112737  000001  024172  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
3446  023744  000403                          BR      $ATYC
3447  023746  112737  000001  024174  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
3448  023754                          $ATYC:
3449  023754  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3450  023756  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3451  023760  105737  024172                 TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
3452  023764  001450                          BEQ     5$              ;;IF NOT:  BR
3453  023766  122737  000001  001214         CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
3454  023774  001031                          BNE     3$              ;;IF NOT:  BR
3455  023776  132737  000100  001215         BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
3456  024004  001425                          BEQ     3$              ;;IF NOT:  BR
3457  024006  017600  000004                 MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
3458  024012  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
3459  024020  005737  001174          1$:    TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
3460  024024  001375                          BNE     1$              ;;IF NOT:  WAIT
3461  024026  010037  001210                 MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
3462  024032  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
3463  024034  001376                          BNE     2$
3464  024036  163700  001210                 SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
3465  024042  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
3466  024044  010037  001212                 MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
3467  024050  012737  000004  001174         MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
3468  024056  000413                          BR      5$
3469  024060  017637  000004  024104  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
3470  024066  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
3471  024074  013746  177776                 MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
3472  024100  004737  022570                 JSR     PC,$TYPE        ;;CALL TYPE MACRO
3473  024104  000000                  4$:     .WORD   0
3474  024106                          5$:
3475  024106  105737  024174         10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
3476  024112  001416                          BEQ     12$             ;;IF NOT:  BR
```

```
3477  024114  005737  001214                    TST     $ENV            ;;RUNNING UNDER APT?
3478  024120  001413                            BEQ     12$             ;;IF NOT:  BR
3479  024122  005737  001174            11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
3480  024126  001375                            BNE     11$             ;;IF NOT:  WAIT
3481  024130  017637  000004  001176            MOV     @4(SP),$FATAL   ;;GET ERROR #
3482  024136  062766  000002  000004            ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
3483  024144  005237  001174                    INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
3484  024150  105037  024174            12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
3485  024154  105037  024173                    CLRB    $LFLG           ;;CLEAR LOG FLAG
3486  024160  105037  024172                    CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
3487  024164  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
3488  024166  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
3489  024170  000207                            RTS     PC              ;;RETURN
3490  024172     000          $MFLG:  .BYTE   0       ;;MESSG. FLAG
3491  024173     000          $LFLG:  .BYTE   0       ;;LOG FLAG
3492  024174     000          $FFLG:  .BYTE   0       ;;FATAL FLAG
3493        024176                     .EVEN
3494        000200                     APTSIZE=200
3495        000001                     APTENV=001
3496        000100                     APTSPOOL=100
3497        000040                     APTCSUP=040
3498
3499                                   .SBTTL  TRAP DECODER
3500
3501                                   ;;**************************************************************
3502                                   ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
3503                                   ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3504                                   ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
3505                                   ;*GO TO THAT ROUTINE.
3506
3507  024176  010046          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
3508  024200  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
3509  024204  005740                  TST     -(R0)           ;;BACKUP BY 2
3510  024206  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
3511  024210  006300                  ASL     R0              ;;POSITION FOR INDEXING
3512  024212  016000  024232          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
3513  024216  000200                  RTS     R0              ;;GO TO ROUTINE
3514
3515
3516                                   ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
3517
3518  024220  011646          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWn
3519  024222  016666  000004  000002  MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
3520  024230  000002                  RTI                     ;;RESTORE THE PSW
3521
3522                                   .SBTTL  TRAP TABLE
3523
3524                                   ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3525                                   ;*BY THE "TRAP" INSTRUCTION.
3526
3527                                   ;       ROUTINE
3528                                   ;       -------
3529  024232  024220          $TRPAD: .WORD   $TRAP2
3530  024234  022570                  $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
```

```
3531   024236  022366                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3532   024240  022342                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
3533   024242  022402                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
3534   024244  021176                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
3535
3536   024246  023122                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
3537
3538   024250  023052                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
3539   024252  023334                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
3540   024254  023454                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
3541   024256  023626                          $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
3542
3543   024260  000074                  BUF2:   .BLKW   60.
3544   024450  000074                  BUF3:   .BLKW   60.
3545   024640  000074                  BUFFER: 60.
3546
3547   024642  000240                          NOP
3548
3549                                           ;*
3550                                           ;*THIS SUB CODE IS USED TO INITIALIZE THE LPA-11
3551                                           ;*FIRST WE WILL LOAD MICROCODE INTO KMC-11
3552                                           ;*NEXT WE WILL INIT BOTH UPROCESSORS
3553                                           ;*THEN WE WILL LOAD DEVICE TABLE IN SLAVE UP.
3554                                           ;*THE ORDER OF LOAD IS DETERMINED BY THE USER.
3555                                           ;*
3556                                           ;*      CALL=   JSR     R5,$LPAI
3557                                           ;*              .WORD   0               ;ADDR. OF DEVICE ADDRESS.
3558                                           ;* ROUTINES REQUIRED:   .LOADLP
3559                                           ;* PROGRAMS REQUIRED:   DRLPX2
3560                                           ;*
3561
3562                                           ;*              ;RETURNS WITH $AERR=1 IF SLAVE
3563                                           ;*              ;MICRO SAYS AN ADDR. DOES NOT EXSIST. IN THE LIST.
3564                                           ;*
3565   024644                          $LPAI:
3566   024644  013746  000004                  MOV     4,-(SP)
3567
3568   024650  000413                          BR      31$             ;FIELD DOES NOT HAVE A BUS SWITCH TO
3569                                                                   ;WORRY ABOUT,SO WE WILL UNCONDITIONALLY
3570                                                                   ;BRANCH ARROUD THE NEXT CODE THAT
3571                                                                   ;WORKS BASED ON A BUS SWITCH.
3572                                                                   ;CODE LEFT IN HERE FOR IN HOUSE
3573                                                                   ;PERSONAL WHO MAY PATCH THIS BRANCH
3574                                                                   ;INSTRUCTION TO A <NOP> OCTAL <240>
3575                                                                   ;IN ORDER TO rUN PROGRAM WITH A SWITCH.
3576
3577                                                                   ;NOTE THIS "SWITCH" IS A PIECE OF INHOUSE
3578                                                                   ;TEST EQUIPMENT ONLY IT CONNECTS
3579                                                                   ;THE UNIBUS TO THE I/O BUS FOR
3580                                                                   ;CERTAIN TESTING.
3581   024652  012737  024676  000004            MOV     #30$,4
3582   024660  005237  170000                    INC     170000
3583   024664  104401  024672                    TYPE    ,65$            ;;TYPE ASCIZ STRING
3584   024670  000401                          BR      64$             ;;GET OVER THE ASCIZ
```

```
3585                                        ;;65$:  .ASCIZ  <7>##
3586  024674                                64$:
3587  024674  000401                              BR      31$
3588  024676  022626                        30$:  CMP     (SP)+,(SP)+
3589  024700  012637  000004                31$:  MOV     (SP)+,4           ;ALL THIS JUNK MUST BE REMOVED!!
3590  024704  005037  025522                      CLR     $AERR
3591  024710  004537  025524                      JSR     R5,$LOAD          ;LOAD MICRO-CODE.
3592  024714  000000G                             .WORD   DRLPX2            ;FILE "DRLPX2.OBJ"
3593
3594  024716  052777  040000  154512              BIS     #BIT14,@KMADO     ;ISSUE KMC+DMC INIT.
3595
3596  024724                                1$:
3597                                                                        ;"HANGS" HERE THEN KMC-11 ERROR.
3598  024724  010146                              MOV     R1,-(SP)
3599  024726  005001                              CLR     R1
3600  024730  005201                        2$:   INC     R1                ;STALL FOR DMC-UP
3601  024732  001376                              BNE     2$
3602  024734  012777  104000  154474              MOV     #BIT15!BIT11,@KMADO     ;SET RUN, AND ENABLE ARBITRATION.
3603  024742  105201                        25$:  INCB    R1
3604  024744  001376                              BNE     25$
3605
3606  024746  032777  000040  154462              BIT     #BIT5,@KMADO      ;SLAVE READY? (READING IPBM SR)
3607  024754  001401                              BEQ     3$
3608                                                                        ;FATAL LPA-11 ERROR SLAVE NOT READY.
3609  024756  104000                              ERROR
3610
3611  024760  012777  000004  154454  3$:   MOV     #4,@KMAD2         ;READ FAST PATH
3612  024766                          4$:
3613  024766  004537  026434                      JSR     R5, $TOUT         ;-TOUT-CHECK FOR TIMEOUT
3614
3615  024772  104000                              ERROR                     ;/TIME-OUT ERROR
3616                                                                        ;/WE FAILED TO COMPLETE
3617                                                                        ;/CURRENT OPERATION.
3618                                                                        ;/CONTINUES IN THIS LOOP
3619                                                                        ;/WOULD MAKE US "HANG" HERE
3620
3621  024774  000774                              BR              4$
3622
3623                                                                        ;/RETURNS HERE-FROM-TIMED OUT.
3624  024776  122777  000377  154436              CMPB    #377,@KMAD2       ;WAIT TILL KMC DONE COMMAND.
3625  025004  001370                              BNE     4$
3626  025006  122777  000377  154432              CMPB    #377,@KMAD4       ;IF FAST PATH=377 THEN ERROR.
3627  025014  001001                              BNE     35$
3628  025016  104000                              ERROR                     ;IPBM ERROR (SLAVE SIDE)
3629                                                                        ;YOU MUST RUN IPBM DIAGNOSTIC.
3630
3631  025020  122777  000004  154420  35$:  CMPB    #4,@KMAD4         ;IS THIS THE CORRECT VERSION OF MICRO-CODE?
3632  025026  001543                              BEQ     5$                ;YES-CONTINUE.
3633  025030  005227  177777                      INC     #-1
3634  025034  001140                              BNE     5$
3635  025036  005227  177777                      INC     #-1
3636  025042  001135                              BNE     5$
3637  025044  104401  025052                      TYPE    ,67$              ;;TYPE ASCIZ STRING
3638  025050  000440                              BR      66$               ;;GET OVER THE ASCIZ
```

```
                                        ;;67$:  .ASCIZ  <200>"W A R N I N G  THIS PROGRAM WAS DESiGNED TO RUN WITH VERSION 4"
025152                                  66$:
025152  104401  025160                        TYPE    ,69$            ;;TYPE ASCIZ STRING
025156  000430                              BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  <200>"MICRO-CODE. ANOTHER VERSION CODE WAS DETECTED."
025240                                  68$:
025240  104401  025246                        TYPE    ,71$            ;;TYPE ASCIZ STRING
025244  000434                              BR      70$             ;;GET OVER THE ASCIZ
                                        ;;71$:  .ASCIZ  <200>"THIS MAY OR MAYNOT CAUSE FALSE ERROR TO BE REPORTED."<200><200>
025336                                  70$:

025336  112737  177777  025470        5$:     MOVB    #0-1,11$        ;DAC CODE FOR SLAVE.
025344  012501                        6$:     MOV     (5)+,R1         ;GET NEXT DEVICE ADDR.
025346  021127  000000                        CMP     (R1),#0         ;TERM REACHED?
025352  001444                              BEQ     10$
025354  105237  025470                        INCB    11$
025360  113777  025470  154060              MOVB    11$,@KMAD4      ;FIFO DATA
025366  004737  025472                        JSR     PC,20$          ;ISSUE SEND
025372  112177  154050                        MOVB    (R1)+,@KMAD4    ;SEND LOW BYTE OF DEVICE ADDR TO SLAVE.
025376  004737  025472                        JSR     PC,20$          ;ISSUE SEND
025402  112177  154040                        MOVB    (R1)+,@KMAD4    ;SEND HIGH BYTE OF DEVICE ADDR. TO SLAVE.
025406  004737  025472                        JSR     PC,20$

025412  032777  000002  154016        7$:     BIT     #BIT1,@KMAD0    ;WAIT FOR FIFO DATA
025420  001374                              BNE     7$              ;=1 NO DATA. =0 DATA.
025422  112777  000002  154012              MOVB    #2,@KMAD2       ;READ FIFO.

025430                                  8$:
025430  004537  026434                        JSR     R5, $TOUt       ;-TOUT-CHECK FOR TIMEOUT

025434  104000                              ERROR                   ;/TIME-OUT ERROR
                                                                    ;/WE FAILED TO COMPLETE
                                                                    ;/CURRENT OPERATION.
                                                                    ;/CONTINUES IN THIS LOOP
                                                                    ;/WOULD MAKE US "HANG" HERE

025436  000774                              BR              8$

                                                                    ;/RETURNS HERE-FROM-TIMED OUT.
025440  122777  000377  153774              CMPB    #377,@KMAD2     ;WAIT FOR READ.
025446  001370                              BNE     8$
025450  105777  153772                        TSTB    @KMAD4          ;WAS A ZERO RETURNED?
025454  001734                              BEQ     6$              ;YES GET NEXT ADDR.
                                                                    ;SLAVE WILL RETURN CODE 0 IF
025456  005237  025522                        INC     $AERR           ;DEV PRESENT.   ELSE
                                                                    ;EXIT $AERR=1 IF SLAVE GIVES ERROR.
025462  005041                              CLR     -(1)            ;GET RID OF REFERENCE TO BAD ADDR.
025464  012601                        10$:    MOV     (SP)+,R1
025466  000205                              RTS     R5              ;RETURN ALL ADDR. CHECKED.

