AA11-K

DAIGNOSTIC

MD-11-DZAAC-A

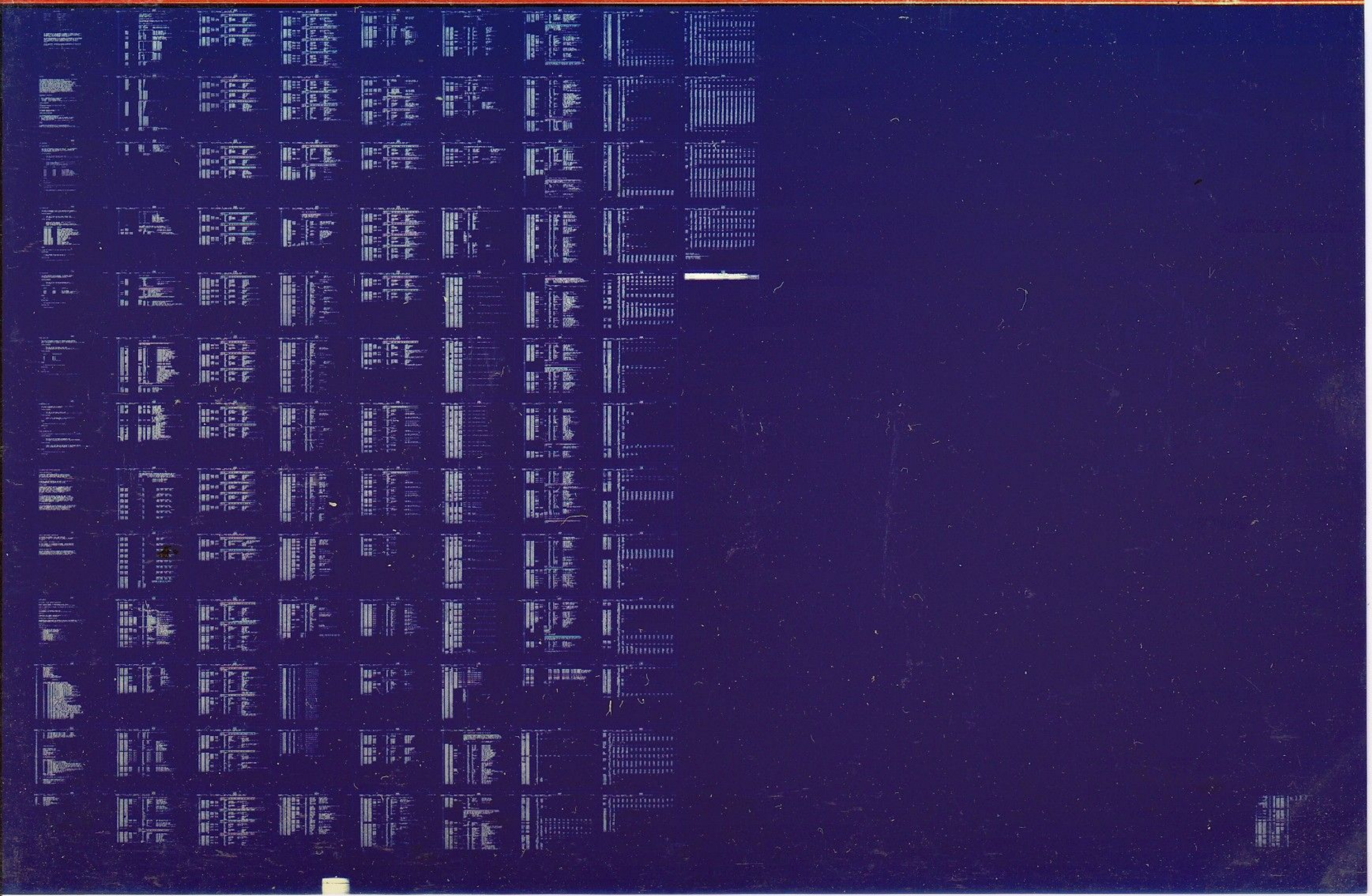
EP-DZAAC-A-DL-A

OCT 1976

digital

FICHE 1 OF 1

Made In U.S.A



IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZAAC-A

PRODUCT NAME: AA11-K DIAGNOSTIC TEST

DATE: OCTOBER 1976

MAINTAINER: DIAGNOSTIC GROUP

FIRST PRINTING, 1976

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 SIX (6) 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 TEST "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.

2. REQUIREMENTS (EQUIPMENT)

A. PDP-11 COMPUTER WITH 8K OF MEMORY AND A CONSOLE I/O TERMINAL
B. AA11-K QUAD OPTION MODULE INSTALLED
C. (OPTIONAL) VR14 OR STORAGE SCOPE
D. (OPTIONAL) UNIQUE HARDWARE TO PERFORM THE AUTO CALIBRATION TEST.

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.

A. TEST PROCEDURE

B. ALTO 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 ALTO 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	EXTURNAL 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 5 SECONDS WITH OUT TEST INTERATIONS.
IT TAKES APPROXIMATELY 30 SECONDS WITH 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. STARTING SEQUENCE

1. TYPE 'AD' TO RUN THE AUTO VISUAL DISPLAY TEST.
2. 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 EIGHT DISPLAY PATTERNS
CONSOLE SW08=1	SELECT PATTERNS IN SW 00-02
CONSOLE SW07=0	HORIZONTAL SETTLING TEST ‹SETTLING TEST›
CONSOLE SW07=1	VERTICAL SETTLING TEST ‹SETTLING TEST›
CONSOLE SW05=0	STORAGE SCOPE NOT CONNECTED ‹PHOSPHOR TEST›
CONSOLE SW05=1	STORAGE SCOPE CONNECTED ‹PHOSPHOR TEST›
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"
CONSOLE SW00-02=4	DISPLAY SETTLING TEST
CONSOLE SW00-02=5	DISPLAY CHARACTER TEST
CONSOLE SW00-02=6	DISPLAY CHANNEL TEST ‹VR14›
CONSOLE SW00-02=7	DISPLAY ERASE AND PHOSPHOR ‹STORAGE SCOPE›

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.

F. EXECUTION TIME

IT TAKES APPROXIMATELY 60 SECONDS TO THIS TEST.

7.3 AUTO CALIBTARION

A. THIS TESTS REQUIRES UNIQUE HARDWARE. THE PROGRAM CONTROLS A KNOWN GOOD AA11-K AND A PROGRAMABLE VOLTAGE STANDARD. THE OPERATOR IS INSTRUCTED TO ADJUST THE POT'S.

B. STARTING SEQUENCE

1. TYPE 'AC' TO RUN THE AUTO CALIBRATION TEST.
2. THE PROGRAM WILL NOW INFORM THE OPERATOR WHAT TO DO.

C. CONTROL SWITCHES

SWITCH	OCTAL	FUNCTION
SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON CURRNET OPERATION
SW13=1	020000	INHIBIT ERROR TYPEOUT
SW08=1	0004XX	LOOP ON SELECTED FUNCTION TEST

D. ERRORS

REF. TO 6.

E. RESTRICTIONS

TEST REQUIRES UNIQUE HARDWARE TO OPERATE.

F. EXECUTION TIME

OPERATOR DEPENDANT

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 COPERATION 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 EXTURNAL 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 EANBLE 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 Y) 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.

DISPLAY SETTLING TEST

A TWO CYCLE SQUARE WAVE WILL BE DISPLAYED TO TEST THE SETTLING DELAY. IF A SETTLING PROBLEM EXISTS, THE LEADING EDGES WILL APPEAR TO BE ROUNDED. THE PROGRAM PLOTS A LINE AT THE MINIMUM AXIS VALUE. UPON COMPLETION, ANOTHER LINE IS PLOTTED AT THE MAXIMUM VALUE. THIS IS REPEATED WITH THE RESULT BEING A SQUARE WAVE. SWITCH BIT 7 DETERMINES HORIZ., OR VERT. SETTLING PATTERN.

J01

SEQ 0009

DISPLAY ALPHA-NUMERIC CHARACTER SET

THE ALPHABET AND NUMBERS 0 THRU 9 ARE DISPLAYED.
THE FIRST ROW CONSISTS OF THE LETTERS 'A' THRU 'M'. THE SECOND CONTAINS
THE LETTERS 'N' THRU 'Z'. THE LAST LINE CONTAINS THE NUMBERS
'0' THRU '9'.

DISPLAY CHANNEL 1 AND CHANNEL 2 <VR14>

THE TEXT "CHANNEL 1" IS DISPLAYED ON CHANNEL 1 SWITCH POSITION.
THE TEXT "CHANNEL 2" IS DISPLAYED ON CHANNEL 2 SWITCH POSITION.
THE COMBINED MESSAGE WILL APPEAR IF THE CHANNEL SELECTOR SWITCH IS
IN THE 1 & 2 POSITION.

PHOSPHOR AND ERASE TEST

THIS TEST PROVIDES A METHOD OF CHECKING FOR PHOSPHOR BURNS ON THE
SCREEN. THIS ROUTINE WILL FIRST ERASE THE STORAGE SCOPE SCREEN.
A DECENDING HORIZONTAL LINE IS DISPLAYED IN 'STORE' MODE. THE RESULT
IS AN INTENSIFICATION OF THE ENTIRE SCREEN. IF EXECUTED USING A VR14,
OR DISPLAY SCOPE THE RESULT WILL BE ONLY A DESCENDING LINE.

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 PPOGRAM MUST BE RESTARTED AT 200.

9.2 XXDP/APT NOTES

THIS DIAGNOSTIC IS CHAINABLE UNDER XYDP.
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 AND 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:
   5 SECONDS WITHOUT INTERATIONS.
   30 SECONDS WITH INTERATIONS.
2. EXECUTION TIME OF THE AUTO DISPLAY TEST IS:
   60 SECONDS
3. EXECUTION TIME OF AUTO CALIBRATION IS:
   OPERATOR DEPENDANT
4. EXECUTION TIMME OF THE MANUAL LOGIC LOOP IS:
   OPERATOR DEPENDANT
5. EXECUTION TIME OF THE MANUAL DISPLAU LOOP IS:
   OPERATOR DEPENDANT
6. EXECUTION TIME OF THE MANUAL CALIBRATION LOOP IS:
   OPERATOR DEPENDANT

10. TABLE OF CONTENTS

ATTACHED

```
                                .TITLE  MAINDEC-11-DZAAC-A      AA11-K  DIAGNOSTIC
                                ;*COPYRIGHT (C) 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY RAYMOND SHOOP
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C0),MAR 21, 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
                                .EQUIV  R6,SP           ;;STACK POINTER
                                .EQUIV  R7,PC           ;;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
```

```
(1)  004000          SW11=   4000
(1)  002000          SW10=   2000
(1)  001000          SW09=   1000
(1)  000400          SW08=   400
(1)  000200          SW07=   200
(1)  000100          SW06=   100
(1)  000040          SW05=   40
(1)  000020          SW04=   20
(1)  000010          SW03=   10
(1)  000004          SW02=   4
(1)  000002          SW01=   2
(1)  000001          SW00=   1
(1)                  .EQUIV  SW09,SW9
(1)                  .EQUIV  SW08,SW8
(1)                  .EQUIV  SW07,SW7
(1)                  .EQUIV  SW06,SW6
(1)                  .EQUIV  SW05,SW5
(1)                  .EQUIV  SW04,SW4
(1)                  .EQUIV  SW03,SW3
(1)                  .EQUIV  SW02,SW2
(1)                  .EQUIV  SW01,SW1
(1)                  .EQUIV  SW00,SW0
(1)
(1)                  ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)  100000          BIT15=  100000
(1)  040000          BIT14=  40000
(1)  020000          BIT13=  20000
(1)  010000          BIT12=  10000
(1)  004000          BIT11=  4000
(1)  002000          BIT10=  2000
(1)  001000          BIT09=  1000
(1)  000400          BIT08=  400
(1)  000200          BIT07=  200
(1)  000100          BIT06=  100
(1)  000040          BIT05=  40
(1)  000020          BIT04=  20
(1)  000010          BIT03=  10
(1)  000004          BIT02=  4
(1)  000002          BIT01=  2
(1)  000001          BIT00=  1
(1)                  .EQUIV  BIT09,BIT9
(1)                  .EQUIV  BIT08,BIT8
(1)                  .EQUIV  BIT07,BIT7
(1)                  .EQUIV  BIT06,BIT6
(1)                  .EQUIV  BIT05,BIT5
(1)                  .EQUIV  BIT04,BIT4
(1)                  .EQUIV  BIT03,BIT3
(1)                  .EQUIV  BIT02,BIT2
(1)                  .EQUIV  BIT01,BIT1
(1)                  .EQUIV  BIT00,BIT0
(1)
(1)                  ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(1)  000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
(1)  000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)  000014          TBITVEC=14              ;;"T" BIT
(1)  000014          TRTVEC= 14              ;;TRACE TRAP
```

```
(1)   000014                BPTVEC= 14          ;;BREAKPOINT TRAP (BPT)
(1)   000020                IOTVEC= 20          ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)   000024                PWRVEC= 24          ;;POWER FAIL
(1)   000030                EMTVEC= 30          ;;EMULATOR TRAP (EMT) **ERROR**
(1)   000034                TRAPVEC=34          ;;"TRAP" TRAP
(1)   000060                TKVEC=  60          ;;TTY KEYBOARD VECTOR
(1)   000064                TPVEC=  64          ;;TTY PRINTER VECTOR
(1)   000240                PIRQVEC=240         ;;PROGRAM INTERRUPT REQUEST VECTOR
  15
  16   170416                       ABASE=170416
  17   100360                       AVECT1=100360
  18   000200                       APRIOR=200
```

```
 20
 21                                     .SBTTL  OPERATIONAL SWITCH SETTINGS
(1)                                     ;*
(1)                                     ;*      SWITCH                  USE
(1)                                     ;*      ------          --------------------
(1)                                     ;*        15            HALT ON ERROR
(1)                                     ;*        14            LOOP ON TEST
(1)                                     ;*        13            INHIBIT ERROR TYPEOUTS
(1)                                     ;*        12            STORAGE SCOPE CONNECTED
(1)                                     ;*        11            INHIBIT ITERATIONS
(1)                                     ;*        10            EXTERNAL DELAY SIGNAL CONNECTED
(1)                                     ;*         8            LOOP ON TEST IN SWR<7:0>
 22                                     ;*         9            TWO'S COMPLEMENT MODE
 23                                     .SBTTL  TRAP CATCHER
(1)
(1)             000000                          .=0
(1)                                     ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
(1)                                     ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
(1)                                     ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
(1)             000174                          .=174
(1)  000174     000000          DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
(1)  000176     000000          SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
(1)                                     .SBTTL  STARTING ADDRESS(ES)
(1)  000200     000137  001462          JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
 24  000204     000137  001500          JMP     BEGIN1                  ;JUMP TO RESTART ADDRESS
 25  00021C     000137  013520          JMP     DYNCAL                  ;JUMP TO DYNAMIC DAC CALIBRATION
```

```
 27                                  .SBTTL  ACT11 HOOKS
(1)
(2)
(1)                                  ;;*****************************************************************
(1)                                  ;HOOKS REQUIRED BY ACT11
(1)          000214                          $SVPC=.                 ;SAVE PC
(1)          000046                          .=46
(1)  000046  006566                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
(1)          000052                          .=52
(1)  000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
(1)          000214                          .=$SVPC                 ;; RESTORE PC
 28          001000                          .=1000
 29                                  .SBTTL  APT PARAMETER BLOCK
(2)
(2)                                  ;;*****************************************************************
(1)                                  ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(2)                                  ;;*****************************************************************
(1)          001000                          .$X=.   ;;SAVE CURRENT LOCATION
(1)          000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
(1)  000024  000200                          200     ;;FOR APT START UP
(1)          000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
(1)  000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
(1)          001000                          .=.$X   ;;RESET LOCATION COUNTER
(2)                                  ;;*****************************************************************
(1)                                  ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(1)                                  ;INTERFACE SPEC.
(1)
(1)  001000                          $APTHD:
(1)  001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(1)  001002  001174                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
(1)  001004  000030                  $TSTM:  .WORD   30      ;;RUN TIM OF LONGEST TEST
(1)  001006  000060                  $PASTM: .WORD   60      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(1)  001010  000120                  $UNITM: .WORD   120     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(1)  001012  000031                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
 30                                     .SBTTL  COMMON TAGS
(1)
(2)                                     ;;**********************************************************************
(1)                                     ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
(1)                                     ;*USED IN THE PROGRAM.
(1)
(1)             001100                          .=1100
(1)  001100                             $CMTAG:                 ;;START OF COMMON TAGS
(1)  001100  000000                             .WORD   0
(1)  001102     000             $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
(1)  001103     000             $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
(1)  001104  000000             $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
(1)  001106  000000             $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
(1)  001110  000000             $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
(1)  001112  000000             $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
(1)  001114     000             $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
(1)  001115     001             $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
(1)  001116  000000             $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
(1)  001120  000000             $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
(1)  001122  000000             $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
(1)  001124  000000             $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
(1)  001126  000000             $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
(1)  001130  000000                             .WORD   0       ;;RESERVED--NOT TO BE USED
(1)  001132  000000                             .WORD   0
(1)  001134     000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
(1)  001135     000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
(1)  001136  000000                             .WORD   0
(1)  001140  177570             $SWR:   .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
(1)  001142  177570             DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
(1)  001144  177560             $TKS:   177560                  ;;TTY KBD STATUS
(1)  001146  177562             $TKB:   177562                  ;;TTY KBD BUFFER
(1)  001150  177564             $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
(1)  001152  177566             $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
(1)  001154     000             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
(1)  001155     002             $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
(1)  001156     012             $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
(1)  001157     000             $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
(1)  001160  000000             $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
 1)                                                             ;;WHICH  ($REGO) WAS OBTAINED
(3)  001162  000000             $REGO:  .WORD   0               ;;CONTAINS (($REGAD)+0)
(3)  001164  000000             $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
(1)  001166  000000             $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
(1)  001170     077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
(1)  001171     015             $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
(1)  001172  000012             $LF:    .ASCIZ  <12>            ;;LINE FEED
(2)                                     ;;**********************************************************************
(2)                                     .SBTTL  APT MAILBOX-ETABLE
(2)
(3)                                     ;;**********************************************************************
(2)                                     .EVEN
(2)  001174                             $MAIL:                  ;;APT MAILBOX
(2)  001174  000000             $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
(2)  001176  000000             $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
(2)  001200  000000             $TESTN: .WORD   ATESTN          ;;TEST NUMBER
(2)  001202  000000             $PASS:  .WORD   APASS           ;;PASS COUNT
(2)  001204  000000             $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
```

```
(2)  001206  000000                $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
(2)  001210  000000                $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
(2)  001212  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
(2)  001214                        $ETABLE:                ;;APT ENVIRONMENT TABLE
(2)  001214     000                $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
(2)  001215     000                $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
(2)  001216  000000                $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
(2)  001220  000000                $USWR:  .WORD   AUSWR   ;;USER SWITCHES
(2)  001222  000000                $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
(2)                                ;*                      BITS 15-11=CPU TYPE
(2)                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(2)                                ;*                              11/70=06,PDQ=07,Q=10
(2)                                ;*                      BIT 10=REAL TIME CLOCK
(2)                                ;*                      BIT  9=FLOATING POINT PROCESSOR
(2)                                ;*                      BIT  8=MEMORY MANAGEMENT
(2)  001224     000                $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
(2)  001225     000                $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
(2)                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
(2)                                ;*                              900 NSEC CORE=001
(2)                                ;*                              300 NSEC BIPOLAR=002
(2)                                ;*                              500 NSEC MOS=003
(2)  001226  000000                $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
(2)                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
(2)  001230     000                $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
(2)  001231     000                $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
(2)  001232  000000                $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
(2)  001234     000                $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
(2)  001235     000                $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
(2)  001236  000000                $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
(2)  001240     000                $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
(2)  001241     000                $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
(2)  001242  000000                $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
(2)  001244  100360                $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)  001246  000000                $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(2)  001250  170416                $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)  001252  000000                $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
(2)  001254  000000                $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
(2)  001256                        $ETEND:
(2)                                .MEXIT
```

```
(1)                                     .SBTTL  ERROR POINTER TABLE
(1)
(1)                                     ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1)                                     ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1)                                     ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1)                                     ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS $ERRPC).
(1)                                     ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1)                                     ;*      EM              ;;POINTS TO THE ERROR MESSAGE
(1)                                     ;*      DH              ;;POINTS TO THE DATA HEADER
(1)                                     ;*      DT              ;;POINTS TO THE DATA
(1)                                     ;*      DF              ;;POINTS TO THE DATA FORMAT
(1)
(1)
(1)  001256                             $ERRTB:
 31
 33
 34                                     ;ITEM   1
 35  001256  016024                             EM1                     ;AA11-K STATUS REGISTER IN ERROR
 36  001260  016645                             DH1                     ;ERRPC  IBASE   GOOD    BAD
 37  001262  020264                             DT1                     ;$ERRPC STAT    $GDDAT  $BDDAT
 38  001264  020400                             DF0
 39
 40                                     ;ITEM   2
 41  001266  016055                             EM2                     ; DAC0 REGISTER IN ERROR
 42  001270  016645                             DH1                     ;ERRPC  IBASE   GOOD    BAD
 43  001272  020276                             DT2                     ;$ERRPC DAC0    $GDDAT  $BDDAT
 44  001274  020400                             DF0
 45
 46                                     ;ITEM   3
 47  001276  016104                             EM3                     ; DAC1 REGISTER IN ERROR
 48  001300  016645                             DH1                     ;ERRPC  IBASE   GOOD    BAD
 49  001302  020310                             DT3                     ;$ERRPC DAC1    $GDDAT  $BDDAT
 50  001304  020400                             DF0
 51
 52                                     ;ITEM   4
 53  001306  016133                             EM4                     ;DAC #2 REGISTER IN ERROR
 54  001310  016645                             DH1                     ;ERRPC  IBASE   GOOD    BAD
 55  001312  020322                             DT4                     ;$ERRPC DAC2    $GDDAT  $BDDAT
 56  001314  020400                             DF0
 57
 58                                     ;ITEM   5
 59  001316  016162                             EM5                     ;DAC #3 REGISTER IN ERROR
 60  001320  016645                             DH1                     ;ERRPC  IBASE   GOOD    BAD
 61  001322  020334                             DT5                     ;$ERRPC DAC3    $GDDAT  $BDDAT
 62  001324  020400                             DF0
 63
 64                                     ;ITEM   6
 65  001326  016211                             EM6                     ;AA11-K FAILED TO INTERRUPT
 66  001330  016701                             DH6                     ;ERRPC  IBASE
 67  001332  020346                             DT6                     ;$ERRPC STAT
 68  001334  020400                             DF0
 69
```

```
                                ;ITEM   7
001336  016244                  EM7             ;AA11-K INTERRUPTED IN ERROR
001340  016701                  DH6             ;ERRPC  IBASE
001342  020346                  DT6             ;$ERRPC STAT
001344  020400                  DF0
                                ;ITEM   10
001346  016300                  EM10            ;BUS TIME-OUT ON AA11-K ADDRESSES
001350  016701                  DH6             ;ERRPC  IBASE
001352  020346                  DT6             ;$ERRPC STAT
001354  020400                  DF0
                                ;ITEM   11
001356  016351                  EM11            ;AA11-K READY BI
001360  016645                  DH1             ;ERRPC  IBASE                 D
001362  020264                  DT1             ;$ERRPC STA                   DAT
001364  020400                  DF0
                                ;ITEM   12
001366  016400                  EM12            ;POWE                 AS INCORREC
001370  016747                  DH14            ;ERRPC                  BAD
001372  020366                  DT14            ;$ERR                   $BDDAT
001374  020400                  DF0
                                ;ITEM   13
001376  016443                  EM13            ;A PREVIOUSLY EXISTING AA11-K DOES NOT RESPOND NOW
001400  016715                  DH13            ;ERRPC  BASE    FIRST#  #NOW
001402  020354                  DT13            ;$ERRPC $BASE   EVER    $UNIT
001404  020400                  DF0
                                ;ITEM   14
001406  016464                  EM14            ;AA11-K LOGIC SIGNAL HIGH OUTPUT TO LOW
001410  016747                  DH14            ;ERRPC  BASE    GOOD    BAD
001412  020366                  DT14            ;$ERRPC $BASE   $GDDAT  $BDDAT  SPREAD
001414  020400                  DF0
                                ;ITEM   15
001416  016533                  EM15            ;AA11-K LOGIC LOW OUTPUT TO HIGH
001420  016747                  DH14            ;ERRPC  BASE    GOOD    BAD
001422  020366                  DT14            ;$ERRPC $BASE   $GDDAT  $BDDAT  SPREAD
001424  020400                  DF0
                                ;ITEM   16
001426  016602                  EM16            ;SELECTED DAC HAS A LINEARITY ERROR
001430  016747                  DH14            ;ERRPC  BASE    GOOD    BAD
001432  020366                  DT14            ;$ERRPC $BASE   $GDDAT  $BDDAT
001434  020400                  DF0
001436  000040          VADDR:  40      ;ADJUSTMENT TO NEXT AA11-K'S ADDRESS
001440  000040          VVECT:  40      ;ADJUSTMENT TO NEXT AA11-K'S VECTOR
001442  000012          DELAY:  10.     ;DELAY COUNTER FOR DELAY LOOPS
001444  170416          STAT:   170416
001446  170420          DAC0:   170420
001450  170422          DAC1:   170422
001452  170424          DAC2:   170424
001454  170426          DAC3:   170426
001456  000350          IV:     350
```

```
  127  001460  000352                    IVS:    352
  128
  129  001462  005037  020432            BEGIN:  CLR     TEMP
  130  001466  005037  020442                    CLR     EVER            ;RESET POINTER
  131  001472  005037  020444                    CLR     EVER1
  132  001476  000403                            BR      BEG
  133  001500  012737  000001  020432    BEGIN1: MOV     #1,TEMP
  134  001506  000005                    BEG:    RESET
  136                                     .SBTTL  INITIALIZE THE COMMON TAGS
  (1)                                     ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
  (1)  001510  012706  001100                    MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
  (1)  001514  005026                            CLR     (R6)+           ;;CLEAR MEMORY LOCATION
  (1)  001516  022706  001140                    CMP     #SWR,R6 ;;DONE?
  (1)  001522  001374                            BNE     .-6             ;;LOOP BACK IF NO
  (1)  001524  012706  001100                    MOV     #STACK,SP       ;;SETUP THE STACK POINTER
  (1)                                     ;;INITIALIZE A FEW VECTORS
  (1)  001530  012737  020744  000020            MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
  (1)  001536  012737  000340  000022            MOV     #340,@#IOTVEC+2 ;;LEVEL 7
  (1)  001544  012737  021160  000030            MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
  (1)  001552  012737  000340  000032            MOV     #340,@#EMTVEC+2 ;;LEVEL 7
  (1)  001560  012737  023520  000034            MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)  001566  012737  000340  000036            MOV     #340,@#TRAPVEC+2;LEVEL 7
  (1)  001574  012737  021444  000024            MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
  (1)  001602  012737  000340  000026            MOV     #340,@#PWRVEC+2 ;;LEVEL 7
  (1)  001610  005037  001166                    CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
  (1)  001614  012737  001614  001106            MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
  (2)                                     ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
  (2)                                     ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
  (2)  001622  013746  000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
  (2)  001626  012737  001662  000004            MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
  (2)  001634  012737  177570  001140            MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
  (2)  001642  012737  177570  001142            MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
  (2)  001650  022777  177777  177262            CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
  (2)  001656  001012                            BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  (2)                                                                     ;;AND  THE HARDWARE SWR IS NOT = -1
  (2)  001660  000403                            BR      65$             ;;BRANCH IF NO TIMEOUT
  (2)  001662  012716  001670            64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
  (2)  001666  000002                            RTI
  (2)  001670  012737  000176  001140    65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
  (2)  001676  012737  000174  001142            MOV     #DISPREG,DISPLAY
  (2)  001704  012637  000004            66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
  (1)
  (2)  001710  005037  001202                    CLR     $PASS           ;;CLEAR PASS COUNT
  (2)  001714  132737  000200  001215            BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
  (2)  001722  001403                            BEQ     67$             ;;YES,USE NON-APT SWITCH
  (2)  001724  012737  001216  001140            MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
  (2)  001732                            67$:
  137  001732  005037  177776                    CLR     @#PS
  138  001736  000137  002126                    JMP     INIT1
```

```
                                        .SBTTL  SUBROUTINE TO LOAD A TRAP CATCHER
001742  012702  000242          LDTRAP: MOV     #242,R2                 ;LOAD R2
001746  012701  000240                  MOV     #240,R1                 ;LOAD R1
001752  010221                  5$:     MOV     R2,(R1)+                ;LOAD .+2
001754  005021                          CLR     (R1)+                   ;LOAD HALT
001756  010102                          MOV     R1,R2                   ;LOAD R2
001760  005722                          TST     (R2)+                   ;BUMP R2
001762  020227  001002                  CMP     R2,#1002                ;TEST FOR LAST
001756  001371                          BNE     5$                      ;BR UNTIL DONE
                                        .SBTTL  LOAD DEVICE ADDRESSES LOCATIONS
001770  012700  001444                  MOV     #STAT,R0                ;LOAD POINTER
001774  013720  001250          10$:    MOV     $BASE,(R0)+             ;LOAD BASE ADDRESS
002000  022700  001456                  CMP     #IV,R0                  ;TEST FOR DONE
002004  001373                          BNE     10$                     ;BR
002006  012700  001446                  MOV     #DAC0,R0                ;LOAD 2ND ADDRESS
002012  012701  000002                  MOV     #2,R1                   ;LOAD R1
002016  060120                  12$:    ADD     R1,(R0)+                ;UPDATE REAL DEVICE WORD
002020  005721                          TST     (R1)+                   ;BUMP R1
002022  022701  000012                  CMP     #12,R1                  ;TEST FOR DONE
002026  001373                          BNE     12$                     ;BR
002030  013737  001244  001456          MOV     $VECT1,IV               ;LOAD INTR. VECTOR ADDRESS
002036  042737  160000  001456          BIC     #160000,IV
002044  013737  001456  001460          MOV     IV,IVS
002052  062737  000002  001460          ADD     #2,IVS
002060  113737  001245  020436          MOVB    $VECT1+1,BRLEV1         ;LOAD BR LEVEL
002066  042737  177437  020436          BIC     #177437,BRLEV1
002074  013737  020436  020440          MOV     BRLEV1,BRLEV2
002102  162737  000040  020440          SUB     #40,BRLEV2
002110  000207                          RTS     PC                      ;EXIT
```

```
                                        .SBTTL  INITIAL HEADER TYPEOUT AND WAIT FOR OPERATOR
002112  046101                  AL:     .ASCII  /AL/
002114  042101                  AD:     .ASCII  /AD/
002116  041501                  AC:     .ASCII  /AC/
002120  046115                  ML:     .ASCII  /ML/
002122  042115                  MD:     .ASCII  /MD/
002124  041515                  MC:     .ASCII  /MC/

002126  004737  001742          INIT1:  JSR     PC,LDTRAP
002132  005737  020432                  TST     TEMP            ;TEST IF START OR RESTART
002136  001065                          BNE     CTRLC           ;RESTART
002140  005737  000042                  TST     @#42            ;TEST IF MONITOR
002144  001067                          BNE     MTEST           ;BR IF NOT
002146  104400                          TYPE                    ;CALL MESSAGE PRINTER VIA 'EMT'
002150  014662                          TITLE                   ;TYPE PROGRAM HEADER.
002152  012706  001100          WAITIN: MOV     #STACK,SP       ;RESET STACK
002156  104406                          CKSWR                   ;TEST FOR CTRL G
002160  104410                          RDLIN                   ;READ TWO CHARACTERS
002162  013637  020446                  MOV     @(SP)+,OPRIN    ;READ THE CHARACTER
002166  042737  000000  020446          BIC     #0,OPRIN        ;MASK OTHER CHARACTERS

002174  023737  002112  020446          CMP     AL,OPRIN        ;TEST FOR "AL"
002202  001427                          BEQ     10$             ;BR IF YES
002204  023737  002114  020446          CMP     AD,OPRIN        ;TEST FOR "AD"
002212  001425                          BEQ     11$             ;BR IF YES
002214  023737  002116  020446          CMP     AC,OPRIN        ;TEST FOR "AC"
002222  001423                          BEQ     12$             ;BR IF YES
002224  023737  002120  020446          CMP     ML,OPRIN        ;TEST FOR "ML"
002232  001421                          BEQ     13$             ;BR FI YES
002234  023737  002122  020446          CMP     MD,OPRIN        ;TEST FOR "MD"
002242  001417                          BEQ     14$             ;BR IF YES
002244  023737  002124  020446          CMP     MC,OPRIN        ;TEST FOR "MC"
002252  001415                          BEQ     15$             ;BR IF YES
002254  104400                          TYPE
002256  016774                          QMARK
002260  000734                          BR      WAITIN          ;WAIT FOR OPERATOR AGAIN

002262  000137  002324          10$:    JMP     MTEST           ;RUN THE LOGIC TEST
002266  000137  006622          11$:    JMP     VISUAL          ;RUN THE VISUAL PATTERN
002272  000137  011500          12$:    JMP     AUTCAL          ;RUN THE AUTO CALIBRATION (FACTORY ONLY)
002276  000137  013170          13$:    JMP     MANUL           ;RUN THE MANUAL LOGIC TEST
002302  000137  011364          14$:    JMP     FULRMP          ;RUN THE RAMP PATTERN ON FOUR DAC'S
002306  000137  013454          15$:    JMP     CALDAC          ;RUN THE MANUAL CAL OF THE DAC

002312  104400                  CTRLC:  TYPE
002314  017002                          CONTC
002316  005037  001202                  CLR     $PASS
002322  000713                          BR      WAITIN
```