025470  000000                        11$:    .WORD   0               ;HOLDS DAC CODE PLUS OFFSET
                                                                    ;TO SLAVES ADDR. TABLE.

025472  112777  000003  153742        20$:    MOVB    #3,@KMAD2       ;ISSUE FIFO WRITE
```

```
3693  025500                          21$:
3694  025500  004537  026434                  JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
3695
3696  025504  104000                          ERROR                   ;/TIME-OUT ERROR
3697                                                                  ;/WE FAILED TO COMPLETE
3698                                                                  ;/CURRENT OPERATION.
3699                                                                  ;/CONTINUES IN THIS LOOP
3700                                                                  ;/WOULD MAKE US "HANG" HERE
3701
3702  025506  000774                          BR              21$
3703
3704                                                                  ;/RETURNS HERE-FROM-TIMED OUT.
3705  025510  122777  000377  153724          CMPB    #377,@KMAD2     ;KMC CODE WILL RETURN A "377"
3706  025516  001370                          BNE     21$             ;WHEN DONE COMMAND.
3707  025520  000207                          RTS     PC
3708
3709  025522  000000                  $AERR:  .WORD   0               ;=0 IF ADDR. LIST OK,=1 IF BAD.
3710
3711                                          ;*
3712                                          ;*THIS SUB CODE USED TO LOAD MICRO-CODE INTO LPA-11.
3713                                          ;*      CALL =  JSR     R5,$LOAD
3714                                          ;*              .WORD   XX              ;ADDR. OF MICRO CODE.
3715                                          ;*              ;RETURNS HERE
3716                                          ;*      NOTE:   MICRO CODE FILE MUST END IN -1 DATA.
3717                                          ;*
3718
3719  025524  010446                  $LOAD:  MOV     R4,-(SP)        ;SAVE R4.
3720  025526  010046                          MOV     R0,-(SP)        ;SAVE R0.
3721  025530  012500                  1$:     MOV     (5)+,R0         ;GET PROG. ADDR.
3722  025532  005077  153700                  CLR     @KMADO          ;CLEAR CSR
3723  025536  005077  153704                  CLR     @KMAD4          ;CLEAR CRAM ADDR.
3724  025542  052777  002000  153666  2$:     BIS     #2000,@KMADO    ;SELECT CRAM.
3725  025550  012077  153676                  MOV     (0)+,@KMAD6     ;WRITE DATA.
3726  025554  052777  020000  153654          BIS     #20000,@KMADO   ;SET CRAM WRITE
3727  025562  005077  153650                  CLR     @KMADO          ;DISABLE CRAM.
3728  025566  005277  153654                  INC     @KMAD4          ;UPDATE CRAM ADDR.
3729  025572  021027  177777                  CMP     (0),#-1         ;ALL DONE?
3730  025576  001361                          BNE     2$              ;NO LOOP.
3731  025600  005077  153642                  CLR     @KMAD4          ;CLEAR CRAM ADDR.
3732  025604  016500  177776                  MOV     -2(5),R0        ;GET MICRO CODE ADDR.
3733
3734  025610  052777  002000  153620  3$:     BIS     #2000,@KMADO    ;SELECT CRAM
3735  025616  022077  153630                  CMP     (R0)+,@KMAD6    ;DATA OK?
3736  025622  001013                          BNE     5$              ;NO - REPORT AN ERROR.
3737  025624  021027  177777                  CMP     (0),#-1         ;ALL DONe?
3738  025630  001405                          BEQ     4$              ;YES - EXIT
3739  025632  005077  153600                  CLR     @KMADO          ;NO - DESeLECT CRAM.
3740  025636  005277  153604                  INC     @KMAD4          ;UPDATE CRAM ADDR.
3741  025642  000762                          BR      3$
3742
3743  025644  012600                  4$:     MOV     (SP)+,R0        ;RESTORE R0
3744  025646  012604                          MOV     (SP)+,R4        ;RESTORE R4
3745  025650  000205                          RTS     R5              ;EXIT
3746
```

```
3747  025652                          5$:                     ;COME HERE ON LOAD ERROR
3748  025652  005745                  TST     -(5)
3749  025654  105204                  INCB    R4              ;UPDATE ERROR COUNTER.
3750  025656  100324                  BPL     1$              ;IF NOT TOO MANY, TRY AGAIN.
3751  025660  000000                  HALT                    ;MICRO CODE LOAD ERROR.
3752                                                          ;KMC-11 FAULT. YOU COULD TRY
3753  025662  000722                  BR      1$              ;TO PRESS CONTINUE TO GIVE IT
3754                                                          ;ANOTHER CHANCE, BUT I DOUBT
3755                                                          ;THAT THAT WOULD WORK. SINCE I'VE
3756                                                          ;ALREADY GIVEN IT 177 (OCTAL) CHANCES.
3757                                                          ;TRY RUNNING THE KMC-11 DIAGNOSTIC.
3758
3759
3760                                                  ;
3761                                                  ;*THIS ROUTINE ISSUES A WRITE COMMAND TO THE LPA-11
3762                                                  ;*
3763                                                  ;*      CALL =  JSR     R5,$TLKW
3764                                                  ;*              .WORD   0               ;OFFSET OF DEVICE ADDR.
3765                                                  ;*              .WORD   0               ;DATA TO BE WRITTEN
3766                                                  ;*
3767  025664  010046          $TLKW:  MOV     RO,-(SP)        ;SAVE RO
3768  025666  012500                  MOV     (5)+,RO         ;GET DEVICE OFFSET
3769  025670  052700  000340          BIS     #340,RO         ;ADD WRITE CODE.
3770  025674  004737  026146          JSR     PC,$LPW         ;WAIT FOR FAST PATH READY
3771  025700  010037  025772          MOV     RO,W1
3772  025704  010077  153536          MOV     RO,@KMAD4
3773  025710  112777  000005  153524  MOVB    #5,@KMAD2       ;ISSUE FAST PATH WRITE
3774  025716  004737  026146          JSR     PC,$LPW         ;WAIT FOR RDY
3775  025722  011537  025774          MOV     (5),W2
3776  025726  112577  153514          MOVB    (5)+,@KMAD4     ;WRITE LOW BYTE DATA.
3777
3778  025732  112777  000005  153502  MOVB    #5,@KMAD2       ;FP WRITE
3779  025740  004737  026146          JSR     PC,$LPW
3780  025744  111537  025776          MOVB    (5),W3
3781  025750  112577  153472          MOVB    (5)+,@KMAD4     ;WRITE HIGH BYTE
3782  025754  112777  000005  153460  MOVB    #5,@KMAD2
3783  025762  004737  026146          JSR     PC,$LPW
3784  025766  012600                  MOV     (SP)+,RO
3785  025770  000205                  RTS     R5              ;EXIT DONE.
3786  025772  000000          W1:     0
3787  025774  000000          W2:     0
3788  025776  000000          W3:     0
3789
3790                                                  ;*
3791                                                  ;*THIS ROUTINE ISSUES A READ COMMAND TO THE LPA-11
3792                                                  ;*
3793                                                  ;*      CALL =  JSR     R5,$TLKr
3794                                                  ;*              .WORD   0               ;OFFSET OF DEVICE
3795                                                  ;*              ;RETURNS HERE
3796                                                  ;*DATA IN WORD $DATR
3797                                                  ;*
3798
3799  026000  010046          $TLKR:  MOV     RO,-(SP)        ;SAVE RO
3800  026002  012500                  MOV     (5)+,RO         ;GET OFFSET
```

```
3801  026004  052700  000300          BIS     #300,R0         ;ADD READ CODE
3802  026010  004737  026146          JSR     PC,$LPW         ;WAIT TILL READY
3803  026014  110077  153426          MOVB    R0,@KMAD4
3804  026020  112777  000005  153414  MOVB    #5,@KMAD2       ;ISSUE WRITE FP
3805  026026  004737  026146          JSR     PC,$LPW
3806  026032  010037  026142          MOV     R0,RD1
3807  026036                    1$:
3808  026036  004537  026434          JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
3809
3810  026042  104000                  ERROR                   ;/TIME-OUT ERROR
3811                                                          ;/WE FAILED TO COMPLETE
3812                                                          ;/CURRENT OPERATION.
3813                                                          ;/CONTINUES IN THIS LOOP
3814                                                          ;/WOULD MAKE US "HANG" HERE
3815
3816  026044  000774                  BR              1$
3817
3818                                                          ;/RETURNS HERE-FROM-TIMED OUT.
3819  026046  032777  000040  153362  BIT     #BIT5,@KMAD0    ;FAST PATH GOT DATA?
3820  026054  001370                  BNE     1$
3821  026056  112777  000004  153356  MOVB    #4,@KMAD2       ;ISSUE FAST PATH READ
3822  026064  004737  026146          JSR     PC,$LPW
3823  026070  117737  153352  026144  MOVB    @KMAD4,$DATR    ;GET LOW BYTE
3824  026076                    2$:
3825  026076  004537  026434          JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
3826
3827  026102  104000                  ERROR                   ;/TIME-OUT ERROR
3828                                                          ;/WE FAILED TC COMPLETE
3829                                                          ;/CURRENT OPERATION.
3830                                                          ;/CONTINUES IN THIS LOOP
3831                                                          ;/WOULD MAKE US "HANG" HERE
3832
3833  026104  000774                  BR              2$
3834
3835                                                          ;/RETURNS HERE-FROM-TIMED OUT.
3836  026106  032777  000040  153322  BIT     #BIT5,@KMAD0    ;FAST PATH READY?
3837  026114  001370                  BNE     2$
3838  026116  112777  000004  153316  MOVB    #4,@KMAD2       ;ISSUE FAST PATH READ
3839  026124  004737  026146          JSR     PC,$LPW
3840  026130  117737  153312  026145  MOVB    @KMAD4,$DATR+1  ;SAVE HIGH BYTE
3841  026136  012600                  MOV     (SP)+,R0
3842  026140  000205                  RTS     R5
3843  026142  000000            RD1:  0
3844  026144  000000            $DATR:.WORD   0
3845                                  ;
3846                                  ;THIS ROUTINE WAITS FOR KMC-CODE TO BECOME READY AS WELL
3847                                  ;AS FAST PATH TO BE READ.
3848                                  ;
3849                                  ;       CALL =  JSR     PC,$LPW
3850                                  ;
3851                                  ;IT WILL TIME OUT IF TOO MUCH TIME IS TAKEN BY
3852                                  ;THE MICRO-PROCESSORS AND REPORT AN ERROR, THEN HALT.
3853                                  ;
3854
```

```
3855  026146  010146                  $LPW:   MOV     R1,-(SP)        ;SAVE R1
3856  026150  005001                          CLR     R1
3857  026152  122777  000377  153262  1$:     CMPB    #377,@KMAD2     ;FINISHED INSTRUCTION?
3858  026160  001403                          BEQ     2$
3859  026162  005201                          INC     R1              ;TIME OUT?
3860  026164  001372                          BNE     1$
3861  026166  000411                          BR      10$
3862
3863  026170  032777  000020  153240  2$:     BIT     #BIT4,@KMAD0    ;FAST PATH READ?
3864  026176  001403                          BEQ     3$
3865  026200  005201                          INC     R1              ;NO - TIME OUT?
3866  026202  001372                          BNE     2$
3867  026204  000402                          BR      10$             ;YES - REPORT AN ERROR
3868
3869  026206  012601                  3$:     MOV     (SP)+,R1        ;RESTORE R1
3870  026210  000207                          RTS     PC              ;EXIT
3871
3872  026212                          10$:
3873  026212  104401  026220                  TYPE    ,65$            ;;TYPE ASCIZ STRING
3874  026216  000407                          BR      64$             ;;GET OVER THE ASCIZ
3875                                  ;;65$:   .ASCIZ  <200>#LPA-11 FAULT#
3876  026236                          64$:
3877
3878  026236  000000                  11$:    HALT                    ;LPA-11 FAULT RUN LPA-11
3879  026240  000776                          BR      11$             ;DIAGNOSTICS.
3880
3881
3882
3883                                  ;*
3884                                  ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE TO
3885                                  ;*A DEVICE ADDRESS ON THE I/O BUSS FOR WRITE ONLY.
3886                                  ;*
3887                                  ;* FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN USED
3888                                  ;* BEFORE. IF NOT WE HAVE TO INITIALIZE THE LPA WITH
3889                                  ;* THAT ADDRESS.
3890                                  ;* WHEN THE ADDR. IS KNOWN BY THE LPA, DO THE OUTPUT BY
3891                                  ;* $TLKw
3892                                  ;*
3893
3894  026242  010046                  $OUTLP: MOV     R0,-(SP)        ;SAVE R0
3895  026244  010146                          MOV     R1,-(SP)        ;SAVE R1
3896
3897  026246  012700  001464                  MOV     #.DVLS,R0       ;PROGRAM DEFINED LIST.
3898  026252  005001                          CLR     R1
3899  026254  005710                  1$:     TST     (0)             ;TERMINATOR REACHED?
3900  026256  001421                          BEQ     10$             ;YES NEXT STEP.
3901  026260  027520  000000                  CMP     @(5),(0)+       ;MATCH WITH ADDR IN LIST?
3902  026264  001402                          BEQ     2$
3903  026266  005201                          INC     R1
3904  026270  000771                          BR      1$
3905
3906  026272  010137  026310          2$:     MOV     R1,3$           ;SAVE OFFSET, DEVICE KNOWN.
3907  026276  005725                          TST     (5)+
3908  026300  013537  026312                  MOV     @(5)+,4$        ;GET DATA TO BE WRITTEN
```