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

002324  013737  001250  001126  MTEST:  MOV     $BASE,$BDDAT    ;GET THE BASE ADDRESS
002332  005037  001206                  CLR     $UNIT           ;CLEAR UNIT #
002336  012737  002364  000004          MOV     #2$,ERRVEC      ;LOAD TRAP RETURN
002344  005777  176556          1$:     TST     @$BDDAT         ;TEST IF ADDR EXISTS
002350  063737  001436  001126          ADD     VADDR,$BDDAT    ;UPDATE THE BUS ADDRESS
002356  005237  001206                  INC     $UNIT           ;UPDATE UNIT COUNT
002362  000406                          BR      3$              ;;
002364  022626                  2$:     CMP     (SP)+,(SP)+     ;CLEAN THE STACK
002366  005737  001206                  TST     $UNIT           ;TEST IF ANY EXIST
002372  001002                          BNE     3$              ;BR IF SOME ARE THERE
002374  104010                          ERROR   10              ;BASE ADDRESS CAUSED AN BUS TRAP
002376  000434                          BR      TST1            ;;
002400  005737  020442          3$:     TST     EVER            ;TEST IF # HAS BEEN REPORTED
002404  100417                          BMI     4$              ;BR IF IT HAS
002406  104400                          TYPE
002410  017010                          FOUND1                  ;TELL OPERATOR THE # OF AA11-K'S
002412  013746  001206                  MOV     $UNIT,-(SP)
002416  104402                          TYPOS
002420     002                          .BYTE   2
002421     000                          .BYTE   0
002422  104400                          TYPE
002424  017034                          FOUND2
002426  013737  001206  020442          MOV     $UNIT,EVER      ;SAVE THE # OF AA11-K'S FOR LATER
002434  052737  100000  020442          BIS     #BIT15,EVER     ;SET "REPORTED # FLAG"
002442  000405                          BR      5$              ;;
002444  123737  020442  001206  4$:     CMPB    EVER,$UNIT      ;TEST IF ANY HAVE GONE AWAY
002452  001401                          BEQ     5$              ;;BR IF ALL ARE STILL HERE
002454  104013                          ERROR   13              ;EXISTING UNIT FAILED TO RESPOND NOW
002456  005037  001206          5$:     CLR     $UNIT           ;RESET UNIT POINTER
002462  004737  001742                  JSR     PC,LDTRAP       ;LOAD TRAP CATCHER AND BUS ADDRESSES

002466  000240                  LTEST:  NOP
                                        ;;*****************************************************************
                                        ;*TEST 1         TEST THAT THE AA11-K RESPONDS TO THE CPU
                                        ;;*****************************************************************
002470  000004                  TST1:   SCOPE
002472  012737  002506  001106          MOV     #3$,$LPADR
002500  012737  002534  000004          MOV     #1$,ERRVEC      ;LOAD BUS TRAP RETURN
002506  005777  176732          3$:     TST     @STAT           ;TEST THE STATUS REGISTER
002512  005777  176730                  TST     @DAC0           ;TEST THE DAC0
002516  005777  176726                  TST     @DAC1           ;TEST THE DAC1
002522  005777  176724                  TST     @DAC2           ;TEST DAC #2
002526  005777  176722                  TST     @DAC3           ;TEST DAC #3
002532  000402                          BR      2$              ;;BR AND RESTORE LOC. 4
002534  022626                  1$:     CMP     (SP)+,(SP)+     ;CLEAN THE STACK
002536  104010                          ERROR   10              ;ERROR, BUS TIMEOUT WHEN ADDRESSING THE AA11-K
002540  012737  000006  000004  2$:     MOV     #6,ERRVEC       ;LOAD RETURN
```

```
285                                     ;;*****************************************************************
(3)                                     ;*TEST 2         TEST THAT THE DACO REGISTER CAN BE CLEARED
(3)                                     ;;*****************************************************************
(2) 002546 000004                       TST2:   SCOPE
286 002550 005037 001124                        CLR     $GDDAT                  ;LOAD EXPECTED
287 002554 013777 001124 176664                 MOV     $GDDAT,@DACO            ;LOAD REG
288 002562 017737 176660 001126                 MOV     @DACO,$BDDAT            ;READ REG
289 002570 023737 001124 001126                 CMP     $GDDAT,$BDDAT           ;COMPARE
290 002576 001401                               BEQ     TST3            ;;BR IF EQUAL
291 002600 104002                               ERROR   2                       ;ERROR, DACO REGISTER NOT = 0
292
293                                     ;;*****************************************************************
(3)                                     ;*TEST 3         TEST THAT THE DACO REGISTER CAN BE LOADED WITH #7777
(3)                                     ;;*****************************************************************
(2) 002602 000004                       TST3:   SCOPE
294 002604 012737 007777 001124                 MOV     #7777,$GDDAT            ;LOAD EXPECTED
295 002612 013777 001124 176626                 MOV     $GDDAT,@DACO            ;LOAD REG
296 002620 017737 176622 001126                 MOV     @DACO,$BDDAT            ;READ REG
297 002626 023737 001124 001126                 CMP     $GDDAT,$BDDAT           ;COMPARE
298 002634 001401                               BEQ     TST4            ;;BR IF EQUAL
299 002636 104002                               ERROR   2                       ;ERROR,  DACO REGISTER NOT = 7777
300
301                                     ;;*****************************************************************
(3)                                     ;*TEST 4         TEST THAT THE DACO REGISTER CAN HOLD A FLOATING 1 PATTERN
(3)                                     ;;*****************************************************************
(2) 002640 000004                       TST4:   SCOPE
(1) 002642 012737 000100 001166                 MOV     #100,$TIMES     ;;DO 100 ITERATIONS
302 002650 012737 004000 001124                 MOV     #BIT11,$GDDAT           ;LOAD EXPECTED
303 002656 013777 001124 176562 1$:             MOV     $GDDAT,@DACO            ;LOAD DACO REGISTER
304 002664 017737 176556 001126                 MOV     @DACO,$BDDAT            ;READ THE REGISTER
305 002672 023737 001124 001126                 CMP     $GDDAT,$BDDAT           ;COMPARE THE DATA
306 002700 001401                               BEQ     2$              ;;BR IF SAME
307 002702 104002                               ERROR   2                       ;ERROR, DACO REGISTER FAILED TO HOLD A FLOATING
308 002704 006237 001124        2$:             ASR     $GDDAT                  ;CHANGE THE DATA
309 002710 001362                               BNE     1$                      ;BR AND TEST MORE DATA
```

```
                                        ;;*****************************************************************
                                        ;*TEST 5        TEST THAT THE DAC #1 REGISTER CAN BE CLEARED
                                        ;;*****************************************************************
002712  000004                  TST5:   SCOPE
002714  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED
002720  013777  001124  176522          MOV     $GDDAT,@DAC1            ;LOAD Y REG
002726  017737  176516  001126          MOV     @DAC1,$BDDAT            ;READ REG
002734  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
002742  001401                          BEQ     TST6            ;;BR IF EQUAL
002744  104003                          ERROR   3                       ;ERROR,  DAC1 REGISTER NOT = 0

                                        ;;*****************************************************************
                                        ;*TEST 6        TEST THAT THE Y REGISTER CAN BE LOADED WITH #7777
                                        ;;*****************************************************************
002746  000004                  TST6:   SCOPE
002750  012737  007777  001124          MOV     #7777,$GDDAT            ;LOAD EXPECTED
002756  013777  001124  176464          MOV     $GDDAT,@DAC1            ;LOAD Y REG
002764  017737  176460  001126          MOV     @DAC1,$BDDAT            ;READ REG
002772  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003000  001401                          BEQ     TST7            ;;BR IF EQUAL
003002  104003                          ERROR   3                       ;ERROR,  DAC1 REGISTER NOT = 7777

                                        ;;*****************************************************************
                                        ;*TEST 7        TEST THAT THE Y REGISTER CAN HOLD A FLOATING 1 PATTERN
                                        ;;*****************************************************************
003004  000004                  TST7:   SCOPE
003006  012737  000100  001166  1$:     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
003014  012737  004000  001124          MOV     #BIT11,$GDDAT           ;LOAD EXPECTED
003022  013777  001124  176420          MOV     $GDDAT,@DAC1            ;LOAD THE REGISTER
003030  017737  176414  001126          MOV     @DAC1,$BDDAT            ;READ THE REGISTER
003036  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE DATA
003044  001401                          BEQ     2$              ;;BR IF DATA IS SAME
003046  104003                          ERROR   3                       ;ERROR, DAC1 REGISTER FAILED TO HOLD A FLOATING
003050  006237  001124          2$:     ASR     $GDDAT                  ;CHANGE THE DATA
003054  001362                          BNE     1$                      ;BR AND TEST MORE DATA
```

```
                                ;;*********************************************************
                                ;*TEST 10       TEST THAT THE DAC #2 REGISTER CAN BE CLEARED
                                ;;*********************************************************
003056  000004                  TST10:  SCOPE
003060  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED
003064  013777  001124  176360          MOV     $GDDAT,@DAC2            ;LOAD REG
003072  017737  176354  001126          MOV     @DAC2,$BDDAT            ;READ REG
003100  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003106  001401                          BEQ     TST11           ;;BR IF EQUAL
003110  104004                          ERROR   4                       ;ERROR, DAC #2 REGISTER NOT = 0

                                ;;*********************************************************
                                ;*TEST 11       TEST THAT THE DAC #2 REGISTER CAN BE LOADED WITH #7777
                                ;;*********************************************************
003112  000004                  TST11:  SCOPE
003114  012737  007777  001124          MOV     #7777,$GDDAT            ;LOAD EXPECTED
003122  013777  001124  176322          MOV     $GDDAT,@DAC2            ;LOAD REG
003130  017737  176316  001126          MOV     @DAC2,$BDDAT            ;READ REG
003136  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003144  001401                          BEQ     TST12           ;;BR IF EQUAL
003146  104004                          ERROR   4                       ;ERROR, DAC #2 REGISTER NOT = 7777

                                ;;*********************************************************
                                ;*TEST 12       TEST THAT THE DAC #2 REGISTER CAN HOLD A FLOATING 1 PATTERN
                                ;;*********************************************************
003150  000004                  TST12:  SCOPE
003152  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
003160  012737  004000  001124          MOV     #BIT11,$GDDAT           ;LOAD EXPECTED
003166  013777  001124  176256  1$:     MOV     $GDDAT,@DAC2            ;LOAD X REGISTER
003174  017737  176252  001126          MOV     @DAC2,$BDDAT            ;READ THE REGISTER
003202  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE DATA
003210  001401                          BEQ     2$              ;;BR IF SAME
003212  104004                          ERROR   4                       ;ERROR, DAC #2 REGISTER FAILED TO HOLD A FLOATIN
003214  006237  001124          2$:     ASR     $GDDAT                  ;CHANGE THE DATA
003220  001362                          BNE     1$                      ;BR AND TEST MORE DATA
```

```
                                ;;*****************************************************************
                                ;*TEST 13       TEST THAT THE DAC #3 REGISTER CAN BE CLEARED
                                ;;*****************************************************************
003222  000004                  TST13:  SCOPE
003224  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED
003230  013777  001124  176216          MOV     $GDDAT,@DAC3            ;LOAD REG
003236  017737  176212  001126          MOV     @DAC3,$BDDAT            ;READ REG
003244  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003252  001401                          BEQ     TST14           ;;BR IF EQUAL
003254  104005                          ERROR   5                       ;ERROR, DAC #3 REGISTER NOT = 0

                                ;;*****************************************************************
                                ;*TEST 14       TEST THAT THE DAC #3 REGISTER CAN BE LOADED WITH #7777
                                ;;*****************************************************************
003256  000004                  TST14:  SCOPE
003260  012737  007777  001124          MOV     #7777,$GDDAT            ;LOAD EXPECTED
003266  013777  001124  176160          MOV     $GDDAT,@DAC3            ;LOAD REG
003274  017737  176154  001126          MOV     @DAC3,$BDDAT            ;READ REG
003302  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003310  001401                          BEQ     TST15           ;;BR IF EQUAL
003312  104005                          ERROR   5                       ;ERROR, DAC #3 REGISTER NOT = 7777

                                ;;*****************************************************************
                                ;*TEST 15       TEST THAT THE DAC #3 REGISTER CAN HOLD A FLOATING 1 PATTERN
                                ;;*****************************************************************
003314  000004                  TST15:  SCOPE
003316  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
003324  012737  004000  001124          MOV     #BIT11,$GDDAT           ;LOAD EXPECTED
003332  013777  001124  176114  1$:     MOV     $GDDAT,@DAC3            ;LOAD DAC #3 REGISTER
003340  017737  176110  001126          MOV     @DAC3,$BDDAT            ;READ THE REGISTER
003346  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE DATA
003354  001401                          BEQ     2$              ;;BR IF SAME
003356  104005                          ERROR   5                       ;ERROR, DAC #3 REGISTER FAILED TO HOLD A FLOATIN
003360  006237  001124          2$:     ASR     $GDDAT                  ;CHANGE THE DATA
003364  001362                          BNE     1$                      ;BR AND TEST MORE DATA
```