```
3909  026304  004537  025664            JSR     R5,$TLKw        ;DO WRITE
3910  026310  000000            3$:     .WORD   0               ;DEVICE OFFSET
3911  026312  000000            4$:     .WORD   0               ;DATA TO BE WRITTEN.
3912  026314  012601                    MOV     (SP)+,R1
3913  026316  012600                    MOV     (SP)+,R0
3914  026320  000205                    RTS     R5
3915  026322  017520  000000    10$:    MOV     @(5),(0)+       ;SAVE ADDR.
3916  026326  005010                    CLR     (0)
3917  026330  004537  024644            JSR     R5,$LPAI
3918  026334  001464                    .WORD   .DVLS
3919  026336  000755                    BR      2$
3920
3921                                    ;*
3922                                    ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE
3923                                    ;*TO A DEVICE ADDR. ON THE I/O BUSS FOR READ ONLY.
3924                                    ;*
3925                                    ;*FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN
3926                                    ;*USED BEFORE. IF NOT, WE HAVE TO INITIALIZE THE LPA
3927                                    ;*WITH THE NEW ADDR.
3928                                    ;*WHEN THE ADDR IS KNOWN WE CAN DO oUTPUT THROUGH
3929                                    ;*$TLKR
3930                                    ;*      CALL THROUGH    MOVEI   DATA,ADDR.
3931                                    ;*              WHICH EQUALS:
3932                                    ;*                      JSR     R5,$INLP
3933                                    ;*                      .WORD   XX      ADDR OF DEVICE
3934                                    ;*                      .WORD   YY      ADDR TO STORE READ DATA.
3935
3936  026340  010046    $INLP:  MOV     R0,-(SP)        ;SAVE R0
3937  026342  010146            MOV     R1,-(SP)        ;SAVE R1
3938
3939  026344  012700  001464            MOV     #.DVLS,R0       ;PROG DEFINED ADDR. LIST.
3940  026350  005001                    CLR     R1
3941  026352  005710    1$:     TST     (0)             ;EOL REACHED?
3942  026354  001420            BEQ     10$             ;YES - DEFINE NEW ADDR.
3943
3944  026356  027520  000000            CMP     @(5),(0)+       ;ADDR. MATCH?
3945  026362  001402                    BEQ     2$
3946  026364  005201                    INC     R1
3947  026366  000771                    BR      1$
3948
3949  026370  010137  026402    2$:     MOV     R1,3$           ;SAVE LIST OFFSET
3950  026374  005725                    TST     (5)+
3951  026376  004537  026000            JSR     R5,$TLKr        ;GO READ DEVICE
3952          026402            $OFS=.
3953  026402  000000            3$:     .WORD   0               ;OFFSET OF DEVICE
3954
3955  026404  013735  026144            MOV     $DATR,@(5)+     ;STORE DATA.
3956  026410  012601                    MOV     (SP)+,R1        ;RESTORE R1
3957  026412  012600                    MOV     (SP)+,R0        ;RESTORE R2
3958  026414  000205                    RTS     R5              ;EXIT
3959
3960  026416  017520  000000    10$:    MOV     @(5),(0)+
3961  026422  005010                    CLR     (0)
3962  026424  004537  024644            JSR     R5,$LPAI
```

```
3963  026430  001464                          .WORD   .DVLS
3964  026432  000756                          BR      2$
3965                                          ;*
3966                                          ;*$TOUT ROUTINE USED TO WATCH IF
3967                                          ;*      WE'RE IN A LOOP TOO-LONG
3968                                          ;*      CALL=   JSR R5, $TOUT
3969                                          ;*              ERROR X     ;RETURNS HERE ON TIMEOUT
3970                                          ;*              BR
3971                                          ;*              ;RETURNS HERE NO ERROR
3972                                          ;*
3973
3974  026434  020537  026470          $TOUt:  CMP     R5,$SAD         ;SAME ADDR?
3975  026440  001405                          BEQ     1$
3976  026442  010537  026470                  MOV     R5,$SAD         ;NO-SAVE THIS ADDR.
3977  026446  005037  026472                  CLR     $CNT            ;CLR CNT AT ADDR.
3978  026452  000403                          BR      2$
3979  026454  005237  026472          1$:     INC     $CNT            ;OVERFLOW?
3980  026460  100402                          BMI     3$              ;YES-ERROR RETURN
3981  026462  062705  000004          2$:     ADD     #4,R5           ;NO-NON ERROR RETURN
3982  026466  000205                  3$:     RTS     R5              ;RETURN.
3983
3984  026470  000000                  $SAD:   .WORD   0               ;CONTAINS LOOP ADDR.
3985  026472  000000                  $CNT:   .WORD   0               ;# OF TIMES AT ADDR.
3986
3987                                          ;*
3988                                          ;* THIS ROUTINE REPLACES WHAT THE USER WOULD ORDINARILY
3989                                          ;*USE FOR A RESET.  FIRST,WE DO A RESET INSTRUCTION.
3990                                          ;*THEN WE CLR ".DVLST" WHICH FORCES US TO RESET BOTH THE
3991                                          ;*KMC AND DMC AS SOON AS A DEVICE IS REFERENCED.
3992                                          ;*
3993                                          ;*      CALL=JSR        PC,$RESET       ;REPLACES "RESET INSTRUCTION
3994                                          ;*              ;RETURNS HERE.
3995                                          ;*
3996  026474  000005                  $RESET: RESET                   ;RESET THE WORLD.
3997
3998                                  ;*      MOV     @2$,1$  ;/READ DEVICE REG 2$,PUT DATA IN 1$.
3999  026506  005737  025522                  TST     $AERR           ;IF NO ERROR,LOOP
4000  026512  001004                          BNE     10$             ;THERE WAS AN ERROR.
4001  026514  062737  000002  026530          ADD     #2,2$           ;UPDATE DEVICE ADDR.
4002                                                                  ;YOU SEE ,WE HAVE TO PROTECT OUR SELF!
4003                                                                  ;IF 2$ CONTAINED A VALID ADDR,WE
4004                                                                  ;MUST KEEP TRYING UNTIL WE GENERATE
4005                                                                  ;AN INVALID ADDR.
4006  026522  000764                          BR      $RESET
4007  026524                          10$:
4008  026524  000207                          RTS     PC
4009  026526  000000                  1$:     .WORD   0               ;JUNK LOC.
4010  026530  160000                  2$:     .WORD   160000          ;DUMB ADDR. FORCES INIT OF DMC/KMC.
4011
4012
4013                                          ;
4014                                          ;SDELAY- ROUTINE TO GIVE A MINOR DELAY.
4015                                          ;        IS NOT TIME DEP◆NDENT CODE SENCE
4016                                          ;        NOT USED TO GET SPECIFIC TIME BUT
```

```
                                        ;       JUST A LITTLE DELAY.
                                        ;
                                        ;       THAT IS UNLESS A REAL TIME CLOCK IS PRESENT!
                                        ;       THEN WE'LL GENERATE A TIME BETWEEN 16MS TO 32 MS
                                        ;
                                        ;
                                        ;       CALL=   JSR PC,  SDELAY
                                        ;
026532                          SDELAY: TST     RTCCSR          ;CLOCK PRESENT?
026532  005737  026614                  BPL     10$
026536  100016                          MOV     #2,TIME
026540  012737  000002  026604          BIS     #115,@RTCCSR    ;START CLOCK
026546  052777  000115  000040          CLR     PS
026554  005037  177776                  TST     TIME
026560  005737  026604          1$:     BNE     1$
026564  001375                          CLR     @RTCCSR         ;STOP CLOCK
026566  005077  000022
                                        RTS PC
026572  000207                  10$:    INCB    TIME
026574  105237  026604                  BNE     10$
026600  001375                          RTS     PC
026602  000207
                                TIME:   .WORD   0
026604  000000
                                CLKINT: DEC     TIME
026606  005337  026604                  RTI
026612  000002                  RTCCSR: .WORD   0               ;CLOCK CSR IF USED.
026614  000000                          ;
                                        ;*
                                        ;*THIS MACRO ALLOWS THE OPERATOR TO TALK TO
                                        ;*ANY DEVICE ON THE I/O BUS
                                        ;*USER MUST START AT THIS ADDR.
                                        ;*HE MUST SAY EITHER "E" FOR EXAMINE, OR "D" FOR DEPOSIT.
                                        ;*"E" IS DEFAULT.
                                        ;*NEXT, HE MUST SUPPLY AN ADDR.
                                        ;*NOTE IF ADDR. IS NOT FOUND ON I/O BUS, A HALT
                                        ;*WILL OCCUR.
                                        ;
026616                          $UTK:
026616  005037  001464                  CLR     .DVLS
026622                          21$:
026622  104401  026630                  TYPE    ,65$            ;;TYPE ASCIZ STRING
026626  000405                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>#E OR D?#
026642                          64$:
026642  105777  152276          1$:     TSTB    @$TKS
026646  100375                          BPL     1$
026650  117737  152272  026772          MOVB    @$TKB,20$       ;GET INPUT
026656  104401  026772                  TYPE,   20$             ;ECHO, NEXT MESSAGE.
026662  142737  000240  026772          BICB    #240,20$        ;STRIP PARITY, LC
026670  104412                          RDOCT                   ;GET ADDR.
026672  012637  026770                  MOV     (SP)+,14$
026676  123727  026772  000104          CMPB    20$,#'D         ;DEPOSIT?
```

```
                                        BEQ     10$

                                        JSR     R5,$INLP        ;GET DATA
                                2$:     .WORD   14$
                                        .WORD   5$

                                        MOV     5$,-(SP)        ;;SAVE 5$ FOR TYPEOUT
                                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                        BR      21$             ;LOOP.
                                5$:     .WORD   0

                                10$:
                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                        BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <200>#DATA= #
                                66$:
                                        RDOCT
                                        MOV     (SP)+,13$

                                11$:    JSR     R5,$OUTLP       ;OUTPUT ROUTINE.
                                12$:    .WORD   14$             ;DEVICE ADDR.
                                        .WORD   13$             ;DATA
                                        BR      21$

                                13$:    .WORD   0
                                14$:    .WORD   0
                                20$:    .ASCIZ  <1><200>#DEVICE ADDR= #

                                        .EVEN


                                ;
                                ;THIS ROUTINE LOOKS THROUGH CURENT .DVLS FOR A/D ADDR.
                                ;IF UNFOUND,GENERATES IT. THIS ROUTINE'S WHOLE PURPOSE IS
                                ;TO SET UP THE USER PROGRAM TO LINK TO FILE "DRLPX2" FOR
                                ;SAMPLE TAKEING PURPOSES.
                                ;       TO TAKE SAMPLES, THE USER PROGRAM MUST SET UP
                                ;       A/D CSR IN BSEL 4,AND 5.
                                ;       (2) HE MUST CALL THIS ROUTINE:
                                ;               JSR     R5,$PUTS        ;CALL SET UP ROUTINE.
                                ;               .WORD   ADCSR           ;ADDR. OF A/D CSR.
                                ;               ;RETURNS HERE ;KMC BSEL 3,6,7 PERMINENTLY SET UP
                                ;                               ;(UNTILL ONE DOES A RESET)
                                ;
                                ;       (3)THE USER MUST PUT CODE 006 INTO KMC REG 2 TO
                                ;          START CONVERSION  CAUTION*DO WITH MOVB INSTR.!
                                ;       (4)MONITOR KMC REG 2 FOR CODE 377 (DRLPX2 IS DONE)
                                ;       (5)READ KMC REG 4,5 FOR A/D RESULT.
                                ;       (6) TO TAKE MORE SAMPLES,SIMPLY PUT A/D CSR INTO
                                ;          BSEL 4,5 AND CODE 6 INTO BSEL 2.
                                ;
4124  027012  012537  027022    $PUTS:  MOV     (5)+,1$                 ;GET ADDR OF ADDR. OF A/D
```

```
4125   027016   004537   026340              JSR     R5,$INLP
4126   027022   000000            1$:       .WORD   0
4127   027024   027120                      .WORD   10$
4128   027026   113777   026402   152416     MOVB    $OFS,@KMAD6
4129   027034   113777   026402   152412     MOVB    $OFS,@KMAD7
4130   027042   013737   027022   027062     MOV     1$,2$
4131   027050   062737   000002   027062     ADD     #2,2$
4132   027056   004537   026340              JSR     R5,$INLP
4133   027062   000000            2$:       .WORD   0
4134   027064   027120                      .WORD   10$
4135   027066   113777   026402   152350     MOVB    $OFS,@KMAD3
4136   027074   152777   000340   152350     BISB    #340,@KMAD6
4137   027102   152777   000300   152344     BISB    #300,@KMAD7
4138   027110   152777   000300   152326     BISB    #300,@KMAD3
4139   027116   000205                      RTS     R5
4140   027120   000000            10$:      .WORD   0
4141
4142            000001                      .END
```

```
ABASE = 170416            163#   274    315
ACDW1 = 000000            274    317
ACDW2 = 000000            274
ACPUOP= 000000            274    289
ACTRLC  013672           1740   2121   2160#  2193   2211   2255   2292   2361   2931   2984
.D      002244            562#   582
ADDW0 = 000000            274
ADDW1 = 000000            274
ADDW10= 000000            274
ADDW11= 000000            274
ADDW12= 000000            274
ADDW13= 000000            274
ADDW14= 000000            274
ADDW15= 000000            274
ADDW2 = 000000            274
ADDW3 = 000000            274
ADDW4 = 000000            274
ADDW5 = 000000            274
ADDW6 = 000000            274
ADDW7 = 000000            274
ADDW8 = 000000            274
ADDW9 = 000000            274
ADEND   015412           1770   1783   1806   1823   1838   1852   1866   1909   1925   2424*  2438*  2443*  2444*
                         2445*  2446*  2447*  2448   2458#
ADEVCT= 000000            274    280
ADEVM = 000000            274    316
AENV  = 000000            274    285
AENVM = 000000            274    286
AFATAL= 000000            274    277
AL      002242            561#   580
AMADR1= 000000            274    302
AMADR2= 000000            274    306
AMADR3= 000000            274    309
AMADR4= 000000            274    312
AMAMS1= 000000            274    296
AMAMS2= 000000            274    304
AMAMS3= 000000            274    307
AMAMS4= 000000            274    310
AMSGAD= 000000            274    282
AMSGLG= 000000            274    283
AMSGTY= 000000            274    276
AMTYP1= 000000            274    297
AMTYP2= 000000            274    305
AMTYP3= 000000            274    308
AMTYP4= 000000            274    311
APASS = 000000            274    279
APRIOR= 000200            165#   274
APTCSU= 000040           3215   3497#
APTENV= 000001           2997   3208   3453   3495#
APTSIZ= 000200            507   3494#
APTSPO= 000100           3210   3455   3496#
ASWREG= 000000            274    287
ATESTN= 000000            274    278
AUNIT = 000000            274    281
```

```
AUSWR = 000000            274     288
AUTCAL  011456            596    1746#   2108
AUTOCL  020076           1749    2723#
AUTOER  020162           2113    2733#
AUTRAP  013460           1755    2110#
AVECT1= 100360            164#    274     313     446     447
AVECT2= 000000            274     314
BEG     001600            469     471#
BEGIN   001554            189     466#   3094
BEGIN1  001572            190     470#
BIT0  = 000001            147#   1056
BIT00 = 000001            137#    147
BIT01 = 000002            136#    146
BIT02 = 000004            135#    145
BIT03 = 000010            134#    144
BIT04 = 000020            133#    143
BIT05 = 000040            132#    142
BIT06 = 000100            131#    141
BIT07 = 000200            130#    140
BIT08 = 000400            129#    139    2946
BIT09 = 001000            128#    138
BIT1  = 000002            146#    904     907    1157    3662
BIT10 = 002000            127#   1025    1028    1134    1172    1817    1832    1860    1887    1890    1919    2237    2281
BIT11 = 004000            126#    701     743     786     830    1039    1042    1172    1832    1860    1903    2954    3602
BIT12 = 010000            125#   1130    1137    1817    1860    1903    1942    2242
BIT13 = 020000            124#   1761    1791    1994    2001    2015    2022    2054    2064    2078    2088    2129    2137
                         2992
BIT14 = 040000            123#   1546    1994    2015    2054    2078    2133    2137    2932    3594
BIT15 = 100000            122#    635    3602
BIT2  = 000004            145#    927     930    1072    1073    1084    1093    1157
BIT3  = 000010            144#    951     954    1103    1104    1114    1122    1157
BIT4  = 000020            143#    975     978    1157    1325    1342    1357    1371    1481    1651    1719    1874    1877
                         1890    3863
BIT5  = 000040            142#   2235    2247    3606    3819    3836
BIT6  = 000100            141#    997    1000    1172
BIT7  = 000200            140#    892     907     916     930     939     954     964     978     987    1000    1014    1028
                         1042    1059    1072    1093    1103    1122    1160    1175    1371    1890    1971    2323    2331
BIT8  = 000400            139#   2364
BIT9  = 001000            138#   1011    1014    1172    1860    1919    1954    1968
BPTVEC= 000014            154#
BRLEV1  021160            554*    555*    556    2387*   2388*   2841#
BRLEV2  021162            556*    557*   2842#
BUFFER  024640           3545#
BUF2    024260           3543#
BUF3    024450           3544#
CALDAC  014014            599    2190#
CALDON  017716           2107    2700#
CHANL   015416           2421*   2460#
CHRCOL  011260           1706#
CHTIME  015064           1451    1469    1518    1622    2387#
CH0   = 000000           1759#
CH1   = 000001           1759#
CH10  = 000010           1759#
CH11  = 000011           1759#
```

```
CH12   = 000012            1759#
CH13   = 000013            1759#
CH14   = 000014            1759#
CH15   = 000015            1759#
CH16   = 000016            1759#
CH17   = 000017            1759#
CH2    = 000002            1759#
CH3    = 000003            1759#   1769
CH4    = 000004            1759#   1782
CH40   = 000040            1759#
CH41   = 000041            1759#
CH42   = 000042            1759#
CH43   = 000043            1759#
CH44   = 000044            1759#
CH45   = 000045            1759#
CH46   = 000046            1759#
CH47   = 000047            1759#
CH5    = 000005            1759#
CH50   = 000050            1759#
CH51   = 000051            1759#   2032    2098    2405
CH52   = 000052            1759#   2037    2104
CH53   = 000053            1759#   1805    1822    1837
CH54   = 000054            1759#
CH55   = 000055            1759#   1851    1865
CH56   = 000056            1759#   1908    1924
CH57   = 000057            1759#
CH6    = 000006            1759#
CH7    = 000007            1759#
CKSWR  = 104407             575    2122    2192    2210    2234    2262    2291    2360    2930    2983    3007    3538#
CLKINT   026606            4042#
CLRVC    014214            2234#
COMPAR   015422            1772    1785    2408    2463#
CONTC    017525             602    2674#
CONVRT   015204            1768    1781    1804    1821    1836    1850    1864    1907    1923    2404    2420#
CPTYPE   015110            2387    2394#
CR     = 000015              62#   3254    3264
CRLF   = 000200              63#   3225    3264
CROSS    014704            1760    1790    1980    1993    2000    2014    2021    2053    2063    2077    2087    2344#
CSPACE   013734            1997    2006    2018    2027    2058    2069    2082    2092    2172#   2173    2184
CTRLC    002430             569     601#   2114    2166    2178
DACO     001540             459#    544     656     677     679     690     692     705     707     848     863    1088    1192
                           1196    1260    1264    1388*   1444    1463    1519    1623    1722    1993    2014    2031    2127
                           2198    2221    2279    2818
DAC1     001542             460#    659     719     721     732     734     747     749     851     870    1118    1209    1213
                           1277    1281    1389*   1445    1462    1520    1624    1724    2004    2025    2036    2131    2200
                           2223    2280    2820
DAC2     001544             461#    662     762     764     775     777     790     792     854     877    1226    1230    1294
                           1298    1390*   1448    1467    1523    1627    1726    2053    2077    2097    2135    2202    2225
                           2276    2822
DAC3     001546             462#    665     806     808     819     821     834     836     857     884    1243    1247    1311
                           1315    1391*   1449    1466    1524    1628    1728    2063    2067    2087    2103    2139    2204
                           2227    2277    2824
DDISP  = 177570              69#    253     495
DELAY    001532             456#   2147    2153
```

```
DFO     021122            341     347     353     359     365     371     377     383     389     395     401     407     413
                          419    2831#
DH1     017370            339     345     351     357     363     387    2655#
DH13    017440            399    2662#
DH14    017472            393     405     411     417    2667#
DH6     017424            369     375     381    2660#
DISPLA  001142            253#    495*    503*   2966*   2967*
DISPRE  000174            186#    503
DRLPX2= ******  G          14#    3592
DSWR   = 177570            68#    252     494
DT1     021006            340     388    2816#
DT13    021076            400    2827#
DT14    021110            394     406     412     418    2829#
DT2     021020            346    2818#
DT3     021032            352    2820#
DT4     021044            358    2822#
DT5     021056            364    2824#
DT6     021070            370     376     382    2826#
DYNCAL  014104            191    2208#
EMTVEC= 000030            157#    482*    483*
EM1     016547            338    2584#
EM10    017023            380    2615#
EM11    017074            386    2622#
EM12    017123            392    2626#
EM13    017166            398    2632#
EM14    017207            404    2635#
EM15    017256            410    2642#
EM16    017325            416    2649#
EM2     016600            344    2589#
EM3     016627            350    2593#
EM4     016656            356    2597#
EM5     016705            362    2601#
EM6     016734            368    2605#
EM7     016767            374    2610#
ERRVEC= 000004            150#    492     493*    504*    649*    669*   1755*   2111*   2937    2938*   2940*   2943*
ETX    = 000003           2806#   2810    2814
EVER    021164            467*    624     634*    635*    637    1385    2827    2843#
EVER1   021166            468*   1436    1442*   2844#
FCHANL  015414           2459#
FILZ    014530           2297#   2318    2323    2326    2331    2334    2340    2349
FOUND1  017533             627   2676#
FOUND2  017557             633   2681#
FULRMP  011264             598   1713#
FUNDAC  015112            2030    2035    2096    2102   2399#
GADCS   015404            2428    2431    2434   2455#
GADDBR  015406            2438    2441   2456#
GNS    = ******  U         185    1794    3530    3531    3532    3533    3534    3536    3538    3539    3540    3541    3585
                          3639    3643    3647    3875    4061    4085
GSTAT   015410            1954    1957    1968    1971   2457#
GTSWR  = 104406           3536#
HIGH    021144           2835#
HT     = 000011             60#   3223    3264
INCR    021146           2836#
INIT1   002254             529    567#
```

```
IOTVEC=  000020   155#   480*   481*
IV       001550   463#   542    550*   551*   552    1392*
IVS      001552   464#   552*   553*   1393*
KM3D0    001436   429#   517    3594*  3602*  3606   3662   3722*  3724*  3726*  3727*  3734*  3739*  3819
                  3836   3863
KMAD1    001440   432#   518
KM3D2    001442   434#   3611*  3624   3664*  3678   3692*  3705   3773*  3778*  3782*  3804*  3821*  3838*
                  3857
KMAD3    001444   436#   4135*  4138*
KM3D4    001446   438#   3626   3631   3655*  3657*  3659*  3680   3723*  3728*  3731*  3740*  3772*  3776*
                  3781*  3803*  3823   3840
KMAD5    001450   440#
KMAD6    001452   442#   3725*  3735   4128*  4136*
KMAD7    001454   444#   522    4129*  4137*
LDTRAP   002072   531#   567    641    1714   1747   2118   2191   2209
LF     =  000012   61#   3258   3264
LOADVC   014372   2269#
LOW      021142   2834#
LPADH    001450   439#
LPADL    001446   437#
LPCI     001436   428#
LPCO     001442   433#
LPMR     001440   431#
LPMS1    001452   441#
LPMS2    001454   443#
LPSO     001444   435#
LTEST    002614   643#   1395
MANHED   017755   2120   2706#
MANUL    013500   597    2117#
MC       002252   565#   588
MD       002250   564#   586
MES15    016232   1441   2546#
MES3     016135   2261   2535#
MES6     016202   1439   2542#
ML       002246   563#   584
MTEST    002442   571    594    608#   1430
NSUPLY   011600   1773   1780#
N50000   020760   1989   2049   2807#
OPRIN    021170   577*   578*   580    582    584    586    588    2845#
P88      007570   1452   1470   1474#
PC     =%000007   81#    558*   567*   641*   893*   1161*  1176*  1193*  1210*  1227*  1244*  1261*  1278*
                  1295*  1312*  1376*  1383*  1411*  1414*  1424*  1429   1451*  1452*  1453*  1469*  1470*
                  1471*  1505*  1518*  1612*  1622*  1696*  1714*  1723*  1725*  1727*  1729*  1740*  1741*
                  1747*  1772*  1785*  1997*  2006*  2018*  2027*  2058*  2069*  2082*  2092*  2118*  2121*
                  2167*  2181*  2191*  2193*  2209*  2211*  2213*  2215*  2230*  2255*  2266*  2292*  2295*
                  2361*  2369*  2392*  2408*  2475*  2479*  2931*  2984*  2994*  3000*  3047*  3093   3213*
                  3232*  3239*  3246*  3260*  3262*  3318*  3472*  3489*  3656*  3658*  3660*  3707*  3770*
                  3774*  3779*  3783*  3802*  3805*  3822*  3839*  3870*  4008*  4035*  4038*
PICO     007402   1437   1444#  1698   2381
PIC1     007502   1455   1462#  2382
PIC3     010024   1473   1514#  2383
PIC4     010562   1614   1618#  2384
PIC4B    010612   1623#
PIC6     011264   1712#  2385
```