```
392                                     ;;*****************************************************************
(3)                                     ;*TEST 16       TEST THAT THE FOUR DAC REGISTERS CAN HOLD DIFFERENT DATA
(3)                                     ;*****************************************************************
(2)  003366  000004                     TST16:  SCOPE
393  003370  012777  001111  176050             MOV     #1111,@DAC0             ;LOAD DAC #0
394  003376  012777  002222  176044             MOV     #2222,@DAC1             ;LOAD DAC #1
395  003404  012777  004444  176040             MOV     #4444,@DAC2             ;LOAD DAC #2
396  003412  012777  007777  176034             MOV     #7777,@DAC3             ;LOAD DAC #3
397  003420  005077  176020                     CLR     @STAT                   ;CLEAR STATUS
398  003424  012737  001111  001124             MOV     #1111,$GDDAT            ;LOAD EXPECTED
399  003432  017737  176010  001126             MOV     @DAC0,$BDDAT            ;READ REG
400  003440  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
401  003446  001401                             BEQ     1$              ;;BR IF EQUAL
402  003450  104002                             ERROR   2                       ;ERROR, SELECTED DAC0 IN ERROR
403
404  003452  012737  002222  001124  1$:       MOV     #2222,$GDDAT            ;LOAD EXPECTED
405  003460  017737  175764  001126             MOV     @DAC1,$BDDAT            ;READ REG
406  003466  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
407  003474  001401                             BEQ     2$              ;;BR IF EQUAL
408  003476  104003                             ERROR   3                       ;ERROR, SELECTED DAC1 IN ERROR
409
410  003500  012737  004444  001124  2$:       MOV     #4444,$GDDAT            ;LOAD EXPECTED
411  003506  017737  175740  001126             MOV     @DAC2,$BDDAT            ;READ REG
412  003514  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
413  003522  001401                             BEQ     3$              ;;BR IF SAME
414  003524  104004                             ERROR   4                       ;ERROR, SELECTED DAC #2 IN ERROR
415
416  003526  012737  007777  001124  3$:       MOV     #7777,$GDDAT            ;LOAD EXPECTED
417  003534  017737  175714  001126             MOV     @DAC3,$BDDAT            ;READ REG
418  003542  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
419  003550  001401                             BEQ     TST17           ;;BR IF SAME
420  003552  104005                             ERROR   5                       ;ERROR, SELECTED DAC #3 IN ERROR
```

```
                                ;;*****************************************************************
                                ;*TEST 17        TEST THAT RESET SETS READY BIT
                                ;;*****************************************************************
003554  000004                  TST17:  SCOPE
003556  012737  000010  001166          MOV     #10.,STIMES             ;;DO 10 ITERATIONS
003564  012737  000200  001124          MOV     #BIT7,@GDDAT                    ;LOAD EXPECTED
003572  000005                          RESET
003574  017737  175644  001126          MOV     @STAT,@BDDAT                    ;READ REGISTER
003602  023737  001124  001126          CMP     @GDDAT,@BDDAT                   ;COMPARE
003610  001401                          BEQ     TST20                   ;;BR IF SET
003612  104001                          ERROR   1                               ;RESET FAILED TO SET READY

                                ;;*****************************************************************
                                ;*TEST 20        TEST THAT FAST INTENSIFY CAN BE SET AND CLEARED
                                ;;*****************************************************************
003614  000004                  TST20:  SCOPE
003616  012777  000002  175620          MOV     #BIT1,@STAT                     ;LOAD BIT 1
003624  012737  000202  001124          MOV     #BIT7!BIT1,@GDDAT               ;LOAD EXPECTED
003632  017737  175606  001126          MOV     @STAT,@BDDAT                    ;READ THE REGISTER
003640  023737  001124  001126          CMP     @GDDAT,@BDDAT                   ;COMAPRE
003646  001401                          BEQ     1$                      ;;BR IF SAME
003650  104001                          ERROR   1                               ;ERROR, STATUS NOT = 202
003652  005077  175566          1$:     CLR     @STAT                           ;CLEAR STATUS REGISTER
003656  012737  000200  001124          MOV     #BIT7,@GDDAT                    ;LOAD EXPECTED
003664  017737  175554  001126          MOV     @STAT,@BDDAT                    ;READ THE REGISTER
003672  023737  001124  001126          CMP     @GDDAT,@BDDAT                   ;COMPARE
003700  001401                          BEQ     TST21                   ;;BR IF SAME
003702  104001                          ERROR   1                               ;ERROR, STATUS NOT = 200

                                ;;*****************************************************************
                                ;*TEST 21        TEST THAT MODE BIT 2 CAN BE SET AND CLEARED
                                ;;*****************************************************************
003704  000004                  TST21:  SCOPE
003706  012777  000004  175530          MOV     #BIT2,@STAT                     ;LOAD DISPLAY STATUS
003714  012737  000204  001124          MOV     #BIT7!BIT2,@GDDAT               ;LOAD EXPECTED
003722  017737  175516  001126          MOV     @STAT,@BDDAT                    ;READ REG
003730  023737  001124  001126          CMP     @GDDAT,@BDDAT                   ;COMPARE
003736  001401                          BEQ     1$                      ;;BR IF EQUAL
003740  104001                          ERROR   1                               ;ERROR, STATUS NOT = 204
003742  005077  175476          1$:     CLR     @STAT                           ;CLEAR STATUS
003746  012737  000200  001124          MOV     #BIT7,@GDDAT                    ;LOAD EXPECTED
003754  017737  175464  001126          MOV     @STAT,@BDDAT                    ;READ REG
003762  023737  001124  001126          CMP     @GDDAT,@BDDAT                   ;COMPARE
003770  001401                          BEQ     TST22                   ;;BR IF CLEARED
003772  104001                          ERROR   1                               ;MODE FAILED TO CLEAR
```

```
                                        ;;*****************************************************************
                                        ;*TEST 22        TEST THAT MODE BIT 3 CAN BE SET
                                        ;;*****************************************************************
(2)  003774  000004                     TST22:  SCOPE
461  003776  012777  000010  175440             MOV     #BIT3,@STAT             ;LOAD
462  004004  012737  000210  001124             MOV     #BIT7!BIT3,$GDDAT       ;LOAD EXPECTED
463  004012  017737  175426  001126             MOV     @STAT,$BDDAT            ;READ REG
464  004020  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
465  004026  001401                             BEQ     1$              ;;BR IF EQUAL
466  004030  104001                             ERROR   1                       ;ERROR, STATUS NOT = 210
467
468  004032  005077  175406             1$:    CLR     @STAT                   ;CLEAR REG.
469  004036  012737  000200  001124             MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
470  004044  017737  175374  001126             MOV     @STAT,$BDDAT            ;READ REG.
471  004052  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
472  004060  001401                             BEQ     TST23           ;;BR IF EQUAL
473  004062  104001                             ERROR   1                       ;ERROR, STATUS NOT = 200
474
475                                     ;;*****************************************************************
                                        ;*TEST 23        TEST THAT EXT. DELAY (BIT 4) CAN BE SET AND CLEARED
                                        ;;*****************************************************************
(2)  004064  000004                     TST23:  SCOPE
476  004066  012777  000020  175350             MOV     #BIT4,@STAT             ;LOAD STATUS
477  004074  012737  000220  001124             MOV     #BIT7!BIT4,$GDDAT       ;LOAD EXPCTED
478  004102  017737  175336  001126             MOV     @STAT,$BDDAT            ;READ REGISTER
479  004110  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
480  004116  001401                             BEQ     1$              ;;BR IF SAME
481  004120  104001                             ERROR   1                       ;ERROR, EXT. DELAY BIT FAILED TO SET
482  004122  005077  175316             1$:    CLR     @STAT                   ;CLEAR BIT 4
483  004126  012737  000200  001124             MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
484  004134  017737  175304  001126             MOV     @STAT,$BDDAT            ;READ REG.
485  004142  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
486  004150  001401                             BEQ     TST24           ;;BR IF SAME
487  004152  104001                             ERROR   1                       ;ERROR, EXT. DELAY BIT FAILED TO CLEAR
```

```
 489                                    ;;*****************************************************************
 (3)                                    ;*TEST 24       TEST THAT INTERRUPT ENABLE (BIT 6) CAN BE SET
 (3)                                    ;;*****************************************************************
 (2)  004154  000004                    TST24:  SCOPE
 490  004156  012777  000100  175260            MOV     #BIT6,@STAT             ;LOAD DISPLAY STATUS
 491  004164  012737  000300  001124            MOV     #BIT7!BIT6,$GDDAT       ;LOAD EXPECTED
 492  004172  017737  175246  001126            MOV     @STAT,$BDDAT            ;READ REG
 493  004200  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
 494  004206  001401                            BEQ     TST2.           ;;BR IF EQUAL
 495  004210  104001                            ERROR   1                       ;ERROR, STATUS NOT = 300
 496
 497                                    ;;*****************************************************************
 (3)                                    ;*TEST 25       TEST THAT CHANNEL (BIT 9) CAN BE SET
 (3)                                    ;;*****************************************************************
 (2)  004212  000004                    TST25:  SCOPE
 498  004214  012777  001000  175222            MOV     #BIT9,@STAT             ;LOAD DISPLAY STATUS
 499  004222  012737  001200  001124            MOV     #BIT9!BIT7,$GDDAT       ;LOAD EXPECTED
 500  004230  017737  175210  001126            MOV     @STAT,$BDDAT            ;READ REG
 501  004236  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
 502  004244  001401                            BEQ     TST26           ;;BR IF EQUAL
 503  004246  104001                            ERROR   1                       ;ERROR, STATUS NOT = 1200
 504
 505                                    ;;*****************************************************************
 (3)                                    ;*TEST 26       TEST THAT STORE (BIT 10) CAN BE SET
 (3)                                    ;;*****************************************************************
 (2)  004250  000004                    TST26:  SCOPE
 506  004252  012777  002000  175164            MOV     #BIT10,@STAT            ;LOAD DISPLAY STATUS
 507  004260  012737  002200  001124            MOV     #BIT10!BIT7,$GDDAT      ;LOAD EXPECTED
 508  004266  017737  175152  001126            MOV     @STAT,$BDDAT            ;READ REG
 509  004274  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
 510  004302  001401                            BEQ     TST27           ;;BR IF EQUAL
 511  004304  104001                            ERROR   1                       ;ERROR, STATUS NOT = 2200
 512
 513                                    ;;*****************************************************************
 (3)                                    ;*TEST 27       TEST THAT WRITE THRU (BIT 11) CAN BE SET
 (3)                                    ;;*****************************************************************
 (2)  004306  000004                    TST27:  SCOPE
 514  004310  012777  004000  175126            MOV     #BIT11,@STAT            ;LOAD DISPLAY STATUS
 515  004316  012737  004200  001124            MOV     #BIT11!BIT7,$GDDAT      ;LOAD EXPECTED
 516  004324  017737  175114  001126            MOV     @STAT,$BDDAT            ;READ REG
 517  004332  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
 518  004340  001401                            BEQ     TST30           ;;BR IF EQUAL
 519  004342  104001                            ERROR   1                       ;ERROR, STATUS NOT = 4200
 520
```

```
                                ;;*****************************************************************
                                ;*TEST 30       TEST THAT THE LOW BYTE OF THE STATUS REGISTER CAN BE CLEARED
                                ;*****************************************************************
004344  000004                  TST30:  SCOPE
004346  012777  005014  175070          MOV     #BIT11!BIT9!BIT3!BIT2,@STAT     ;LOAD THE STATUS REGISTER
004354  012737  005200  001124          MOV     #BIT11!BIT9!BIT7,$GDDAT         ;LOAD THE EXPECTED RESULT
004362  105077  175056                  CLRB    @STAT                           ;CLEAR LOW STATUS BYTE
004366  017737  175052  001126          MOV     @STAT,$BDDAT                    ;READ THE REGISTER
004374  023737  001124  001126          CMP     $GDDAT,$BDDAT                   ;COMAPRE
004402  001401                          BEQ     TST31                   ;;BR IF EXPECTED
004404  104001                          ERROR   1                               ;FAILED TO ADDRESS ONLY THE LOW BYTE OF STATUS R

                                ;;*****************************************************************
                                ;*TEST 31       TEST THAT THE HIGH BYTE OF THE STATUS REGISTER CAN BE CLEARED
                                ;*****************************************************************
004406  000004                  TST31:  SCOPE
004410  012777  005014  175026          MOV     #BIT11!BIT9!BIT3!BIT2,@STAT     ;LOAD THE STATUS REGISTER
004416  012737  000214  001124          MOV     #BIT7!BIT3!BIT2,$GDDAT  ;LOAD EXPECTED RESULTS
004424  013737  001444  010512          MOV     STAT,CHRCOL                     ;MAKE THE HIGH BYTE ADDRESS
004432  005237  010512                  INC     CHRCOL                          ;MAKE IT ODD
004436  105077  004050                  CLRB    @CHRCOL                         ;CLEAR THE HIGH BYTE
004442  017737  174776  001126          MOV     @STAT,$BDDAT                    ;RWEAD THE REGISTER
004450  023737  001124  001126          CMP     $GDDAT,$BDDAT                   ;COMPARE
004456  001401                          BEQ     TST32                   ;;BR IF EXPECTED
004460  104001                          ERROR   1                               ;FAILED TO CLEAR ONLY THE HIGH BYTE OF STATUS RE
```

```
                                ;;*****************************************************************
                                ;*TEST 32       TEST THAT WHEN INTENSIFY BIT IS SET THAT THE READY BIT CLEARS
                                ;;*****************************************************************
(2) 004462  000004              TST32:  SCOPE
                                ; AND THEN SETS AFTER A DELAY
004464  012700  001000                  MOV     #1000,R0
004470  005037  001124                  CLR     @GDDAT                  ;LOAD EXPECTED
004474  012777  000001  174742          MOV     #BIT0,@STAT             ;INTENSIFY
004502  017737  174736  001126          MOV     @STAT,$BDDAT            ;READ REG
004510  105737  001126                  TSTB    $BDDAT                  ;TEST READY
004514  100002                          BPL     1$              ;;BR IF NOT SET
004516  104011                          ERROR   11                      ;READY FAILED TO CLEAR
004520  000414                          BR      TST33           ;;BR TO SCOPE
004522  105777  174716          1$:     TSTB    @STAT                   ;TEST READY
004526  100411                          BMI     TST33           ;;NEXT TEST
004530  005300                          DEC     R0                      ;DELAY
004532  001373                          BNE     1$                      ;
004534  012737  000200  001124          MOV     #BIT7,@GDDAT            ;LOAD EXPECTED
004542  017737  174676  001126          MOV     @STAT,$BDDAT            ;READ REG
004550  104011                          ERROR   11                      ;READY FAILED TO SET AFTER A DELAY
                                                        ;IS STORAGE SCOPE CONNECTED BUT NOT TURNED ON ??
                                ;;*****************************************************************
                                ;*TEST 33       TEST THAT MODE 1 (INTENSIFY ON DACO) CLEARS THE READY FLAG
                                ;;*****************************************************************
(2) 004552  000004              TST33:  SCOPE
                                ; AND THEN SETS IT
004554  012700  001000                  MOV     #1000,R0                ;SET UP DELAY
004560  012737  000204  001124          MOV     #BIT7!BIT2,@GDDAT       ;LOAD EXPECTED
004566  012777  000004  174650          MOV     #BIT2,@STAT             ;LOAD MODE 1
004574  017737  174644  001126          MOV     @STAT,$BDDAT            ;READ REG
004602  105737  001126                  TSTB    $BDDAT                  ;TEST READY
004606  100402                          BMI     2$              ;;BR IF READY STILL SET
004610  104011                          ERROR   11                      ;ERROR, IN MODE 1 READY SHOULD NOT
                                                                        ; CLEAR UNTIL DACO IS LOADED
004612  000432                          BR      TST34           ;;BR TO SCOPE

004614  012737  000004  001124  2$:     MOV     #BIT2,@GDDAT            ;LOAD EXPECTED
004622  005077  174620                  CLR     @DACO                   ;ADDRESS DAC 0
004626  017737  174612  001126          MOV     @STAT,$BDDAT            ;READ REG
004634  105737  001126                  TSTB    $BDDAT                  ;TEST READY
004640  000240                          NOP
004642  100002                          BPL     1$              ;;BR IF CLEAR
004644  104011                          ERROR   11                      ;ERROR, MODE 1 LOAD DACO FAILED TO CLEAR READY F
004646  000414                          BR      TST34           ;;BR TO SCOPE