```
PIRQ   = 177772            67#
PIRQVE= 000240            161#
PR0    = 000000             84#
PR1    = 000040             85#
PR2    = 000100             86#
PR3    = 000140             87#
PR4    = 000200             88#
PR5    = 000240             89#
PR6    = 000300             90#
PR7    = 000340             91#
PS     = 177776             64#     65     528*   4030*
PSUPLY  011536           1757    1758    1767#
PSW    = 177776             65#
PWRMSG  022274           3092    3099#
PWRVEC= 000024            156#    486*    487*   3061*   3062*   3071*   3077*   3089*   3090*
P3      010124           1527#   1615
P3A     010252           1543    1549#   1562
P3B     010332           1565#   1578
P3C     010412           1581#   1594
P3D     010472           1597#   1611
P4      010662           1631#   1699
P4A     011010           1648    1654#   1669
P4B     011132           1678#   1695
P49976  020773           2010    2073    2811#
P7CNT   021172           2846#
P7PNT   021174           2847#
QMARK   017517            591    2183    2672#
RDCHR  = 104410           3377    3539#
RDLIN  = 104411            576    3418    3540#
RDOCT  = 104412           3541#   4068    4087
RD1     026142           3806*   3843#
RESVEC= 000010             151#
ROUTPT  015052           2376    2381#
RTCCSR  026614           4026    4029*   4033*   4044#
R0     =%000000             72#     515     517*    520*    521     526*    540*    541*    542     544*    546*   1054*   1071*
                         1102*   1132*   1147*   1333*   1339*   1360*   1369*   1421*   1424    1541*   1561*   1564*   1577*
                         1580*   1593*   1596*   1610*   1646*   1668*   1677*   1694*   2123*   2124    2125*   2129    2133
                         2137    2194*   2195    2212*   2214*   2227*   2228*   2245*   2256*   2273*   2311*   2312    2349*
                         2353*   2378    2463    2465*   2467*   2469*   2470    2473*   2478*   2862    2872*   2876    2892
                         2893    2906*   3019    3020*   3021*   3028*   3029*   3030*   3031*   3032*   3033    3038    3043*
                         3045*   3049    3051    3063    3088*   3206    3207*   3212    3217    3220*   3415    3419*   3422
                         3438*   3449    3457*   3461    3462    3464*   3465*   3466    3488*   3507    3508*   3509    3510*
                         3511*   3512*   3513*   3720    3721*   3732*   3735    3743*   3767    3768*   3769*   3771    3772
                         3784*   3799    3800*   3801*   3803    3806    3841*   3894    3897*   3913*   3936    3939*   3957*
ROX       007470         1444*   1448*   1456#   1462*   1466*   1495    1498    1511    1722*   1724*   1726*   1728*   1735
                         1738
ROY       007474         1458#
R1     =%000001             73#     516     518*    522     525*    532*    533*    534*    535     545*    546     547     548
                         2124*   2143    2246*   2253*   2276*   2279*   2312*   2314*   2315    2318*   2319*   2326*   2327*
                         2336*   2340*   2350*   2351*   2464    2466*   2467    2472*   2477*   2863    2876*   2877    2881
                         2905*   3064    3087*   3416    3420*   3424*   3426*   3428*   3431*   3434    3437*   3450    3487*
                         3598    3599*   3600*   3603*   3651*   3652    3657    3659    3686*   3855    3856*   3859*   3865*
                         3869*   3895    3898*   3903*   3906    3912*   3937    3940*   3946*   3949    3956*
R1X       007476         1459#   1519*   1523*   1532    1538    1552    1555    1584    1587    1623*   1627*   1638    1642
```

```
                    1667    1677    1691    1694
R1Y     007472      1445*   1449*   1457#   1463*   1467*   1492    1509
R2      =%000002      74#    531*    533     535*    536     537    1486*   1495    2127*   2131*   2135*   2139*   2141*
                    2144    2277*   2280*   2864    2875*   2879*   2882    2889*   2890*   2891    2896*   2904*   3065
                    3086*   3417    3421*   3425*   3427*   3429*   3435    3436*
R2Y     007500      1460#   1520*   1524*   1534    1540    1568    1571    1600    1603    1624*   1628*   1636    1644
                    1662    1665    1674    1686    1669
R22     020422      2763#
R3      =%000003      75#   1485*   1503    1525*   1629*   2147*   2148*   2153*   2154*   2865    2873*   2874*   2888*
                    2891*   2900*   2901*   2903*   3066    3085*   3141    3150*   3156*   3157*   3160*   3165*   3166*
                    3167    3176*   3373    3374*   3375    3378*   3379    3383    3385    3387*   3389*
R37     020534      2017    2777#
R38     020310      1996    2749#
R39     020601      2026    2784#
R4      =%000004      76#   1484*   1526*   1552    1568    1584    1600    1630*   1662    1686    1691    3067    3084*
                    3142    3144*   3145*   3146*   3147    3148*   3162    3164*   3172*   3175*   3719    3744*   3749*
R40     020355      2005    2756#
R41     020646      2081    2791#
R43     020713      2091    2798#
R44     020467      2068    2770#
R5      =%000005      77#    614*    653*    656*    659*    662*    665*    677*    679*    690*    692*    705*    707*
                     719*    721*    732*    734*    747*    749*    762*    764*    775*    777*    790*    792*    806*
                     808*    819*    821*    834*    836*    848*    851*    854*    857*    860*    863*    870*    877*
                     884*    896*    907*    910*    916*    919*    930*    933*    939*    942*    954*    957*    964*
                     967*    978*    981*    987*    990*   1000*   1003*   1014*   1017*   1028*   1031*   1042*   1045*
                    1059*   1062*   1076*   1078*   1088*   1090*   1107*   1109*   1118*   1120*   1137*   1140*   1143*
                    1160*   1164*   1175*   1179*   1192*   1196*   1209*   1213*   1226*   1230*   1243*   1247*   1260*
                    1264*   1277*   1281*   1294*   1298*   1311*   1315*   1328*   1330*   1333*   1337*   1345*   1360*
                    1364*   1367*   1374*   1477*   1481*   1484*   1492*   1495*   1498*   1500*   1503*   1509*   1511*
                    1517*   1532*   1534*   1538*   1540*   1546*   1549*   1552*   1555*   1557*   1560*   1568*   1571*
                    1573*   1576*   1584*   1587*   1589*   1592*   1600*   1603*   1605*   1608*   1621*   1636*   1638*
                    1642*   1644*   1651*   1654*   1657*   1660*   1662*   1665*   1667*   1674*   1677*   1681*   1684*
                    1686*   1689*   1691*   1694*   1719*   1722*   1735*   1738*   1760*   1768*   1781*   1790*   1803*
                    1804*   1817*   1820*   1821*   1835*   1836*   1849*   1850*   1863*   1864*   1877*   1881*   1887*
                    1890*   1893*   1899*   1906*   1907*   1922*   1923*   1933*   1941*   1945*   1947*   1954*   1957*
                    1960*   1968*   1971*   1974*   1980*   1988*   1993*   2000*   2004*   2009*   2014*   2021*   2025*
                    2030*   2035*   2048*   2053*   2063*   2067*   2072*   2077*   2087*   2096*   2102*   2147*   2153*
                    2198*   2200*   2202*   2204*   2221*   2223*   2225*   2227*   2240*   2242*   2245*   2251*   2272*
                    2311    2318*   2323*   2326*   2331*   2334*   2340*   2342*   2344    2349*   2355*   2399    2400
                    2404*   2420    2428*   2431*   2434*   2438*   2441*   2449*   2866    2868*   2870*   2877*   2881*
                    2896    2902*   3068    3083*   3143    3149*   3151*   3153*   3154*   3155*   3156    3174*   3591*
                    3613*   3667*   3687*   3694*   3745*   3785*   3808*   3825*   3842*   3909*   3914*   3917*   3951*
                    3958*   3962*   3974    3976    3981*   3982*   3999*   4073*   4090*   4125*   4132*   4139*
R6      =%000006      78#    474*    475*    476
R7      =%000007      79#
SDELAY  026532      4025#
SELD01  017604      1751    2686#
SELD23  017651      2046    2693#
SNDVLT  014532      1988    2009    2048    2072    2311#
SP      =%000006      80#    478*    492*    500*    504     515*    516*    525     526     574*    577     628*    667
                    1418*   1435*   1713*   1746*   2110    2117*   2190*   2208*   2378    2463*   2464*   2472    2473
                    2474*   2477    2478    2862*   2863*   2864*   2865*   2866*   2867*   2868    2871*   2884    2886*
                    2888    2898    2900    2902    2903    2904    2905    2906    2908*   2909*   2937*   2940    2942
                    2943    2965    2967*   2989    3019*   3024*   3045    3049*   3063*   3064*   3065*   3066*   3067*
```

```
                    3068*  3069*  3070   3078*  3082   3083   3084   3085   3086   3087   3088   3093*  3133*
                    3134   3135   3136*  3141*  3142*  3143*  3149   3174   3175   3176   3177*  3178*  3206*
                    3207   3217*  3219   3220   3221*  3223   3225   3227   3233   3235*  3237*  3245*  3249
                    3253   3254   3258   3278*  3279*  3280   3287*  3290*  3291*  3295*  3296*  3300   3303*
                    3307   3309   3311   3312*  3319   3321   3323*  3324   3326*  3327*  3328*  3329*  3330*
                    3345*  3346*  3349*  3350*  3351   3355*  3356*  3357   3360   3362   3364*  3373*  3378
                    3389   3390*  3391*  3392*  3413*  3414*  3415*  3416*  3417*  3419   3422*  3430*  3431
                    3433   3434*  3436   3437   3438   3449*  3450*  3457   3458*  3469   3470*  3471*  3481
                    3482*  3487   3488   3507*  3508   3518*  3519*  3566*  3588   3589   3598*  3686   3719*
                    3720*  3743   3744   3767*  3784   3799*  3841   3855*  3869   3894*  3895*  3912   3913
                    3936*  3937*  3956   3957   4069   4077*  4088
SPREAD  015420      1771*  1784*  2407*  2461#  2470
STACK = 001100        55#   478    574   1435   1713   1746   2117   2190   2208
STAT    001536       458#   540    653    860    896    907    910    916    919    930    933    939    942
                     954    957    964    967    978    981    987    990   1000   1003   1014   1017   1028
                    1031   1042   1045   1059   1062   1076   1078   1090   1107   1109   1120   1137   1140
                    1143   1160   1164   1175   1179   1328   1330   1333   1337   1345   1360   1364   1367
                    1374   1387*  1477   1481   1484   1500   1503   1517   1525   1546   1549   1557   1560
                    1573   1576   1589   1592   1605   1608   1621   1629   1651   1654   1657   1660   1681
                    1684   1719   1722   1803   1817   1820   1835   1849   1863   1877   1881   1887   1890
                    1893   1899   1906   1922   1933   1941   1945   1947   1960   1974   2141   2240   2242
                    2245   2251   2272   2273   2816   2826   2827   2829
STKLMT= 177774        66#
STX   = 000001      2805#  2807   2811
SUPOK   011642      1786   1789#
SWR     001140       252#   476    494*   496    502*   509*  1130   1187   1204   1221   1238   1255   1272
                    1289   1306   1323   1355   1446   1464   1477   1487   1521   1527   1542   1625   1631
                    1647   1715   2123   2194   2212   2235   2247   2258   2263   2274   2362   2932   2946
                    2948   2954   2992   3004   3069   3082*  3274   3311*
SWREG   000176       187#   502   3274   3287
SW0   = 000001       119#
SW00  = 000001       109#   119
SW01  = 000002       108#   118
SW02  = 000004       107#   117
SW03  = 000010       106#   116   1446   1464   1521   1625   2274
SW04  = 000020       105#   115
SW05  = 000040       104#   114
SW06  = 000100       103#   113
SW07  = 000200       102#   112
SW08  = 000400       101#   111
SW09  = 001000       100#   110   1187   1204   1221   1238   1255   1272   1289   1306   1487   1527   1631
SW1   = 000002       118#
SW10  = 002000        99#  1323   1355   1477   1542   1647   1715
SW11  = 004000        98#
SW12  = 010000        97#  2258
SW13  = 020000        96#
SW14  = 040000        95#
SW15  = 100000        94#
SW2   = 000004       117#
SW3   = 000010       116#
SW4   = 000020       115#
SW5   = 000040       114#
SW6   = 000100       113#
SW7   = 000200       112#
```

```
SW8    = 000400            111#
SW9    = 001000            110#
TBITVE= 000014             152#
TEMP    021154             466*   470*   568    1133*  1145*  2839#
TEMp1   021156            2272*  2282   2840#
TICKS   021152            1450*  1468*  1517*  1621*  2366*  2390*  2838#
TIME    026604            4028*  4031   4036*  4040#  4042*
TIMER   014756            1453   1471   1612   1696   2360#
TIMERA  014772            2364#
TIMER1  015014            2367   2369#
TIMER2  015016            2365   2374#
TIMER4  015046            2379#
TIMSV   021150            2362*  2364   2374*  2375*  2376*  2377*  2379   2837#
TITLE   015472             573   2483#
TKVEC  = 000060            159#
TPVEC  = 000064            160#
TRAPVE= 000034             158#   484*   485*
TRTVEC= 000014             153#
TST1    002616             623    647#
TST10   003304             758#
TST11   003344             765    771#
TST12   003406             778    784#
TST13   003464             802#
TST14   003524             809    815#
TST15   003566             822    828#
TST16   003644             844#
TST17   004114             885    890#
TST2    002744             673#
TST20   004160             897    903#
TST21   004276             920    926#
TST22   004414             943    950#
TST23   004532             968    974#
TST24   004650             991    996#
TST25   004720            1004   1010#
TST26   004770            1018   1024#
TST27   005040            1032   1038#
TST3    003004             680    686#
TST30   005110            1046   1052#
TST31   005170            1063   1069#
TST32   005316            1082   1092   1100#
TST33   005444            1113   1121   1128#
TST34   005562            1131   1144   1155#
TST35   005644            1165   1170#
TST36   005726            1180   1185#
TST37   006016            1188   1197   1202#
TST4    003046             693    699#
TST40   006106            1205   1214   1219#
TST41   006174            1222   1231   1236#
TST42   006262            1239   1248   1253#
TST43   006354            1256   1265   1270#
TST44   006446            1273   1282   1287#
TST45   006540            1290   1299   1304#
TST46   006632            1307   1316   1321#
TST47   006774            1324   1341   1346   1353#
```

```
TST5    003124              715#
TST50   007120             1356    1381#
TST51   011534             1766#
TST52   011576             1779#
TST53   011706             1788    1799#
TST54   011762             1808    1814#
TST55   012046             1825    1831#
TST56   012122             1840    1845#
TST57   012176             1854    1859#
TST6    003164              722     728#
TST60   012252             1868    1873#
TST61   012422             1902#
TST62   012476             1911    1918#
TST63   012570             1937#
TST64   013014             1985#
TST65   013224             2042#
TST7    003226              735     741#
TYPDS = 104405             1419    3534#
TYPE  = 104401              572     590     601     626     632    1417    1420    1438    1440    1748    1750    1792    1995
                           2004    2016    2025    2045    2056    2067    2080    2090    2106    2112    2119    2182    2260
                           2907    2995    3018    3035    3037    3040    3042    3046    3053    3091    3168    3228    3285
                           3286    3289    3302    3313    3332    3381    3384    3388    3530#   3583    3637    3641    3645
                           3873    4059    4066    4083
TYPOC = 104402             3026    3050    3288    3531#   4078
TYPON = 104404             3533#
TYPOS = 104403              629    3532#
VADDR   001526              454#    616    1387    1388    1389    1390    1391
VCXTMP  011250             1702#
VCYTMP  011252             1703#
VECTOR  001456              446#
VECTPS  001460              447#
VERSN   001462              449#
VISUAL  007354              595    1435#
VVECT   001530              455#   1392    1393
V1754   015400             1767    2453#
V24     015402             1771    1780    1784    2454#
WAITIN  002300              574#    592     604
W1      025772             3771*   3786#
W2      025774             3775*   3787#
W3      025776             3780*   3788#
XDACUT  015202             2399*   2401*   2412*   2417#
XPOS    011256             1705#
YPOS    011254             1704#
YPT     011262             1707#
$AERR   025522              614    3590*   3683*   3709#   3999
$APTHD  001000              212     218#
$ASTAT= ****** U           3475    3490
$ATYC   023754             3446    3448#
$ATY1   023730             3444#
$ATY3   023736             3213    3445#
$ATY4   023746             3000    3447#
$AUTOB  001134              249#   3282    3403
$BASE   001250              315#    541     608
$BDADR  001122              244#
```

```
$BDDAT  001126           246#   608*   614    616*   679    692    707    721    734    749    764    777    792
                         808    821    836    863    870    877    884    896    910    919    933    942    957
                         967    981    990   1003   1017   1031   1045   1062   1078   1090   1091   1109   1120
                        1164   1179   1196   1213   1230   1247   1264   1281   1298   1315   1345   1374   1770*
                        1783*  1806*  1807   1823*  1824   1838*  1839   1852*  1853   1866*  1867   1881   1893
                        1909*  1910   1925*  1926   1947   1960   1974   2144*  2147   2153   2448*  2466   2816
                        2818   2820   2822   2824   2829
$CDW1   001254           317#
$CHARC  023046          3230*  3240*  3247   3256*  3261#
$CKSWR  023052          3274#  3538
$CMTAG  001100           232#   473    474    482    488    489
$CM1    =  000002        264#   265#   266#
$CM2    =  000004        264#   265#   266#
$CM3    =  000002        262#   264
$CNT    026472          3977*  3979*  3985#
$CNTLG  023577          3285   3398#
$CNTLU  023572          3302   3397#
$CPUOP  001222           289#
$CRLF   001171           268#   2995   3010   3018   3037   3042   3046   3229   3264   3313   3397
$DATR   026144          3823*  3840*  3844#  3955
$DBLK   021412          2873   2907   2915#
$DEVCT  001204           280#
$DEVm   001252           316#
$DOAGN  007330          1413   1422   1428#
$DTBL   021402          2876   2911#
$ENDAD  007320           198   1424#
$ENDCT  007266          1415#
$ENDMG  007337          1417   1432#
$ENULL  007334          1420   1431#
$ENV    001214           285#  2997   3208   3453   3477
$ENvM   001215           286#   507   3210   3215   3455
$EOP    007232          1386   1405#
$EOPCT  007260          1412#  1416
$ERFLG  001103           235#  2921   2950   2952*  2970   2985*  3010
$ERMAX  001115           241#  2970
$ERrOR  021636           482   2982#
$ERRPC  001116           242#  2816   2818   2820   2822   2824   2826   2827   2829   2989*  2990*  2991   3010
                        3024
$ERRTB  001256           334#  3032
$ERRTY  021766          2994   3017#
$ERtTL  001112           239#  2988*  3010
$ETABL  001214           284#
$ETEND  001256           224    318#
$FAtAL  001176           277#  3481*
$FFLG   024174          3444*  3447*  3475   3484*  3492#
$FILLC  001156           260#  3233   3264
$FILLS  001155           259#  3264
$GDADR  001120           243#
$GDDAT  001124           245#   674*   677    679    687*   690    692    701*   705    707    710*   716*   719
                         721    729*   732    734    743*   747    749    752*   759*   762    764    772*   775
                         777    786*   790    792    795*   803*   806    808    816*   819    821    830*   834
                         836    839*   860*   863    867*   870    874*   877    881*   884    892*   896    907*
                         910    916*   919    930*   933    939*   942    954*   957    964*   967    978*   981
                         987*   990   1000*  1003   1014*  1017   1028*  1031   1042*  1045   1055*  1059*  1062
```

```
                    1072*  1084*  1093*  1103*  1114*  1122*  1160*  1164   1175*  1179   1192*  1196   1209*
                    1213   1226*  1243*  1260*  1264   1277*  1281   1294*  1298   1311*  1315   1342*  1345
                    1371*  1500*  1503   1767*  1780*  1803*  1807   1820*  1824   1835*  1839   1849*  1853
                    1863*  1867   1877*  1881   1890*  1893   1906*  1910   1922*  1926   1941*  1947   1957*
                    1960   1971*  1974   2143*  2147   2150*  2153   2402*  2413*  2465   2816   2818   2820
                    2822   2824   2829
$GET42  007310      1421#
$GTSWR  023122      3286#  3536
$HD   = 000000        52
$HIBTS  001000       219#
$HIOCT  023726      3435*  3440#
$ICNT   001104       236#  2958*  2959   2961*  2969
$ILLUP  022266      3061   3077   3096#
$INLP   026340       614    653    656    659    662    665    679    692    707    721    734    749    764
                     777    792    808    821    836    863    870    877    884    896    910    919    933
                     942    957    967    981    990   1003   1017   1031   1045   1062   1078   1090   1109
                    1120   1143   1164   1179   1196   1213   1230   1247   1264   1281   1298   1315   1330
                    1337   1345   1367   1374   1481   1500   1546   1552   1557   1568   1573   1584   1589
                    1600   1605   1651   1657   1662   1681   1686   1691   1719   1881   1887   1893   1947
                    1954   1960   1968   1974   2242   2251   2323   2331   2434   2438   3936#  3999   4073
                    4125   4132
$INTAG  001135       250#  3314   3403
$ITEMB  001114       240#  2991*  2999   3010   3021
$LF     001172       269#  3010   3264   3388   3397
$LFLG   024173      3485*  3491#
$LOAD   025524      3591   3719#
$LPADR  001106       237#   489*   648*  1758*  2965*  2967   2969
$LPAI   024644      3565#  3917   3962
$LPERR  001110       238#  1757*
$LPW    026146      3770   3774   3779   3783   3802   3805   3822   3839   3855#
$MADR1  001226       302#
$MADR2  001232       306#
$MADR3  001236       309#
$MADR4  001242       312#
$MAIL   001174       220    224    275#   506   2964   2997   3208
$MAMS1  001224       296#
$MAMS2  001230       304#
$MAMS3  001234       307#
$MAMS4  001240       310#
$MBADR  001002       220#
$MFLG   024172      3445*  3451   3486*  3490#
$MNEW   023615      3289   3401#
$MSGAD  001210       282#  3461*  3464
$MSGLG  001212       283#  3466*
$MSgTY  001174       276#  3459   3467*  3479   3483*
$MSWR   023604      3286   3399#
$MTYP1  001225       297#
$MTYP2  001231       305#
$MTYP3  001235       308#
$MTYP4  001241       311#
$MXCNT  021634      2962   2969#
$NULL   001154       258#  3235   3264
$NWTST= 000001       644#   670#   683#   696#   712#   725#   738#   755#   768#   781#   799#   812#   825#
                     841#   887#   900#   923#   947#   971#   993#  1007#  1021#  1035#  1049#  1066#  1097#
```

```
                    1125#  1152#  1167#  1182#  1199#  1216#  1233#  1250#  1267#  1284#  1301#  1318#  1350#
                    1378#  1763#  1776#  1796#  1811#  1828#  1842#  1856#  1870#  1899#  1915#  1934#  1982#
                    2039#
$OCNT   022564      3140*  3169*  3182#
$OFS  = 026402      3952#  4128   4129   4135
$OMODE  022566      3135*  3139*  3144   3147*  3158*  3184#
$OUTLP  026242       677    690    705    719    732    747    762    775    790    806    819    834    848
                     851    854    857    860    907    916    930    939    954    964    978    987   1000
                    1014   1028   1042   1059   1076   1088   1107   1118   1137   1140   1160   1175   1192
                    1209   1226   1243   1260   1277   1294   1311   1328   1333   1360   1364   1477   1484
                    1492   1495   1498   1503   1509   1511   1517   1532   1534   1538   1540   1549   1555
                    1560   1571   1576   1587   1592   1603   1608   1621   1636   1638   1642   1644   1654
                    1660   1665   1667   1674   1677   1684   1689   1694   1722   1735   1738   1803   1817
                    1820   1835   1849   1863   1877   1890   1899   1906   1922   1933   1941   1945   1957
                    1971   1993   2004   2014   2025   2053   2063   2067   2077   2087   2147   2153   2198
                    2200   2202   2204   2221   2223   2225   2227   2240   2245   2272   2318   2326   2334
                    2340   2349   2428   2431   2441   3894#  4090
$OVER   021620      2933   2944   2949   2960   2966#
$PASS   001202       279#   506#   603#  1409#  1410#  1418   1431   2956   2970
$PASTM  001006       222#
$PUTS   027012      4124#
$PWRAD  022262      3094#
$PWRDN  022122       486   3061#  3089
$PWRMG  022256      3092#
$PWRUP  022174      3071   3077#
$QUES   001170       267#  3010   3264   3332   3381   3397
$RDCHR  023334      3345#  3539
$RDDEC= ****** U    3542
$RDLIN  023454      3373#  3540
$RDOCT  023626      3413#  3541
$RDSZ = 000010      3366#
$REGAD  001160       262#
$REGO   001162       264#
$REG1   001164       265#
$RESET  026474       893   1161   1176   1193   1210   1227   1244   1261   1278   1295   1312   1376   1383
                    3996#  4006
$RTNAD  007332      1430#
$R2A  = ****** U    3542
$SAD    026470      3974   3976*  3984#
$SAVRE= ****** U    3542
$SAVR6  022272      3070*  3078   3079*  3080*  3098#
$SCOPE  021422       480   2929#
$SETUP= 000117       336#   472#   479    480    482    484    486    488    489   1407   2930   2983   3007
                    3009   3269   3403
$STUP = 177777       336#   472#
$SVLAD  021602      2941   2963#
$SVPC = 000214       196#   201
$SWR  = 164400        30#    52    171    172    173    174    175    176    177    266    267    488    489
                     490    648    674    687    700    716    729    742    759    772    785    803    816
                     829    845    891    904    927    951    975    997   1011   1025   1039   1053   1070
                    1101   1129   1156   1171   1186   1203   1220   1237   1254   1271   1288   1305   1322
                    1354   1382   1402   1408   1423   1429   1431   1767   1780   1800   1815   1832   1846
                    1860   1874   1903   1919   1938   1986   2043   2922   2923   2924   2925   2932   2944
                    2946   2947   2950   2951   2952   2953   2954   2966   2969   2976   2977   2978   2979
```