004650  105777  174570          1$:     TSTB    @STAT                   ;TEST READY
004654  100411                          BMI     TST34           ;;NEXT TEST
004656  005300                          DEC     R0                      ;DELAY
004660  001373                          BNE     1$              ;;TEST READY AGAIN
004662  012737  000204  001124          MOV     #BIT7!BIT2,@GDDAT       ;LOAD EXPECTED
004670  017737  174550  001126          MOV     @STAT,$BDDAT            ;READ REG
004676  104011                          ERROR   11                      ;ERROR, READY FAILED TO SET
                                                                        ; AFTER MODE 1 OPERATION
```

```
                                ;;*****************************************************************
                                ;*TEST 34       TEST THAT MODE 2 (INTENSIFY ON DAC1) CLEARS THE READY FLAG
                                ;*****************************************************************
004700  000004                  TST34:  SCOPE
                                ; AND THEN SETS IT
004702  012700  001000                  MOV     #1000,R0                ;SET UP DELAY
004706  012737  000210  001124          MOV     #BIT7!BIT3,$GDDAT       ;LOAD EXPECTED
004714  012777  000010  174522          MOV     #BIT3,@STAT             ;LOAD MODE 2
004722  017737  174516  001126          MOV     @STAT,$BDDAT            ;READ REG
004730  105737  001126                  TSTB    $BDDAT                  ;TEST READY
004734  100402                          BMI     2$              ;;BR IF SET
004736  104011                          ERROR   11                      ;ERROR, IN MODE 2 READY SHOULD NOT CLEAR
                                                                        ;UNTIL DAC1 IS LOADED
004740  000431                          BR      TST35           ;;BR TO SCOPE
004742  012737  000010  001124  2$:     MOV     #BIT3,$GDDAT            ;LOAD EXPECTED
004750  005077  174474                  CLR     @DAC1                   ;ADDRESS DAC1
004754  017737  174464  001126          MOV     @STAT,$BDDAT            ;READ REG
004762  105737  001126                  TSTB    $BDDAT                  ;TEST READY
004766  100002                          BPL     1$              ;;BR IF CLEARED
004770  104011                          ERROR   11                      ;ERROR, MODE 2 LOAD DAC1 FAILED TO CLEAR READY F
004772  000414                          BR      TST35           ;;BR TO SCOPE
004774  105777  174444          1$:     TSTB    @STAT                   ;TEST READY
005000  100411                          BMI     TST35           ;;NEXT TEST
005002  005300                          DEC     R0                      ;DELAY
005004  001373                          BNE     1$              ;;TEST READY AGAIN
005006  012737  000210  001124          MOV     #BIT7!BIT3,$GDDAT       ;LOAD EXPECTED
005014  017737  174424  001126          MOV     @STAT,$BDDAT            ;
005022  104011                          ERROR   11                      ;ERROR, READY FAILED TO SET
                                                                        ; AFTER MODE 2 OPERATION
                                ;;*****************************************************************
                                ;*TEST 35       TEST WHEN ERASE IS SET, READY BIT CLEARS AND SET AFTER DELAY    (SW12=1)
                                ;*****************************************************************
005024  000004                  TST35:  SCOPE
005026  012737  000001  001166          MOV     #1,$TIMES       ;;DO 1 ITERATION
005034  032777  010000  174076          BIT     #BIT12,@SWR             ;TEST BIT 12
005042  001430                          BEQ     TST36           ;;BYPASS IF NO STORAGE SCOPE
005044  012700  000010                  MOV     #10,R0
005050  005037  020432                  CLR     TEMP                    ;CLEAR DELAY
005054  012777  002000  174362          MOV     #BIT10,@STAT            ;SET STORE MODE
005062  052777  010000  174354          BIS     #BIT12,@STAT            ;SET ERASE BIT
005070  105777  174350                  TSTB    @STAT                   ;TEST THAT READY CLEARS
005074  100002                          BPL     1$              ;;BR IF CLEARED
005076  104011                          ERROR   11                      ;ERROR, READY FAILED TO RESET
005100  000411                          BR      TST36           ;;BR TO SCOPE
005102  105777  174336          1$:     TSTB    @STAT                   ;TEST FOR READY
005106  100406                          BMI     TST36           ;;BR IF SET
005110  005337  020432                  DEC     TEMP                    ;DELAY
005114  001372                          BNE     1$              ;;BR IF NOT READY
005116  005300                          DEC     R0                      ;DECREMENT COUNTER
005120  001370                          BNE     1$              ;;BR IF NOT DONE
005122  104011                          ERROR   11                      ;ERROR, ERASE CLEARED READY AND FAILED
                                                                        ;TO SET READY AFTER A DELAY
                                                        ;SWR 12 = 1 AND NO STORAGE SCOPE CONNECTED ??
```

```
639                                     ;;*****************************************************************
(3)                                     ;*TEST 36       TEST THAT THE DISPLAY DOES INTERRUPT AT LEVEL INDICATED -1
(3)                                     ;;*****************************************************************
(2)  005124  000004                     TST36:  SCOPE
(1)  005126  012737  000040  001166             MOV     #40,STIMES      ;;DO 40 ITERATIONS
640  005134  012737  000340  177776             MOV     #340,PSW
641  005142  012777  005204  174306             MOV     #1$,@IV         ;SET INTERRUPT VECTOR
642  005150  012700  000400                     MOV     #400,R0                 ;SET UP DELAY
643  005154  013737  020440  177776             MOV     BRLEV2,PSW              ;LOAD CPU PSW WITH DEVICE LEVEL -1
644  005162  012777  000101  174254             MOV     #BIT6!BIT0,@STAT        ;START DISPLAY
645  005170  005300                             DEC     R0                      ;DELAY
646  005172  001376                             BNE     .-2
647  005174  005077  174244                     CLR     @STAT                   ;DO NOT LET INTERRUPT ENABLE SET
648  005200  104006                             ERROR   6                       ;ERROR, FAILED TO INTERRUPT
649  005202  000401                             BR      2$              ;;NEXT TEST
650  005204  022626                     1$:     CMP     (SP)+,(SP)+             ;POP THE STACK
651  005206  013777  001460  174242     2$:     MOV     IVS,@IV         ;RESET VECTOR
652  005214  005077  174240                     CLR     @IVS
653
654                                     ;;*****************************************************************
(3)                                     ;*TEST 37       TEST THAT THE DISPLAY DOES NOT INTERRUPT AT LEVEL INDICATED
(3)                                     ;;*****************************************************************
(2)  005220  000004                     TST37:  SCOPE
(1)  005222  012737  000040  001166             MOV     #40,STIMES      ;;DO 40 ITERATIONS
655  005230  012737  000340  177776             MOV     #340,PSW
656  005236  012777  005272  174212             MOV     #1$,@IV         ;SET INTERRUPT VECTOR
657  005244  012700  004000                     MOV     #4000,R0                ;SET UP DELAY
658  005250  013737  020436  177776             MOV     BRLEV1,PSW              ;LOAD CPU PSW WITH DEVICE LEVEL
659  005256  012777  000101  174160             MOV     #BIT6!BIT0,@STAT        ;START DISPLAY
660  005264  005300                             DEC     R0                      ;DELAY
661  005266  001376                             BNE     .-2
662  005270  000404                             BR      2$              ;;
663  005272  005077  174146             1$:     CLR     @STAT                   ;DO NOT LET INTERRUPT ENABLE SET
664  005276  104007                             ERROR   7                       ;ERROR INTERRUPTED IN ERROR
665  005300  000417                             BR      TST40           ;;NEXT TEST
666  005302  012777  005324  174146     2$:     MOV     #3$,@IV         ;LOAD RETURN VECTOR
667  005310  005037  177776                     CLR     PSW                     ;LOWER PSW
668  005314  005077  174124                     CLR     @STAT                   ;CLEAR INT ENABLE
669  005320  104007                             ERROR   7                       ;LOWERING THE PRIORITY FAILED TO ALLOW INTERRUPT
670  005322  000401                             BR      4$              ;;NEXT TEST
671  005324  022626                     3$:     CMP     (SP)+,(SP)+
672  005326  013777  001460  174122     4$:     MOV     IVS,@IV         ;RESET VECTOR
673  005334  005077  174120                     CLR     @IVS
```

```
                                ;;*****************************************************************
                                ;*TEST 40        TEST THAT RESET CLEARS MODE, EXT. DELAY AND FAST INTENISIFY BITS
                                ;;*****************************************************************
005340  000004                  TST40:  SCOPE
005342  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
005350  012777  000036  174066          MOV     #BIT4!BIT3!BIT2!BIT1,@STAT
005356  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
005364  000005                        RESET
005366  017737  174052  001126          MOV     @STAT,$BDDAT            ;READ STATUS
005374  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
005402  001401                        BEQ     TST41           ;;BR IF EQUAL
005404  104001                        ERROR   1               ;ERROR, RESET FAILED TO CLEAR STATUS REG
                                ;;*****************************************************************
                                ;*TEST 41        TEST THAT RESET CLEARS INTERRUPT ENABLE, CHANNEL, STORE, WRITE THRU
                                ;;*****************************************************************
005406  000004                  TST41:  SCOPE
005410  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
005416  012777  007100  174020          MOV     #BIT11!BIT10!BIT9!BIT6,@STAT
005424  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
005432  000005                        RESET
005434  017737  174004  001126          MOV     @STAT,$BDDAT            ;READ STATUS
005442  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
005450  001401                        BEQ     TST42           ;;BR IF EQUAL
005452  104001                        ERROR   1               ;ERROR, RESET FAILED TO CLEAR STATUS
                                ;;*****************************************************************
                                ;*TEST 42        TEST THAT RESET CLEARS DACO REGISTER    (SW09=0)
                                ;;*****************************************************************
005454  000004                  TST42:  SCOPE
005456  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
005464  032777  001000  173446          BIT     #SW09,@SWR      ;TEST IF BIT 9 IS SET
005472  001016                        BNE     TST43           ;;BR IF SET
005474  012777  177777  173744          MOV     #-1,@DACO
005502  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED
005506  000005                        RESET
005510  017737  173732  001126          MOV     @DACO,$BDDAT            ;READ REG
005516  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
005524  001401                        BEQ     TST43           ;;BR IF EQUAL
005526  104002                        ERROR   2               ;ERROR, RESET FAILED TO CLEAR  DAC O REGISTER
                                ;;*****************************************************************
                                ;*TEST 43        TEST THAT RESET CLEARS DAC1 REGISTER    (SW09=0)
                                ;;*****************************************************************
005530  000004                  TST43:  SCOPE
005532  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
005540  032777  001000  173372          BIT     #SW09,@SWR      ;TEST IF BIT 9 IS SET
005546  001016                        BNE     TST44           ;;BR IF SET
005550  012777  177777  173672          MOV     #-1,@DAC1
005556  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED
005562  000005                        RESET
005564  017737  173660  001126          MOV     @DAC1,$BDDAT            ;READ REG
005572  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
005600  001401                        BEQ     TST44           ;;BR IF EQUAL
005602  104003                        ERROR   3               ;ERROR, RESET FAILED TO CLEAR DAC1 REGISTER
```

```
                                ;;*****************************************************************
                                ;*TEST 44        TEST THAT RESET CLEARS DAC #2 REGISTER  (SW09=0)
                                ;;*****************************************************************
005604  000004                  TST44:  SCOPE
005606  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
005614  032777  001000  173316          BIT     #SW09,@SWR      ;TEST IF BIT 9 IS SET
005622  001013                          BNE     TST45           ;;BR IF SET
005624  012777  177777  173620          MOV     #-1,@DAC2               ;LOAD THE REGISTER
005632  005037  001124                  CLR     $GDDAT                  ;CLEAR EXPECTED
005636  000005                          RESET
005640  017737  173606  001126          MOV     @DAC2,$BDDAT            ;READ THE REGISTER
005646  001401                          BEQ     TST45           ;;BR IF CLEARED
005650  104004                          ERROR   4                       ;ERROR, RESET FAILED TO CLEAR DAC #2
                                ;;*****************************************************************
                                ;*TEST 45        TEST THAT RESET CLEARS DAC #3 REGISTER  (SW09=0)
                                ;;*****************************************************************
005652  000004                  TST45:  SCOPE
005654  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
005662  032777  001000  173250          BIT     #SW09,@SWR      ;TEST IF BIT 9 IS SET
005670  001013                          BNE     TST46           ;;BR IF SET
005672  012777  177777  173554          MOV     #-1,@DAC3               ;LOAD THE REGISTER
005700  005037  001124                  CLR     $GDDAT                  ;CLEAR THE EXPECTED
005704  000005                          RESET
005706  017737  173542  001126          MOV     @DAC3,$BDDAT            ;READ THE REGISTER
005714  001401                          BEQ     TST46           ;;BR IF CLEARED
005716  104005                          ERROR   5                       ;ERROR, RESET FAILED TO CLEAR DAC #3
                                ;;*****************************************************************
                                ;*TEST 46        TEST THAT RESET SET DAC0 TO 4000        (SW09=1)
                                ;;*****************************************************************
005720  000004                  TST46:  SCOPE
005722  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
005730  032777  001000  173202          BIT     #SW09,@SWR              ;TEST BIT 9
005736  001417                          BEQ     TST47           ;;BR IF CLEARED
005740  012777  003777  173500          MOV     #3777,@DAC0             ;LOAD DAC0
005746  012737  004000  001124          MOV     #4000,$GDDAT            ;LOAD EXPTECTED
005754  000005                          RESET
005756  017737  173464  001126          MOV     @DAC0,$BDDAT            ;READ THE DAC0 REGISTER
005764  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
005772  001401                          BEQ     TST47           ;;BR IF SAME
005774  104002                          ERROR   2                       ;RESET FAILED TO SET DAC0 TO 4000
                                ;;*****************************************************************
                                ;*TEST 47        TEST THAT RESET SET DAC1 TO 4000        (SW09=1)
                                ;;*****************************************************************
005776  000004                  TST47:  SCOPE
006000  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
006006  032777  001000  173124          BIT     #SW09,@SWR              ;TEST BIT 9
006014  001417                          BEQ     TST50           ;;BR IF CLEARED
006016  012777  003777  173424          MOV     #3777,@DAC1             ;LOAD DAC1
006024  012737  004000  001124          MOV     #4000,$GDDAT            ;LOAD EXPTECTED
006032  000005                          RESET
006034  017737  173410  001126          MOV     @DAC1,$BDDAT            ;READ THE DAC1 REGISTER
006042  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
006050  001401                          BEQ     TST50           ;;BR IF SAME
006052  104003                          ERROR   3                       ;RESET FAILED TO SET DAC1 TO 4000
                                ;;*****************************************************************
                                ;*TEST 50        TEST THAT RESET SET DAC2 TO 4000        (SW09=1)
```