```
                    2988    2992    3004    3008    3010    3095
$SWREG  001216       287#     509
$SWRMK= 000000       177     178    2925    2926    2948
$TEsTN  001200       278#   2964#
$TIMES  001166       266#    488#    700#    742#    785#    829#    891#   1129#   1156#   1171#   1186#   1203#   1220#
                    1237#   1254#   1271#   1288#   1305#   1322#   1354#   1382#   1408#   1986#   2043#   2953#   2959
                    2962#   2969
$TKB    001146       255#   2162    2174    3267    3278    3295    3349    3355    4065
$TKS    001144       254#   2160    2172    3267    3276    3292   3316#    3347    3353    4063
$TLKR   026000      3799#   3951
$TLKW   025664      3767#   3909
$TMDAT  001534       457#    614     653     656     659     662     665     845#    848#    851#    854#    857#    860
                     904#    907     913#    916     927#    930     936#    939     951#    954     961#    964     975#
                     978     984#    987     997#   1000    1011#   1014    1025#   1028    1039#   1042    1056#   1059
                    1073#   1076    1085#   1088    1104#   1107    1115#   1118    1134#   1137#   1140    1143    1157#
                    1160    1172#   1175    1189#   1192    1206#   1209    1223#   1226    1240#   1243    1257#   1260
                    1274#   1277    1291#   1294    1308#   1311    1325#   1328    1330#   1333    1337    1357#   1360
                    1361#   1364    1367    1474#   1477    1481#   1484    1489#   1492#   1495#   1498    1503    1506#
                    1509    1511    1514#   1517    1529#   1532    1534    1535#   1538    1540    1546#   1549    1552#
                    1555    1557#   1560    1568#   1571    1573#   1576    1584#   1587    1589#   1592    1600#   1603
                    1605#   1608    1618#   1621    1633#   1636    1638    1639#   1642    1644    1651#   1654    1657#
                    1660    1662#   1665    1667    1671#   1674#   1677    1681#   1684    1686#   1689    1691#   1694
                    1719#   1722    1732#   1735#   1738    1800#   1803    1817#   1820    1832#   1835    1846#   1849
                    1860#   1863    1874#   1877    1878#   1887#   1890    1896#   1899    1903#   1906    1919#   1922
                    1930#   1933    1938#   1941    1942#   1945    1954#   1957    1968#   1971    1990#   1993    2004
                    2011#   2014    2025    2050#   2053    2063    2067    2074#   2077    2087    2195#   2198    2200
                    2202    2204    2221    2223    2225    2227    2237#   2240    2242#   2245    2251    2269#   2272
                    2315#   2318    2323#   2326    2331#   2334    2337#   2340    2346#   2349    2425#   2428#   2431
                    2434    2438    2441
$TN     000066       30#     52     623     644     648#    670     674#    680     683     687#    693     696     700#
                     712     716#    722     725     729#    735     738     742#    755     759#    765     768     772#
                     778     781     785#    799     803#    809     812     816#    822     825     829#    841     845#
                     885     887     891#    897     900     904#    920     923     927#    943     947     951#    968
                     971     975#    991     993     997#   1004    1007    1011#   1018    1021    1025#   1032    1035
                    1039#   1046    1049    1053#   1063    1066    1070#   1082    1092    1097    1101#   1113    1121
                    1125    1129#   1131    1144    1152    1156#   1165    1167    1171#   1180    1182    1186#   1188
                    1197    1199    1203#   1205    1214    1216    1220#   1222    1231    1233    1237#   1239    1248
                    1250    1254#   1256    1265    1267    1271#   1273    1282    1284    1288#   1290    1299    1301
                    1305#   1307    1316    1318    1322#   1324    1341    1346    1350    1354#   1356    1378    1382#
                    1756    1763    1767#   1776    1780#   1788    1796    1800#   1808    1811    1815#   1825    1828
                    1832#   1840    1842    1846#   1854    1856    1860#   1868    1870    1874#   1899    1903#   1911
                    1915    1919#   1934    1938#   1982    1986#   2039    2043#
$TOUt   026434      3613    3667    3694    3808    3825    3974#
$TPB    001152       257#   3253#   3264
$TPFLG  001157       261#   3202    3264
$TPS    001150       256#   3251    3264
$TRAP   024176       484    3507#
$TRAP2  024220      3518#   3529
$TRP    000013      3522#   3531#   3532#   3533#   3534#   3535#   3536    3537#   3538    3539#   3540#   3541#   3542#
$TRPAD  024232      3512    3529#
$TSTM   001004       221#
$TSTNM  001102       234#   1394#   1407#   1756#   2921    2948    2963#   2964    2966    2970    2987    3010
$TTYIN  023562      3374    3375    3392    3396#
$TYPBN= ****** U    3535
```

```
$TYPDS   021176            2861#   3534
$TYpE    022570            3202#   3472    3522    3530
$TYpEC   023002            3232    3239    3246    3251#   3252    3318
$TYPEX   023050            3257    3259    3262#
$TYPOC   022366            3138#   3531
$TYPON   022402            3137    3140#   3533
$TYPOS   022342            3133#   3532
$UNIT    001206             281#    610*    617*    620     628     634     637     640*    1384*   1385    2827
$UNITM   001010             223#
$USWR    001220             288#
$UTK     026616            4056#
$VECT1   001244             313#    550     554
$VECT2   001246             314#
$XTSTR   021440            2935#
$$GET4=  000000            1423#
$OFILL   022565            3134*   3138*   3148    3183#
$40CAT=  ****** U          2932    2994
.      = 027122             181#    185#    196     197#    199#    201#    202#    208     209#    211#    213#    231#    270
                            452#    477     489    1431    1435    1795#   2288    2915#   2969    2970    3010    3056#   3073
                           3097    3264    3267    3396#   3397    3403    3493#   3543#   3544#   3952    4062#
.DVLS    001464             451#    524*    609*   3897    3918    3939    3963    4057*
.$ASTA=  ****** U          3445    3448
.$X    = 001000             208#    213
```

```
COMMEN   162#
CROSS    644#  1789  1979  1993  2000  2014  2021  2053  2063  2077  2087
CSPACE   644#  1997  2006  2018  2027  2058  2069  2082  2092
ENDCOM   162#
ERROR     56#   622   639   668   681   694   709   723   736   751   766   779   794   810   823
           838   865   872   879   886   898   912   921   935   944   959   969   983   992  1005
          1019  1033  1047  1064  1080  1094  1111  1123  1149  1166  1181  1198  1215  1232  1249
          1266  1283  1300  1317  1347  1374  1774  1787  1809  1826  1841  1855  1869  1883  1895
          1912  1928  1949  1962  1976  2410  3609  3615  3628  3669  3696  3810  3827
ESCAPE   162#
GETPRI   162#
GETSWR   162#
LRXX    2749#  2756  2763  2770  2777  2784  2791  2798
MOVEI     20#   611   650   654   657   660   663   677   690   705   719   732   747   762   775
           790   806   819   834   861   868   875   882   894   908   917   931   940   955   965
           979   988  1001  1015  1029  1043  1060  1076  1088  1107  1118  1140  1162  1177  1194
          1211  1228  1245  1262  1279  1296  1313  1328  1334  1343  1364  1372  1479  1498  1544
          1549  1555  1565  1571  1581  1587  1597  1603  1649  1654  1660  1678  1684  1689  1717
          1879  1884  1891  1945  1951  1958  1965  1972  2240  2249  2321  2329  2431  2436  3997
MOVEM     19#   675   688   702   717   730   744   760   773   787   804   817   831   846   849
           852   855   858   905   914   928   937   952   962   976   985   998  1012  1026  1040
          1057  1074  1086  1105  1116  1135  1138  1158  1173  1190  1207  1224  1241  1258  1275
          1292  1309  1326  1331  1358  1362  1475  1482  1490  1493  1496  1501  1507  1509  1515
          1530  1532  1536  1538  1547  1553  1558  1569  1574  1585  1590  1601  1606  1619  1634
          1636  1640  1642  1652  1658  1663  1665  1672  1675  1682  1687  1692  1720  1733  1736
          1801  1815  1818  1833  1847  1861  1875  1888  1897  1904  1920  1931  1939  1943  1955
          1969  1991  2002  2012  2023  2051  2061  2065  2075  2085  2145  2151  2196  2198  2200
          2202  2218  2221  2223  2225  2238  2243  2270  2316  2324  2332  2338  2347  2426  2429
          2439
MULT     162#
NEWONE  1250#  1267  1284  1301
NEWTST   162#   644   670   683   696   712   725   738   755   768   781   799   812   825   841
           887   900   923   947   971   993  1007  1021  1035  1049  1066  1097  1125  1152  1167
          1182  1199  1216  1233  1250  1267  1284  1301  1318  1350  1378  1763  1776  1796  1811
          1828  1842  1856  1870  1899  1915  1934  1982  2039
POP      162#  2902  3082  3083  3436  3487  3488
PUSH     162#  2861  3063  3069  3415  3448  3450  3471
REPORT   162#
SCOPE     57#   647   673   686   699   715   728   741   758   771   784   802   815   828   844
           890   903   926   950   974   996  1010  1024  1038  1052  1069  1100  1128  1155  1170
          1185  1202  1219  1236  1253  1270  1287  1304  1321  1353  1381  1406  1766  1779  1799
          1814  1831  1845  1859  1873  1902  1918  1937  1985  2042
SETPRI   162#
SETTRA  3522#  3531  3532  3533  3534  3536  3538  3539  3540  3541
SETUP    162#   472
SKIP     162#   623   636   638   666   680   693   708   722   735   750   765   778   793   809
           822   837   864   871   878   885   897   911   920   934   943   958   968   982   991
          1004  1018  1032  1046  1063  1079  1082  1092  1110  1113  1121  1131  1144  1146  1148
          1165  1180  1188  1197  1205  1214  1222  1231  1239  1248  1256  1265  1273  1282  1290
          1299  1307  1316  1324  1341  1346  1356  1386  1773  1786  1788  1808  1825  1840  1854
          1868  1882  1894  1911  1927  1948  1961  1975  2409  2414
SLASH    162#
SPACE    162#
STARS    162#   194   205   207   214   227   270   273   644   646   670   672   683   685   696
```

```
                 698     712     714     725     727     738     740     755     757     768     770     781     783     799     801
                 812     814     825     827     841     843     887     889     900     902     923     925     947     949     971
                 973     993     995    1007    1009    1021    1023    1035    1037    1049    1051    1066    1068    1097    1099
                1125    1127    1152    1154    1167    1169    1182    1184    1199    1201    1216    1218    1233    1235    1250
                1252    1267    1269    1284    1286    1301    1303    1318    1320    1350    1352    1378    1380    1399    1763
                1765    1776    1778    1796    1798    1811    1813    1828    1830    1842    1844    1856    1858    1870    1872
                1899    1901    1915    1917    1934    1936    1962    1984    2039    2041    2851    2918    2972    3012    3059
                3075    3110    3187    3266    3269    3337    3366    3405    3443    3501
SWRSU            162#    490#
TOUT             454#    3612    3666    3693    3807    3824
TRMTRP          3522#
TSTSW9           644#    1187    1204    1221    1238
TYPBIN           162#
TYPDEC           162#    1418
TYPNAM           162#
TYPNUM           162#
TYPOCS           162#
TYPOCT           162#    3024    3048    3287    4077
TYPTXT           162#    1792    3583    3637    3641    3645    3872    4058    4083
$CAL.             30#    3548
$DMAST            30#
$DMDT             30#
$MMAST            30#
$$CMRE           225#     264     265
$$CMTM           225#
$$ESCA           162#
$$NEWT           162#     644     670     683     696     712     725     738     755     768     781     799     812     825     841
                 887     900     923     947     971     993    1007    1021    1035    1049    1066    1097    1125    1152    1167
                1182    1199    1216    1233    1250    1267    1284    1301    1318    1350    1378    1763    1776    1796    1811
                1828    1842    1856    1870    1899    1915    1934    1982    2039
$$SET           3522#    3531    3532    3533    3534    3536    3538    3539    3540    3541
$$SETM           506#
$$SKIP           162#     623     680     693     722     735     765     778     809     822     885     897     920     943     968
                 991    1004    1018    1032    1046    1063    1082    1092    1113    1121    1131    1144    1165    1180    1188
                1197    1205    1214    1222    1231    1239    1248    1256    1265    1273    1282    1290    1299    1307    1316
                1324    1341    1346    1356    1788    1808    1825    1840    1854    1868    1911
.EQUAT            30#      52
.HEADE            30#      42
.KMADR            17#     420
.KSIS             22#     511
.LOADL            25#    3711
.LPAIN            22#    3549
.PUTCS            24#    4104
.RESET            22#    3986
.SETUP            30#     336     472
.SWRHI            30#     167
.SWRLO           178#
.UTK              29#    4046
.$ACT1            30#     192
.$APTB            30#     271#
.$APTH            30#     203
.$APTY            30#    3441
.$CATC            30#     179
.$CMTA            30#     225
```

.$EOP	30#	1397
.$ERRO	30#	2970
.$ERRT	30#	3010
.$INLP	28#	3921
.$MMAC	18#	
.$OUTL	27#	3882
.$PARM	30#	
.$POWE	30#	3057
.$RDOC	30#	3403
.$READ	30#	3264
.$SAVE	30#	
.$SCOP	30#	2916
.$SPAC	30#	
.$SWDO	30#	
.$TLKw	26#	3760
.$TOuT	454#	3965
.$TRAP	30#	3499
.$TYPD	30#	2849
.$TYPE	30#	3185
.$TYPO	30#	3108

```
                1199  1201  1203  1204  1205  1214  1216  1218  1220  1221  1222  1231  1233  1235  1237
                1238  1239  1248  1250  1252  1254  1255  1256  1265  1267  1269  1271  1272  1273  1282
                1284  1286  1288  1289  1290  1299  1301  1303  1305  1306  1307  1316  1318  1320  1322
                1323  1324  1341  1346  1350  1352  1354  1355  1356  1378  1380  1382  1383  1386  1399
                1400  1401  1402  1403  1404  1406  1412  1415  1417  1421  1423  1429  1431  1432  1763
                1765  1767  1773  1776  1778  1780  1786  1788  1794  1796  1798  1800  1808  1811  1813
                1815  1825  1828  1830  1832  1840  1842  1844  1846  1854  1856  1858  1860  1868  1870
                1872  1874  1882  1894  1899  1901  1903  1911  1915  1917  1919  1927  1934  1936  1938
                1948  1961  1975  1982  1984  1986  1987  2039  2041  2043  2044  2409  2414  2851  2918
                2921  2925  2931  2932  2944  2946  2947  2948  2950  2951  2952  2954  2956  2964  2966
                2968  2969  2970  2972  2975  2984  2988  2992  2994  2995  2997  3004  3008  3009  3010
                3012  3027  3043  3059  3069  3070  3075  3082  3083  3091  3093  3095  3099  3110  3187
                3208  3266  3268  3269  3270  3298  3337  3338  3366  3374  3375  3379  3380  3396  3397
                3403  3405  3408  3420  3443  3445  3448  3475  3490  3501  3507  3511  3522  3531  3532
                3533  3534  3535  3536  3538  3539  3540  3541  3542  3585  3615  3621  3639  3643  3647
                3669  3675  3696  3702  3810  3816  3827  3833  3875  4046  4061  4085
.IFF              54   174   176   177   178   195   199   201   206   208   215   228   231   234   262
                 271   274   478   624   636   638   645   646   647   648   666   671   672   673   674
                 681   684   685   686   687   694   697   698   699   700   708   713   714   715   716
                 723   726   727   728   729   736   739   740   741   742   750   756   757   758   759
                 766   769   770   771   772   779   782   783   784   785   793   800   801   802   803
                 810   813   814   815   816   823   826   827   828   829   837   842   843   844   845
                 864   871   878   886   888   889   890   891   898   901   902   903   904   911   921
                 924   925   926   927   934   944   948   949   950   951   958   969   972   973   974
                 975   982   992   994   995   996   997  1005  1008  1009  1010  1011  1019  1022  1023
                1024  1025  1033  1036  1037  1038  1039  1047  1050  1051  1052  1053  1064  1067  1068
                1069  1070  1079  1083  1093  1098  1099  1100  1101  1110  1114  1122  1126  1127  1128
                1129  1130  1132  1145  1146  1148  1153  1154  1155  1156  1166  1168  1169  1170  1171
                1181  1183  1184  1185  1186  1189  1198  1200  1201  1202  1203  1206  1215  1217  1218
                1219  1220  1223  1232  1234  1235  1236  1237  1240  1249  1251  1252  1253  1254  1257
                1266  1268  1269  1270  1271  1274  1283  1285  1286  1287  1288  1291  1300  1302  1303
                1304  1305  1308  1317  1319  1320  1321  1322  1325  1342  1347  1351  1352  1353  1354
                1357  1379  1380  1381  1382  1383  1386  1400  1403  1407  1413  1416  1431  1764  1765
                1766  1767  1773  1777  1778  1779  1780  1786  1789  1797  1798  1799  1800  1809  1812
                1813  1814  1815  1826  1829  1830  1831  1832  1841  1843  1844  1845  1846  1855  1857
                1858  1859  1860  1869  1871  1872  1873  1874  1882  1894  1900  1901  1902  1903  1912
                1916  1917  1918  1919  1927  1935  1936  1937  1938  1948  1961  1975  1983  1984  1985
                1986  1987  2040  2041  2042  2043  2044  2409  2414  2852  2919  2944  2948  2949  2952
                2969  2970  2973  2975  2989  3008  3009  3010  3013  3028  3057  3060  3076  3091  3111
                3188  3267  3270  3338  3340  3345  3366  3367  3376  3380  3397  3406  3444  3502  3508
                3615  3621  3669  3675  3696  3702  3810  3816  3827  3833
.IFT            1795  2954  2995  3340  3345  3424  3440  3441  3586  3640  3644  3648  3876  4062  4086
.IFTF           1795  2952  2994  3285  3338  3341  3420  3424  3440  3586  3640  3644  3648  3876  4062
                4086
.IIF              42    47    52   171   172   173   175   177   178   185   270   274   479   482   488
                 489   490  1401  1407  1408  1419  1431  1435  2922  2923  2924  2925  2926  2930  2953
                2966  2969  2970  2976  2977  2978  2979  2983  3007  3009  3010  3025  3050  3264  3267
                3288  3389  3397  3403  3530  3531  3532  3533  3534  3536  3538  3539  3540  3541  4078
.IRP             336   472   644   670   683   696   712   725   738   755   768   781   799   812   825
                 841   887   900   923   947   971   993  1007  1021  1035  1049  1066  1097  1125  1152
                1167  1182  1199  1216  1233  1250  1267  1284  1301  1318  1350  1378  1763  1776  1796
                1811  1828  1842  1856  1870  1899  1915  1934  1982  2039  2862  2902  2931  2984  3063
                3069  3082  3083  3415  3436  3449  3450  3471  3487  3488
.LIST             14    30   162   177   185   262   264   265   266   271   274   336   472   490   559
```

```
                 614   644   648   653   656   659   662   665   670   674   677   679   683   687   690
                 692   696   700   705   707   712   716   719   721   725   729   732   734   738   742
                 747   749   755   759   762   764   768   772   775   777   781   785   790   792   799
                 803   806   808   812   816   819   821   825   829   834   836   841   845   848   851
                 854   857   860   863   870   877   884   887   891   896   900   904   907   910   916
                 919   923   927   930   933   939   942   947   951   954   957   964   967   971   975
                 978   981   987   990   993   997  1000  1003  1007  1011  1014  1017  1021  1025  1028
                1031  1035  1039  1042  1045  1049  1053  1059  1062  1066  1070  1076  1078  1088  1090
                1097  1101  1107  1109  1118  1120  1125  1129  1137  1140  1143  1152  1156  1160  1164
                1167  1171  1175  1179  1182  1186  1192  1196  1199  1203  1209  1213  1216  1220  1226
                1230  1233  1237  1243  1247  1250  1254  1260  1264  1267  1271  1277  1281  1284  1288
                1294  1298  1301  1305  1311  1315  1318  1322  1328  1330  1333  1337  1345  1350  1354
                1360  1364  1367  1374  1378  1382  1407  1423  1435  1477  1481  1484  1492  1495  1498
                1500  1503  1509  1511  1517  1532  1534  1538  1540  1546  1549  1552  1555  1557  1560
                1568  1571  1573  1576  1584  1587  1589  1592  1600  1603  1605  1608  1621  1636  1638
                1642  1644  1651  1654  1657  1660  1662  1665  1667  1674  1677  1681  1684  1686  1689
                1691  1694  1719  1722  1735  1738  1759  1763  1767  1776  1780  1795  1796  1800  1803
                1811  1815  1817  1820  1828  1832  1835  1842  1846  1849  1856  1860  1863  1870  1874
                1877  1881  1887  1890  1893  1899  1903  1906  1915  1919  1922  1933  1934  1938  1941
                1945  1947  1954  1957  1960  1968  1971  1974  1982  1986  1993  2004  2014  2025  2039
                2043  2053  2063  2067  2077  2087  2147  2153  2198  2200  2202  2204  2221  2223  2225
                2227  2232  2240  2242  2245  2251  2272  2318  2323  2326  2331  2334  2340  2349  2428
                2431  2434  2438  2441  2925  3009  3366  3522  3530  3531  3532  3533  3534  3535  3536
                3537  3538  3539  3540  3541  3542  3586  3640  3644  3648  3876  3999  4062  4086
.MACRO            17    18    19    20    22    24    25    26    27    28    29    30   178   225   454
                 506   644  1250  2749  3522
.MCALL            30   162   271   490
.MEXIT           319
.NLIST            14    30   162   177   185   262   264   265   266   271   274   336   472   490   559
                 614   644   648   653   656   659   662   665   670   674   677   679   683   687   690
                 692   696   700   705   707   712   716   719   721   725   729   732   734   738   742
                 747   749   755   759   762   764   768   772   775   777   781   785   790   792   799
                 803   806   808   812   816   819   821   825   829   834   836   841   845   848   851
                 854   857   860   863   870   877   884   887   891   896   900   904   907   910   916
                 919   923   927   930   933   939   942   947   951   954   957   964   967   971   975
                 978   981   987   990   993   997  1000  1003  1007  1011  1014  1017  1021  1025  1028
                1031  1035  1039  1042  1045  1049  1053  1059  1062  1066  1070  1076  1078  1088  1090
                1097  1101  1107  1109  1118  1120  1125  1129  1137  1140  1143  1152  1156  1160  1164
                1167  1171  1175  1179  1182  1186  1192  1196  1199  1203  1209  1213  1216  1220  1226
                1230  1233  1237  1243  1247  1250  1254  1260  1264  1267  1271  1277  1281  1284  1288
                1294  1298  1301  1305  1311  1315  1318  1322  1328  1330  1333  1337  1345  1350  1354
                1360  1364  1367  1374  1378  1382  1407  1423  1435  1477  1481  1484  1492  1495  1498
                1500  1503  1509  1511  1517  1532  1534  1538  1540  1546  1549  1552  1555  1557  1560
                1568  1571  1573  1576  1584  1587  1589  1592  1600  1603  1605  1608  1621  1636  1638
                1642  1644  1651  1654  1657  1660  1662  1665  1667  1674  1677  1681  1684  1686  1689
                1691  1694  1719  1722  1735  1738  1759  1763  1767  1776  1780  1795  1796  1800  1803
                1811  1815  1817  1820  1828  1832  1835  1842  1846  1849  1856  1860  1863  1870  1874
                1877  1881  1887  1890  1893  1899  1903  1906  1915  1919  1922  1933  1934  1938  1941
                1945  1947  1954  1957  1960  1968  1971  1974  1982  1986  1993  2004  2014  2025  2039
                2043  2053  2063  2067  2077  2087  2147  2153  2198  2200  2202  2204  2221  2223  2225
                2227  2232  2240  2242  2245  2251  2272  2318  2323  2326  2331  2334  2340  2349  2428
                2431  2434  2438  2441  2925  3009  3366  3522  3530  3531  3532  3533  3534  3535  3536
                3537  3538  3539  3540  3541  3542  3586  3640  3644  3648  3876  3999  4062  4086
.PAGE             30   225   320
```

```
.PSECT      14
.REM         1      14      30
.REPT      185     264
.SBTTL      52     167     179     188     192     203     225     271     320     472     530     539     559     560     606
           644     670     683     696     712     725     738     755     768     781     799     812     825     841     887
           900     923     947     971     993    1007    1021    1035    1049    1066    1097    1125    1152    1167    1182
          1199    1216    1233    1250    1267    1284    1301    1318    1350    1378    1397    1435    1443    1461    1512
          1616    1710    1745    1763    1776    1796    1811    1828    1842    1856    1870    1899    1915    1934    1982
          2039    2115    2188    2206    2232    2233    2268    2357    2481    2849    2916    2970    3010    3057    3108
          3185    3264    3403    3441    3499    3522
.TITLE      42
.WORD      185     186     187     200     219     220     221     222     223     224     233     236     237     238     239
           242     243     244     245     246     247     248     251     252     253     262     264     265     276     277
           278     279     280     281     282     283     287     288     289     302     306     309     312     313     314
           315     316     317     429     432     434     436     438     440     442     444     446     447     449     451
           614     653     656     659     662     665     677     679     690     692     705     707     719     721     732
           734     747     749     762     764     775     777     790     792     806     808     819     821     834     836
           848     851     854     857     860     863     870     877     884     896     907     910     916     919     930
           933     939     942     954     957     964     967     978     981     987     990    1000    1003    1014    1017
          1028    1031    1042    1045    1059    1062    1076    1078    1088    1090    1107    1109    1118    1120    1137
          1140    1143    1160    1164    1175    1179    1192    1196    1209    1213    1226    1230    1243    1247    1260
          1264    1277    1281    1294    1298    1311    1315    1328    1330    1333    1337    1345    1360    1364    1367
          1374    1412    1415    1430    1456    1457    1458    1459    1460    1477    1481    1484    1492    1495    1498
          1500    1503    1509    1511    1517    1532    1534    1538    1540    1546    1549    1552    1555    1557    1560
          1568    1571    1573    1576    1584    1587    1589    1592    1600    1603    1605    1608    1621    1636    1638
          1642    1644    1651    1654    1657    1660    1662    1665    1667    1674    1677    1681    1684    1686    1689
          1691    1694    1719    1722    1735    1738    1803    1817    1820    1835    1849    1863    1877    1881    1887
          1890    1893    1899    1906    1922    1933    1941    1945    1947    1954    1957    1960    1968    1971    1974
          1993    2004    2014    2025    2053    2063    2067    2077    2087    2147    2153    2198    2200    2202    2204
          2221    2223    2225    2227    2240    2242    2245    2251    2272    2318    2323    2326    2331    2334    2340
          2349    2428    2431    2434    2438    2441    3036    3041    3092    3094    3184    3214    3261    3440    3473
          3529    3592    3689    3709    3844    3910    3911    3918    3953    3963    3984    3985    3999    4009    4010
          4040    4044    4074    4075    4080    4091    4092    4095    4096    4126    4127    4133    4134    4140

        000000

ERRORS DETECTED:  0
```

```
*DRLPB,DRLPB/SOL/CRF=DRLPA.MAC,DRLP3
RUN-TIME: 30 19 3 SECONDS
CORE USED:  41K
```