```
(4)
(3)  006054  000004                          ;;**********************************************************
                                              TST50:  SCOPE
(2)  006056  012737  000040  001166                   MOV     #40,$TIMES      ;;DO 40 ITERATIONS
(1)  006064  032777  001000  173046                   BIT     #SW09,@SWR              ;TEST BIT 9
(3)  006072  001417                                   BEQ     TST51           ;;BR IF CLEARED
(1)  006074  012777  003777  173350                   MOV     #3777,@DAC2             ;LOAD DAC2
(1)  006102  012737  004000  001124                   MOV     #4000,$GDDAT            ;LOAD EXPTECTED
(1)  006110  000005                                   RESET
(1)  006112  017737  173334  001126                   MOV     @DAC2,$EDDAT            ;READ THE DAC2 REGISTER
(1)  006120  023737  001124  001126                   CMP     $GDDAT,$BDDAT           ;COMPARE
(3)  006126  001401                                   BEQ     TST51           ;;BR IF SAME
(1)  006130  104004                                   ERROR   4                       ;RESET FAILED TO SET DAC2 TO 4000
741                                           ;;**********************************************************
(4)                                           ;*TEST 51       TEST THAT RESET SET DAC3 TO 4000        (SW09=1)
(4)                                           ;;**********************************************************
(3)  006132  000004                           TST51:  SCOPE
(2)  006134  012737  000040  001166                   MOV     #40,$TIMES      ;;DO 40 ITERATIONS
(1)  006142  032777  001000  172770                   BIT     #SW09,@SWR              ;TEST BIT 9
(3)  006150  001417                                   BEQ     TST52           ;;BR IF CLEARED
(1)  006152  012777  003777  173274                   MOV     #3777,@DAC3             ;LOAD DAC3
(1)  006160  012737  004000  001124                   MOV     #4000,$GDDAT            ;LOAD EXPTECTED
(1)  006166  000005                                   RESET
(1)  006170  017737  173260  001126                   MOV     @DAC3,$BDDAT            ;READ THE DAC3 REGISTER
(1)  006176  023737  001124  001126                   CMP     $GDDAT,$BDDAT           ;COMPARE
(3)  006204  001401                                   BEQ     TST52           ;;BR IF SAME
(1)  006206  104005                                   ERROR   5                       ;RESET FAILED TO SET DAC3 TO 4000
742                                           ;;**********************************************************
(3)                                           ;*TEST 52       TEST THAT EXTERNAL DELAY DOES NOT SET DISPLAY READY     SW10=0
(3)                                           ;;**********************************************************
(2)  006210  000004                           TST52:  SCOPE
(1)  006212  012737  000010  00.166                   MOV     #10,$TIMES      ;;DO 10 ITERATIONS
743  006220  032777  002000  172712                   BIT     #SW10,@SWR              ;TEST IF SW 10 IS SET
744  006226  001024                                   BNE     TST53           ;;BR IF EXT. DELAY SWITCH IS SET
745  006230  012777  000020  173206                   MOV     #BIT4,@STAT             ;LOAD EXT. DELAY ENABLE
746  006236  105277  173202                          INCB    @STAT                   ;START DISPLAY
747  006242  012700  000000                          MOV     #0,R0                   ;LOAD DELAY
748  006246  105777  173172               1$:         TSTB    @STAT                   ;WAIT FOR READY
749  006252  100403                                   BMI     2$                      ;BR IF SET
750  006254  005300                                   DEC     R0                      ;DELAY
751  006256  001373                                   BNE     1$                      ;BR IF NOT FINISHED
752  006260  000407                                   BR      TST53           ;;
753  006262  012737  000020  001124  2$:              MOV     #BIT4,$GDDAT            ;LOAD EXPECTED
754  006270  017737  173150  001126                   MOV     @STAT,$BDDAT            ;READ ACUTAL
755  006276  104011                                   ERROR   11                      ;EXT. DELAY CAUSED READY TO SET
756                                                                           ;WHEN THE OPERATOR (VIA SW10) SAID NO EXT. DELAY WAS CON
757
```

```
759                                     ;;*****************************************************************
(3)                                     ;*TEST 53       TEST THE EXTERNAL DELAY DOES SET DISPLAY READY  SW10=1
(3)                                     ;;*****************************************************************
(2) 006300  000004                      TST53:  SCOPE
(1) 006302  012737  000010  001166              MOV     #10,$TIMES      ;;DO 10 ITERATIONS
760 006310  032777  002000  172622              BIT     #SW10,@SWR              ;TEST SR BIT 10
761 006316  001424                              BEQ     TST54           ;;BR IF CLEARED
762 006320  012777  000020  173116              MOV     #BIT4,@STAT             ;ENABLE EXT. DELAY
763 006326  012700  000000                      MOV     #0,R0                   ;LOAD COUNTER
764 006332  105277  173106                      INCB    @STAT                   ;ENABLE DISPLAY
765 006336  105777  173102              1$:     TSTB    @STAT                   ;WAIT FOR READY
766 006342  100411                              BMI     2$                      ;BR IF DONE
767 006344  005300                              DEC     R0                      ;DELAY
768 006346  001373                              BNE     1$                      ;BR IF NOT DONE
769 006350  012737  000220  001124              MOV     #BIT7!BIT4,$GDDAT       ;LOAD EXPECTED VALUE
770 006356  017737  173062  001126              MOV     @STAT,$BDDAT            ;READ ACTUAL
771 006364  104011                              ERROR   11                      ;EXT. DELAY FAILED TO SET READY
772                                                                     ;WHEN THE OPERATOR (VIA SW 10=1) SAID AN EXT. DELAY WAS
773 006366  000005                      2$:     RESET                           ;ENSURE CLEARED AA11-K
774
775                                     ;;*****************************************************************
(3)                                     ;*TEST 54       DETERMINE IF MORE AA11-K'S REMAIN TO BE TESTED
(3)                                     ;;*****************************************************************
(2) 006370  000004                      TST54:  SCOPE
(1) 006372  012737  000001  001166              MOV     #1,$TIMES       ;;DO 1 ITERATION
776 006400  000005                              RESET
777 006402  005237  001206                      INC     $UNIT                   ;MORE UNITS?
778 006406  123737  001206  020442              CMPB    $UNIT,EVER
779 006414  001431                              BEQ     $EOP                ;;BR IF NONE
780 006416  063737  001436  001444              ADD     VADDR,STAT              ;UPDATE ADDRESS
781 006424  063737  001436  001446              ADD     VADDR,DAC0
782 006432  063737  001436  001450              ADD     VADDR,DAC1
783 006440  063737  001436  001452              ADD     VADDR,DAC2
784 006446  063737  001436  001454              ADD     VADDR,DAC3
785 006454  063737  001440  001456              ADD     VVECT,IV                ;UPDATE VECTOR
786 006462  063737  001440  001460              ADC     VVECT,IVS
787 006470  005037  001[??                      CLR     $TSTNM
788 006474  000137  002[??                      JMP     @TEST                   ;TEST ANOTHER AA11-K
```

```
                                .SBTTL  END OF PASS ROUTINE

                                ;;**********************************************************************
                                ;*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

006500                          $EOP:
006500  000004                          SCOPE
006502  005037  001102                   CLR     $TSTNM          ;;ZERO THE TEST NUMBER
006506  005037  001166                   CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
006512  005237  001202                   INC     $PASS           ;;INCREMENT THE PASS NUMBER
006516  042737  100000  001202           BIC     #100000,$PA_    ;;DON'T ALLOW A NEG. NUMBER
006524  005327                           DEC     (PC)+           ;;LOOP?
006526  000001          $EOPCT: .WORD   1
006530  003022                           BGT     $DOAGN          ;;YES
006532  012737                           MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
006534  000001          $ENOCT: .WORD   1
006536  006526                           $EOPCT
006540  104400  006605                   TYPE    ,$ENDMG         ;;TYPE "END PASS #"
006544  013746  001202                   MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
006550  104404                           TYPOS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
006552  104400  006602                   TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
006556  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
006562  001405                           BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
006564  000005                           RESET                   ;;CLEAR THE WORLD
006566  004710          $ENDAD: JSR     PC,(R0)                 ;;GO TO MONITOR
006570  000240                           NOP                     ;;SAVE ROOM
006572  000240                           NOP                     ;;FOR
006574  000240                           NOP                     ;;ACT11
006576                          $DOAGN:
006576  000137                           JMP     @(PC)+          ;;RETURN
006600  002324          $RTNAD: .WORD   MTEST
006602     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
006605     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
006612  050040  051501  020123
006620  000043
```

```
                                VISUAL: MOV     #STACK,SP               ;LOAD SP
                                        TST     EVER1                   ;TEST IF TYPED ONCE
                                        BNE     PIC0                    ;BR IF YES
                                        TYPE
                                        MES6                    ;HEADER ABOUT SCOPE ADJ.
                                        TYPE
                                        MES15                   ;TEXT ABOUT SWR
                                        INC     EVER1                   ;SET FLAG
                                        .SBTTL  DISPLAY HORIZONTAL LINE
                                PIC0:   MOV     DAC0,R0
                                        MOV     DAC1,R1
                                        BIT     #SW03,@SWR              ;TEST SR BIT 3
                                        BEQ     2$                      ;BR IF DAC #0 AND #1
                                        MOV     DAC2,R0         ;USE DAC #2 AND 3
                                        MOV     DAC3,R1
                        2$:     MOV     #30,TICKS               ;LOAD TIMER FOR CP TYPE
                                JSR     PC,CHTIME               ;CHANGE TIMER FOR CP TYPE
                        1$:     JSR     PC,PBB
                                JSR     PC,TIMER                ;DONE ?
                                BR      1$
                                .SBTTL  DISPLAY A VERTICAL LINE
                                PIC1:   MOV     DAC1,R0
                                        MOV     DAC0,R1
                                        BIT     #SW03,@SWR              ;TEST SR BIT 3
                                        BEQ     2$                      ;BR IF DAC #0 AND 1
                                        MOV     DAC3,R0         ;USE DAC #2 AND 3
                                        MOV     DAC2,R1
                        2$:     MOV     #30,TICKS               ;LOAD TIME
                                JSR     PC,CHTIME               ;CHANGE TIMER FOR CP TYPE
                        1$:     JSR     PC,PBB
                                JSR     PC,TIMER                ;DONE ?
                                BR      1$
                                BR      PIC3
                                PBB:    CLR     @STAT
                                        BIT     #SW10,@SWR              ;TEST IS SW10 =1
                                        BEQ     10$                     ;BR IF NOT
                                        BIS     #BIT4,@STAT             ;ENABLE EXT. DELAY
                                10$:    MOV     STAT,R4
                                        MOV     #7777,R3        ;SET HIGH LIMIT
                                        MOV     #1,R2           ;INITIALIZE INCREMENTS BETWEEN POINTS
                                        BIT     #SW09,@SWR              ;TEST SW 9 = 1
                                        BNE     2$                      ;BR IF YES
                                        MOV     #4000,(1)
                                        MOV     #0,(0)
                                1$:     ADD     R2,(0)          ;INCREMENT
                                        INC     (4)             ;INTENSIFY
                                        TSTB    (4)
                                        BPL     .-2
                                        CMP     (0),R3          ;DONE ALL POINTS?
                                        BNE     1$              ;NO
                                        RTS     PC
                                2$:     MOV     #0,(R1)                 ;LOAD TWO'S COMP. ORIGIN
                                        MOV     #0,(R0)
                                        BR      1$
```

```
                                        .SBTTL  PINCUSHION TEST (DISPLAY SQUARE)

007110  005077  172330          PIC3:   CLR     @STAT
007114  012737  000140  020430          MOV     #140,TICKS
007122  004737  014324                  JSR     PC,CHTIME
007126  013701  001446                  MOV     DAC0,R1
007132  013702  001450                  MOV     DAC1,R2
007136  032777  000010  171774          BIT     #SW03,@SWR      ;TEST SR BIT 3
007144  001404                          BEQ     1$              ;BR IF DAC #0 AND 1
007146  013701  001452                  MOV     DAC2,R1         ;USE DAC #2 AND 3
007152  013702  001454                  MOV     DAC3,R2
007156  013703  001444          1$:     MOV     STAT,R3
007162  012704  000020                  MOV     #20,R4
007166  032777  001000  171744  P3:     BIT     #SW09,@SWR      ;TEST BIT 9
007174  001405                          BEQ     1$
007176  012711  004000                  MOV     #4000,(R1)      ;LOAD TWO'S COMP ORIGIN
007202  012712  004000                  MOV     #4000,(R2)
007206  000402                          BR      2$
007210  005011          1$:             CLR     (R1)            ;CLEAR AXIS
007212  005012                          CLR     (R2)
                                        ;DRAW BOTTOM LINE
007214  012700  000377          2$:     MOV     #377,R0
007220  032777  002000  171712          BIT     #SW10,@SWR              ;TEST IF SW10 =1
007226  001403                          BEQ     P3A                     ;BR IF NOT
007230  052777  000020  172206          BIS     #BIT4,@STAT             ;ENABLE EXT. DELAY
007236  060411          P3A:            ADD     R4,(1)
007240  005213                          INC     (R3)            ;INTENSIFY
007242  105713                          TSTB    (3)
007244  100376                          BPL     .-2             ;WAIT FOR READY
007246  005300                          DEC     R0
007250  001372                          BNE     P3A             ;NO
                                ;DRAW RIGHT LINE
007252  012700  000377                  MOV     #377,R0
007256  060412          P3B:            ADD     R4,(2)
007260  005213                          INC     (R3)            ;INTENSIFY
007262  105713                          TSTB    (3)
007264  100376                          BPL     .-2             ;WAIT FOR READY
007266  005300                          DEC     R0
007270  001372                          BNE     P3B             ;NO
                                ;DRAW TOP LINE
007272  012700  000377                  MOV     #377,R0
007276  160411          P3C:            SUB     R4,(1)
007300  005213                          INC     (R3)            ;INTENSIFY
007302  105713                          TSTB    (3)
007304  100376                          BPL     .-2             ;WAIT FOR READY
007306  005300                          DEC     R0
007310  001372                          BNE     P3C             ;NO
                                ;DRAW LEFT LINE
007312  012700  000377                  MOV     #377,R0
007316  160412          P3D:            SUB     R4,(2)
007320  005213                          INC     (R3)            ;INTENSIFY
007322  105713                          TSTB    (3)
007324  100376                          BPL     .-2             ;WAIT FOR READY
007326  005300                          DEC     R0
007330  001372                          BNE     P3D             ;NO
007332  004737  014210                  JSR     PC,TIMER
```

```
                                        BR      P3
                                        .SBTTL  PLOT AN X

007340  005077  172100        PIC4:   CLR     @STAT
007344  012737  000100  020430        MOV     #100,TICKS
007352  004737  014324                JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
007356  013701  001446        PIC4B:  MOV     DAC0,R1
007362  013702  001450                MOV     DAC1,R2
007366  032777  000010  171544        BIT     #SW03,@SWR      ;TEST SWR BIT 3
007374  001404                        BEQ     1$              ;BR IF DAC #0 AND 1
007376  013701  001452                MOV     DAC2,R1         ;USE DAC #2 AND 3
007402  013702  001454                MOV     DAC3,R2
007406  013703  001444        1$:     MOV     STAT,R3
007412  012704  000004                MOV     #4,R4
007416  032777  001000  171514  P4:    BIT     #SW09,@SWR      ;TEST BIT 9
007424  001405                        BEQ     1$
007426  012712  004000                MOV     #4000,(R2)
007432  012711  004000                MOV     #4000,(R1)
007436  000402                        BR      2$
007440  005012        1$:     CLR     (R2)
007442  005011                        CLR     (R1)

                        ;PLOT LINE BEGINNING IN LOWER LEFT CORNER
007444  012700  001777        2$:     MOV     #1777,R0
007450  032777  002000  171462        BIT     #SW10,@SWR              ;TEST IF SW10 =1
007456  001403                        BEQ     P4A                     ;BR IF NOT
007460  052777  000020  171756        BIS     #BIT4,@STAT             ;ENABLE EXT. DELAY
007466  005213          P4A:  INC     (R3)            ;INTENSIFY
007470  105713                TSTB    (3)
007472  100376                BPL     .-2
007474  060412                ADD     R4,(2)          ;+4 TO Y
007476  060411                ADD     R4,(1)          ;+4 TO X
007500  005300                DEC     R0
007502  001371                BNE     P4A             ;NO
007504  105713                TSTB    (3)
007506  100376                BPL     .-2

                        ;PLOT LINE BEGINNING IN UPPER LEFT CORNER
007510  012712  007774        MOV     #7774,(2)
007514  005011                CLR     (1)
007516  012700  001777        MOV     #1777,R0
007522  005213          P4B:  INC     (R3)            ;INTENSIFY
007524  105713                TSTB    (3)
007526  100376                BPL     .-2
007530  160412                SUB     R4,(2)          ;-4 TO Y
007532  060411                ADD     R4,(1)          ;+4 TO X
007534  005300                DEC     R0
007536  001371                BNE     P4B             ;NO
007540  004737  014210        JSR     PC,TIMER
007544  000724                BR      P4
```

```
                                                .SBTTL  SCOPE SETTLING TIME TEST

007546  012737  000060  020430  PICS:   MOV     #60,TICKS       ;LOAD TIMER
007554  004737  014324                  JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE

007560  005077  171660          1$:     CLR     @STAT           ;CLEAR STATUS
007564  005077  171656                  CLR     @DAC0           ;CLEAR X AXIS
007570  005077  171654                  CLR     @DAC1           ;CLEAR Y AXIS
007574  005077  171652                  CLR     @DAC2
007600  005077  171650                  CLR     @DAC3
007604  032777  002000  171326          BIT     #SW10,@SWR      ;TEST SW BIT 10
007612  001403                          BEQ     10$             ;BR IF NOT
007614  052777  000020  171622          BIS     #BIT4,@STAT     ;ENABLE EXT. DELAY
007622  032777  000010  171310  10$:    BIT     #SW03,@SWR      ;TEST SR BIT 3
007630  001405                          BEQ     2$              ;BR IF DAC #0 AND 1
007632  013700  001454                  MOV     DAC3,R0         ;LOAD Y DAC
007636  013701  001452                  MOV     DAC2,R1         ;LOAD X DAC
007642  000404                          BR      3$
007644  013700  001450          2$:     MOV     DAC1,R0         ;LOAD Y DAC
007650  013701  001446                  MOV     DAC0,R1         ;LOAD X DAC
007654  032777  000200  171256  3$:     BIT     #SW07,@SWR      ;TEST X OR Y TEST BIT
007662  001403                          BEQ     4$              ;NORMAL
007664  010003                          MOV     R0,R3           ;REVERSE THE AXIS
007666  010100                          MOV     R1,R0
007670  010301                          MOV     R3,R1
007672  013703  001444          4$:     MOV     STAT,R3         ;LOAD STATUS REG
007676  032777  001000  171234  5$:     BIT     #SW09,@SWR      ;TEST TWO'S COMP SWITCH
007704  001007                          BNE     6$
007706  004537  007754                  JSR     R5,LODPNT       ;LOAD A LINE
007712  000000                          0
007714  004537  007754                  JSR     R5,LODPNT       ;LOAD A LINE
007720  007777                          7777
007722  000406                          BR      7$
007724  004537  007754          6$:     JSR     R5,LODPNT       ;LOAD A LINE
007730  004000                          4000
007732  004537  007754                  JSR     R5,LODPNT
007736  003777                          3777
007740  005710          7$:     TST     (R0)            ;END
007742  001355                  BNE     5$              ;BR IF NOT
007744  004737  014210                  JSR     PC,TIMER        ;TEST TIME
007750  000703                          BR      1$
007752  000414                          BR      PIC6

007754  012702  000200  LODPNT: MOV     #200,R2
007760  011511          1$:     MOV     (R5),(R1)       ;LOAD AXIS
007762  005213                  INC     (R3)            ;INTENSIFY
007764  105713          2$:     TSTB    (R3)            ;DONE
007766  100376                  BPL     2$              ;WAIT
007770  062710  000002          ADD     #2,(R0)         ;UPDATE AXIS
007774  005302                  DEC     R2              ;DONE
007776  001370                  BNE     1$              ;BR IF NOT
010000  005725                  TST     (R5)+           ;UPDATE
010002  000205                  RTS     R5              ;EXIT
```

```
                                                .SBTTL  PLOT CHARACTER SET

1032
1033
1034  010004  012737  000100  020430  PIC6:   MOV     #100,TICKS
1035  010012  004737  014324                  JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
1036  010016  013737  001446  010502          MOV     DAC0,VCXTMP     ;LOAD THE X AXIS
 (1)  010024  013737  001450  010504          MOV     DAC1,VCYTMP     ;LOAD THE Y AXIS
 (1)  010032  032777  000010  171100          BIT     #SW03,@SWR      ;TEST SR BIT 3
 (1)  010040  001406                          BEQ     1$              ;BR IF DAC #0 AND 1
 (1)  010042  013737  001452  010502          MOV     DAC2,VCXTMP     ;LOAD THE  X AXIS
 (1)  010050  013737  001454  010504          MOV     DAC3,VCYTMP
 (1)  010056  000240                  1$:     NOP
1037  010060  012737  001400  010510  PIC6A:  MOV     #1400,XPOS
1038  010066  012737  005400  010506          MOV     #5400,YPOS
1039  010074  032777  001000  171036          BIT     #SW09,@SWR      ;TEST SW 9
1040  010102  001406                          BEQ     1$
1041  010104  012737  005400  010510          MOV     #5400,XPOS      ;LOAD 2'S COMP ORIGIN
1042  010112  012737  001400  010506          MOV     #1400,YPOS
1043  010120  012702  010516          1$:     MOV     #A,R2           ;LOAD STARTING CHARACTER
1044  010124  005077  171314                  CLR     @STAT
1045  010130  004737  010276                  JSR     PC,PIC6B
1046  010134  012737  001400  010510          MOV     #1400,XPOS      ;LOAD X POS
1047  010142  012737  004000  010506          MOV     #4000,YPOS      ;LOAD Y POS
1048  010150  032777  001000  170762          BIT     #SW09,@SWR      ;TEST SW 9
1049  010156  001406                          BEQ     2$              ;BR IF 0
1050  010160  012737  005400  010510          MOV     #5400,XPOS      ;LOAD 2'S COMP ORIGIN
1051  010166  012737  000000  010506          MOV     #0,YPOS
1052  010174  012702  010617          2$:     MOV     #N,R2           ;LOAD STARTING CHARACTER
1053  010200  005077  171240                  CLR     @STAT
1054  010204  004737  010276                  JSR     PC,PIC6B        ;DISPLAY CHARACTERS
1055  010210  012737  001400  010510          MOV     #1400,XPOS      ;LOAD X POS
1056  010216  012737  002400  010506          MOV     #2400,YPOS      ;LOAD Y POS
1057  010224  032777  001000  170706          BIT     #SW09,@SWR      ;TEST SW 9
1058  010232  001406                          BEQ     3$
1059  010234  012737  005400  010510          MOV     #5400,XPOS
1060  010242  012737  006400  010506          MOV     #6400,YPOS      ;LOAD 2'S COMP ORIGIN
1061  010250  012702  010720          3$:     MOV     #NO,R2          ;LOAD STARTING CHARACTER
1062  010254  005077  171164                  CLR     @STAT
1063  010260  004737  010276                  JSR     PC,PIC6B        ;DISPLAY CHARACTERS
1064  010264  004737  014210                  JSR     PC,TIMER
1065  010270  000673                          BR      PIC6A
1066  010272  000137  011022                  JMP     PIC7
1067  010276  012737  177763  010512  PIC6B:  MOV     #-13.,CHRCOL    ;CHARACTERS PER ROW
1068  010304  013705  001444                  MOV     STAT,R5
1069  010310  004737  010324          GEN1:   JSR     PC,CHAR
1070  010314  005237  010512                  INC     CHRCOL          ;
1071  010320  001373                          BNE     GEN1
1072  010322  000207                          RTS     PC
1073
```

```
1075
1076                                        ;PLOT CHARACTER
1077  010324  013737  010506  010514  CHAR:   MOV     YPOS,YPT
1078  010332  042715  000016                  BIC     #16,(5)
1079  010336  013777  010510  000136          MOV     XPOS,@VCXTMP
1080  010344  013777  010506  000132          MOV     YPOS,@VCYTMP
1081  010352  032777  002000  170560          BIT     #SW10,@SWR      ;TEST SR BIT 10
1082  010360  001403                          BEQ     CHAR4           ;BR IF NOT
1083  010362  052777  000020  171054          BIS     #BIT4,@STAT     ;ENABLE EXT. DELAY
1084  010370  105777  171050          CHAR4:  TSTB    @STAT
1085  010374  100375                          BPL     CHAR4
1086  010376  052715  000002                  BIS     #BIT1,(5)       ;ENABLE FAST INTENSIFY ON LOADING Y
1087  010402  012704  000040                  MOV     #40,R4
1088  010406  012700  177773                  MOV     #-5,R0          ;INITIALIZE COLUMN COUNT
1089  010412  012701  177771          CHAR1:  MOV     #-7,R1          ;INITIALIZE ROW COUNT
1090  010416  112203                          MOVB    (2)+,R3         ;PUT CHARACTER POINTS IN R3
1091  010420  106103                  CHAR2:  ROLB    R3
1092  010422  100007                          BPL     CHAR3           ;NO
1093  010424  013777  010510  000050          MOV     XPOS,@VCXTMP
1094  010432  013777  010506  000044          MOV     YPOS,@VCYTMP
1095  010440  005215                          INC     (R5)            ;INTENSIFY
1096  010442  105715                  CHAR3:  TSTB    (R5)
1097  010444  100376                          BPL     CHAR3
1098  010446  060437  010506                  ADD     R4,YPOS
1099  010452  005201                          INC     R1              ;+1 TO ROW
1100  010454  001361                          BNE     CHAR2           ;FINISH ROW
1101  010456  013737  010514  010506          MOV     YPT,YPOS        ;REINITIALIZE ROW FOR NEXT COLUMN
1102  010464  060437  010510                  ADD     R4,XPOS
1103  010470  005200                          INC     R0              ;+1 TO COLUMN COUNT
1104  010472  001347                          BNE     CHAR1
1105  010474  060437  010510                  ADD     R4,XPOS
1106  010500  000207                          RTS     PC              ;EXIT
1107
1108  010502  000000          VCXTMP: 0
1109  010504  000000          VCYTMP: 0
1110  010506  000000          YPOS:   0       ;CONTAINS Y POSITION AT ANY GIVEN TIME
1111  010510  000000          XPOS:   0       ;CONTAINS X POSITION AT ANY GIVEN TIME
1112  010512  000000          CHRCOL: 0
1113  010514  000000          YPT:    0
1114
```

```
 1116
 1117  010516  176  021  021  A:   .BYTE 176,21,21,21,176
       010521  021  176
 1118  010523  177  111  111  B:   .BYTE 177,111,111,111,66
       010526  111  066
 1119  010530  076  101  101  C:   .BYTE 76,101,101,101,42
       010533  101  042
 1120  010535  177  101  101  D:   .BYTE 177,101,101,101,76
       010540  101  076
 1121  010542  177  111  111  E:   .BYTE 177,111,111,111,101
       010545  111  101
 1122  010547  177  011  011  F:   .BYTE 177,11,11,11,1
       010552  011  001
 1123  010554  076  101  121  G:   .BYTE 76,101,121,121,62
       010557  121  062
 1124  010561  177  010  010  H:   .BYTE 177,10,10,10,177
       010564  010  177
 1125  010566  000  101  177  I:   .BYTE 0,101,177,101,0
       010571  101  000
 1126  010573  060  100  100  J:   .BYTE 60,100,100,100,77
       010576  100  077
 1127  010600  177  010  024  K:   .BYTE 177,10,24,42,101
       010603  042  101
 1128  010605  177  100  100  L:   .BYTE 177,100,100,100,100
       010610  100  100
 1129  010612  177  004  010  M:   .BYTE 177,4,10,4,177
       010615  004  177
 1130  010617  177  004  010  N:   .BYTE 177,4,10,20,177
       010622  020  177
 1131  010624  076  101  101  O:   .BYTE 76,101,101,101,76
       010627  101  076
 1132  010631  177  011  011  P:   .BYTE 177,11,11,11,6
       010634  011  006
 1133  010636  076  101  121  Q:   .BYTE 76,101,121,141,176
       010641  141  176
 1134  010643  177  011  031  R:   .BYTE 177,11,31,51,106
       010646  051  106
 1135  010650  046  111  111  S:   .BYTE 46,111,111,111,62
       010653  111  062
 1136  010655  001  001  177  T:   .BYTE 1,1,177,1,1
       010660  001  001
 1137  010662  077  100  100  U:   .BYTE 77,100,100,100,77
       010665  100  077
 1138  010667  037  040  100  V:   .BYTE 37,40,100,40,37
       010672  040  037
 1139  010674  177  020  010  W:   .BYTE 177,20,10,20,177
       010677  020  177
 1140  010701  143  024  010  X:   .BYTE 143,24,10,24,143
       010704  024  143
 1141  010706  003  004  170  Y:   .BYTE 3,4,170,4,3
       010711  004  003
 1142  010713  141  121  111  Z:   .BYTE 141,121,111,105,103
       010716  105  103
 1143  010720  076  121  111  NO:  .BYTE   76,121,111,105,76
       010723  105  076
 1144  010725  000  102  177  N1:  .BYTE 0,102,177,100,0
```

```
                010730     100     000
1145            010732     142     121     111  N2:     .BYTE 142,121,111,105,102
                010735     105     102
1146            010737     042     101     111  N3:     .BYTE 42,101,111,111,66
                010742     111     066
1147            010744     030     024     022  N4:     .BYTE 30,24,22,177,20
                010747     177     020
1148            010751     047     105     105  N5:     .BYTE 47,105,105,105,71
                010754     105     071
1149            010756     076     111     111  N6:     .BYTE 76,111,111,111,62
                010761     111     062
1150            010763     101     041     021  N7:     .BYTE 101,41,21,11,7
                010766     011     007
1151            010770     066     111     111  N8:     .BYTE 66,111,111,111,66
                010773     111     066
1152            010775     046     111     111  N9:     .BYTE 46,111,111,111,76
                011000     111     076
1153            011002     000     000     000  SPACEA: .BYTE 0,0,0,0,0
                011005     000     000
1154            011007     000     000     000          .BYTE   0,0,0,0,0
                011012     000     000
1155            011014     000     000     000          .BYTE   0,0,0,0,0
                011017     000     000
1156
1157                    011022                           .EVEN
1158                                                     .SBTTL  CHANNEL 1 AND CHANNEL 2 TEST
```

```
1160  011022  012737  000200  020430  PIC7:   MOV     #200,TICKS      ;SET UP A TIMER
1161  011030  004737  014324                  JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
1162  011034  013737  001446  010502          MOV     DAC0,VCXTMP     ;LOAD THE X AXIS
 (1)  011042  013737  001450  010504          MOV     DAC1,VCYTMP     ;LOAD THE Y AXIS
 (1)  011050  032777  000010  170062          BIT     #SW03,@SWR      ;TEST SR BIT 3
 (1)  011056  001406                          BEQ     1$              ;BR IF DAC #0 AND 1
 (1)  011060  013737  001452  010502          MOV     DAC2,VCXTMP     ;LOAD THE  X AXIS
 (1)  011066  013737  001454  010504          MOV     DAC3,VCYTMP
 (1)  011074  000240                  1$:     NOP
1163  011076  013705  001444                  MOV     STAT,R5         ;SET UP R5 FOR STATUS REGISTER POINTER
1164  011102  012777  000000  170334  PIC7AA: MOV     #0,@STAT        ;SET UP SCOPE CONTROL
1165  011110  012737  002000  010510          MOV     #2000,XPOS      ;LOAD X POSITION
1166  011116  012737  005000  010506          MOV     #5000,YPOS      ;LOAD Y POSITION
1167  011124  032777  001000  170006          BIT     #SW09,@SWR      ;TEST BIT 9
1168  011132  001406                          BEQ     1$
1169  011134  012737  006000  010510          MOV     #6000,XPOS      ;LOAD 2'S COMP ORIGIN
1170  011142  012737  001000  010506          MOV     #1000,YPOS
1171  011150  012737  000011  020450  1$:     MOV     #9.,P7CNT       ;SAVE THE NUMBER OF CHARACTERS
1172  011156  012737  020454  020452          MOV     #CH01,P7PNT     ;SAVE CHANNEL 1 POINTER
1173  011164  017702  007262          PIC7A:  MOV     @P7PNT,R2       ;MOVE MESSAGE POINTER INTO R2 FOR DISPLAY ROUTINE
1174  011170  004737  010324                  JSR     PC,CHAR         ;DISPLAY A CHARACTER
1175  011174  062737  000002  020452          ADD     #2,P7PNT        ;ADD 2 TO THE MESSAGE POINTER
1176  011202  005337  020450                  DEC     P7CNT           ;DECREMENT CHARACTER COUNT
1177  011206  001366                          BNE     PIC7A           ;NOT FINISHED WITH ALL CHARACTERS
1178  011210  012777  001000  170226          MOV     #1000,@STAT
1179  011216  012737  002000  010510          MOV     #2000,XPOS      ;SET UP X POS FOR CHANNEL 2
1180  011224  012737  003000  010506          MOV     #3000,YPOS      ;SET UP Y
1181  011232  032777  001000  167700          BIT     #SW09,@SWR      ;TEST BIT 9
1182  011240  001406                          BEQ     1$
1183  011242  012737  006000  010510          MOV     #6000,XPOS      ;LOAD 2'S COMP ORIGIN
1184  011250  012737  007000  010506          MOV     #7000,YPOS
1185  011256  012737  000011  020450  1$:     MOV     #9.,P7CNT       ;SET UP CHARACTER COUNT
1186  011264  012737  020476  020452          MOV     #CH02,P7PNT     ;SET UP CHANNEL 2 POINTER
1187  011272  017702  007154          PIC7B:  MOV     @P7PNT,R2       ;SET UP
1188  011276  004737  010324                  JSR     PC,CHAR         ;DISPLAY A CHARACTER
1189  011302  062737  000002  020452          ADD     #2,P7PNT        ;ADD 2 TO THE POINTER
1190  011310  005337  020450                  DEC     P7CNT           ;DECREMENT COUNT
1191  011314  001366                          BNE     PIC7B           ;NOT FINISHED
1192  011316  004737  014210                  JSR     PC,TIMER        ;CHECK THE RUNTIME OF THIS ROUTINE
1193  011322  000667                          BR      PIC7AA          ;NOT FINISHED
1194  011324  000400                          BR      PIC12
```

```
                                                .SBTTL  PHOSPHOR TEST
                
011326  012737  000002  020430  PIC12:  MOV     #2,TICKS
011334  004737  013610          1$:     JSR     PC,CLRVC                ;ERASE THE SCREEN
011340  004737  013726                  JSR     PC,LOADVC               ;LOAD THE SCALE
011344  004737  014210                  JSR     PC,TIMER                ;CHECK THE TIME
011350  000771                          BR      1$
011352  004737  013610                  JSR     PC,CLRVC
011356  000005                          RESET
011360  000137  006650                  JMP     PIC0
                
                                                .SBTTL  MANUAL DISPLAY ROUTINE
                
011364  012706  001100          FULRMP: MOV     #STACK,SP               ;LOAD POINTER
011370  004737  001742                  JSR     PC,LDTRAP               ;LOAD BUS ADDRESS
011374  032777  002000  167536          BIT     #SW10,@SWR      ;TEST SR 10=1
011402  001403                          BEQ     1$              ;BR IF NOT
011404  052777  000020  170032          BIS     #BIT4,@STAT     ;ENABLE EXT. DELAY
011412  013700  001446          1$:     MOV     DAC0,R0         ;GET BUS ADDRESS
011416  004737  011454                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #1
011422  013700  001450                  MOV     DAC1,R0         ;GET BUS ADDRESS
011426  004737  011454                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #1
011432  013700  001452                  MOV     DAC2,R0         ;GET BUS ADDRESS
011436  004737  011454                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #2
011442  013700  001454                  MOV     DAC3,R0         ;GET THE BUS ADDRESS
011446  004737  011454                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #3
011452  000757                          BR      1$                      ;BR BACK
                
011454  005010          10$:    CLR     (R0)                    ;CLEAR DAC
011456  062710  000010  11$:    ADD     #10,(R0)                ;UPDATE THE DATA
011462  005710                  TST     (R0)                    ;TEST IF DONE
011464  001374                  BNE     11$                     ;BR IF NOT
011466  004737  013332                  JSR     PC,@CTRLC       ;TEST FOR CTRL C
011472  000207                          RTS     PC              ;EXIT
011474  000240                          NOP
011476  000240                          NOP
```

```
                                        .SBTTL  AUTO CALIBRATION
011500  012706  001100                AUTCAL: MOV     #STACK,SP               ;LOAD STACK POINTER
011504  004737  001742                        JSR     PC,LDTRAP               ;LOAD TRAN AND ADDRESSES
011510  104400                                TYPE
011512  017353                                AUTOCL                          ;TELL OPERATOR
011514  104400                                TYPE
011516  017061                                SELD01

                                        ;LOAD BUS TRAP FOR OPERTOR FALSE SELECTION

011520  012737  013150  000004                MOV     #AUTRAP,ERRVEC
011526  012737  000054  001102                MOV     #STN-1,$TSTNM           ;LOAD TEST NUMBER
011534  012737  011560  001110                MOV     #PSUPLY,$LPERR          ;LOAD LOOP RETURN
011542  012737  011560  001106                MOV     #PSUPLY,$LPADR          ;LOAD LOOP ADDRESS
                                                ;@@@@@@@@@@@@@@@@@@@@@@@@@@@@
011550  004537  014146                        JSR     R5,CROSS                ;CHANGE RELAYS
011554  020000                                BIT13                           ;USING BIT 13 FOR CRISS-CROSS WRAP MODE
                                                ;@@@@@@@@@@@@@@@@@@@@@@@@@@@@
                                        ;;**************************************************************
                                        ;*TEST 55       TEST THAT CH 3 IS AT +2.5 BIPOLAR
                                        ;;**************************************************************
000004                                  TST55:  SCOPE
011560  013737  014570  001124  PSUPLY: MOV     V1754,$GODAT            ;LOAD EXPECTED VALUE
011566  004537  014444                        JSR     R5,CONVRT               ;CONVERT
000003                                      CH3                             ;CH 3
011574  013737  014602  001126              MOV     ADENO,$BODAT            ;LOAD VALUE READ
011602  013737  014572  014610              MOV     V24,SPREAD              ;LOAD + OR - COUNT SPREAD
011610  004737  014612                        JSR     PC,COMPAR               ;COMPARE $GODAT AND $BODAT
011614  000402                                BR      NSUPLY          ;;BR IF WITHIN TOLERANCE
011616  104012                                ERROR   12                      ;CH 3 FAILED TO EQUAL EXPECTED VALUE

                                        ;;**************************************************************
                                        ;*TEST 56       TEST THAT CH 4 IS AT -2.5 BIPOLAR
                                        ;;**************************************************************
000004                                  TST56:  SCOPE
011622  013737  014572  001124  NSUPLY: MOV     V24,$GODAT              ;LOAD EXPECTED VALUE
011630  004537  014444                        JSR     R5,CONVRT               ;CONVERT
000004                                      CH4                             ;CH 4
011636  013737  014602  001126              MOV     ADENO,$BODAT            ;LOAD VALUE READ
011644  013737  014572  014610              MOV     V24,SPREAD              ;LOAD + OR - COUNT SPREAD
011652  004737  014612                        JSR     PC,COMPAR               ;COMPARE $GODAT AND $BODAT
011656  000402                                BR      SUPOK           ;;BR IF WITHIN TOLERANCE
011660  104012                                ERROR   12                      ;CH 4 FAILED TO EQUAL EXPECTED VALUE
011662  000422                                BR      TST57           ;;
011664                                  SUPOK:
011664  004537  014146                        JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
011670  020000                                BIT13
011672  104400  011700                        TYPE    ,65$            ;;TYPE ASCIZ STRING
011676  000414                                BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <15><12>/ANALOG SUPPLIES OK/<15><12>
011730                                  64$:
```

```
1320                                    ;;*****************************************************************
 (3)                                    ;*TEST 57       TEST THAT ERASE L (CH 53) IS GREATER THAN +3 VOLTS WHEN HIGH
 (3)                                    ;;*****************************************************************
 (2)  011730  000004                    TST57:  SCOPE
1321  011732  012777  000000  167504             MOV     #0,@STAT                ;LOAD STATUS
1322  011740  012737  001146  001124             MOV     #1146,$GDDAT            ;LOAD EXPECTED VALUE
1323  011746  004537  014444                     JSR     R5,CONVRT               ;CONVERT
1324  011752  000053                             CH53                            ;CH 53
1325  011754  013737  014602  001126             MOV     ADENO,$BDDAT            ;LOAD VALUE READ
1326  011762  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
1327  011770  003401                             BLE     TST60           ;;BR IF LESS THAN OR EQUAL
1328  011772  104014                             ERROR   14                      ;CH 53 FAILED TO EQUAL EXPECTED VALUE
1329
1330                                    ;;*****************************************************************
 (3)                                    ;*TEST 60       TEST THAT ERASE L (CH 53) IS LESS THAN +400 MVOLTS WHEN LOW
 (3)                                    ;;*****************************************************************
 (2)  011774  000004                    TST60:  SCOPE
1331  011776  005077  167442                     CLR     @STAT                   ;CLEAR STATUS
1332  012002  012777  012000  167434             MOV     #BIT12!BIT10,@STAT      ;SET ERASE
1333  012010  012737  000120  001124             MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
1334  012016  004537  014444                     JSR     R5,CONVRT               ;CONVERT
1335  012022  000053                             CH53                            ;CH 53
1336  012024  013737  014602  001126             MOV     ADENO,$BDDAT            ;LOAD VALUE READ
1337  012032  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
1338  012040  002001                             BGE     TST61           ;;BR IF GREATER THAN OR EQUAL
1339  012042  104015                             ERROR   15                      ;CH 53 FAILED TO EQUAL EXPECTED VALUE
1340
```

```
                                        ;;*******************************************************************
                                        ;*TEST 61        TEST THAT WRITE-THRU (CH 53) IS LESS THAN +400 MVOLTS WHEN LOW
                                        ;;*******************************************************************
                                        TST61:  SCOPE
1343  012046  012777  006000  167370            MOV     #BIT11!BIT10,@STAT      ;SET WRITE-THRU
1344  012054  012737  000120  001124            MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
1345  012062  004537  014444                    JSR     R5,CONVRT               ;CONVERT
1346  012066  000053                            CH53                            ;CH 54
1347  012070  013737  014602  001126            MOV     @DENO,$BDDAT            ;LOAD VALUE READ
1348  012076  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1349  012104  002001                            BGE     TST62           ;;BR IF GREATER THAN OR EQUAL
1350  012106  104015                            ERROR   15                      ;CH 54 FAILED TO EQUAL EXPECTED VALUE
                                        ;;*******************************************************************
                                        ;*TEST 62        TEST THAT NON-STORE (CH 55) IS LESS THAN +400 MVOLTS WHEN LOW
                                        ;;*******************************************************************
                                        TST62:  SCOPE
1352  012112  005077  167326                    CLR     @STAT                   ;CLEAR STORE MODE
1353  012116  012737  000120  001124            MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
1354  012124  004537  014444                    JSR     R5,CONVRT               ;CONVERT
1355  012130  000055                            CH55                            ;CH 55
1356  012132  013737  014602  001126            MOV     @DENO,$BDDAT            ;LOAD VALUE READ
1357  012140  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1358  012146  002001                            BGE     TST63           ;;BR IF GREATER THAN OR EQUAL
1359  012150  104015                            ERROR   15                      ;CH 53 FAILED TO EQUAL EXPECTED VALUE
                                        ;;*******************************************************************
                                        ;*TEST 63        TEST THAT NON-STORE L (CH 55) IS GREATER THAN +3 VOLTS WHEN HIGH
                                        ;;*******************************************************************
                                        TST63:  SCOPE
1361  012154  012777  017000  167262            MOV     #BIT12!BIT11!BIT10!BIT9,@STAT           ;SET STORE MODE
1362  012162  012737  001146  001124            MOV     #1146,$GDDAT            ;LOAD EXPECTED VALUE
1363  012170  004537  014444                    JSR     R5,CONVRT               ;CONVERT
1364  012174  000055                            CH55                            ;CH 55
1365  012176  013737  014602  001126            MOV     @DENO,$BDDAT            ;LOAD VALUE READ
1366  012204  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1367  012212  003401                            BLE     TST64           ;;BR IF LESS THAN OR EQUAL
1368  012214  104014                            ERROR   14                      ;CH 55 FAILED TO EQUAL EXPECTED VALUE
                                        ;;*******************************************************************
                                        ;*TEST 64        TEST THAT DELAY RETURN SETS READY
                                        ;;*******************************************************************
                                        TST64:  SCOPE
1370  012220  012777  000020  167216            MOV     #BIT4,@STAT             ;LOAD EXT DELAY BIT
1371  012226  012737  000020  001124            MOV     #BIT4,$GDDAT            ;LOAD EXPECTED
1372  012234  005277  167204                    INC     @STAT                   ;MAKE READY GO AWAY
1373  012240  017737  167200  001126            MOV     @STAT,$BDDAT            ;READ STAT
1374  012246  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1375  012254  001401                            BEQ     1$              ;;BR IF EXPECTED
1376  012256  104001                            ERROR   1                       ;READY FAILED TO CLEAR
1377  012260  052777  002000  167156  1$:       BIS     #BIT10,@STAT            ;LOAD NON-STORE L
1378  012266  012737  002220  001124            MOV     #BIT10!BIT7!BIT4,$GDDAT ;LOAD EXPECTED
1379  012274  017737  167144  001126            MOV     @STAT,$BDDAT            ;READ THE REGISTER
1380  012302  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1381  012310  001401                            BEQ     2$              ;;BR IF SAME
1382  012312  104001                            ERROR   1                       ;EXT. DELAY RETURN FAILED TO SET READY
1383  012314  005077  167124          2$:       CLR     @STAT
```

```
1385                                    ;;*************************************************************
 (3)                                    ;*TEST 65       TEST THAT CHANNEL 02 L (CH 56) IS GREATER THAN +3 VOLTS WHEN HIGH
 (3)                                    ;*************************************************************
 (2)  012320  000004                    TST65:  SCOPE
1386  012322  012777  014000  167114             MOV     #BIT12!BIT11,@STAT      ;LOAD STATUS
1387  012330  012737  001100  001124             MOV     #1100,$GDDAT            ;LOAD EXPECTED VALUE
1388  012336  004537  014444                     JSR     R5,CONVRT               ;CONVERT
1389  012342  000056                             CH56                            ;CH 56
1390  012344  013737  014602  001126             MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1391  012352  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
1392  012360  003401                             BLE     TST66           ;;BR IF LESS THAN OR EQUAL
1393  012362  104014                             ERROR   14                      ;CH 56 FAILED TO EQUAL EXPECTED
1394                                                                              ; VALUE
1395
1396                                    ;;*************************************************************
 (3)                                    ;*TEST 66       TEST THAT CHANNEL 02 L (CH 56) IS LESS THAN +400 MVOLTS WHEN LOW
 (3)                                    ;*************************************************************
 (2)  012364  000004                    TST66:  SCOPE
1397  012366  012777  003000  167050             MOV     #BIT9!BIT10             ;SET CHANNEL 2
1398  012374  012737  000120  001124             MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
1399  012402  004537  014444                     JSR     R5,CONVRT               ;CONVERT
1400  012406  000056                             CH56                            ;CH 56
1401  012410  013737  014602  001126             MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1402  012416  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
1403  012424  002001                             BGE     1$              ;;BR IF GREATER THAN
1404  012426  104015                             ERROR   15                      ;CH 56 FAILED TO EQUAL EXPECTED
1405                                                                              ; VALUE
1406  012430  005077  167010         1$:         CLR     @STAT                   ;CLEAR BIT 9
1407
```

```
1409                                    ;;*****************************************************************
 (3)                                    ;*TEST 67       TEST THAT "ERASE" AND DONE CAN BE CLEARED AND SET
 (3)                                    ;;*****************************************************************
 (2)  012434  000004                    TST67:  SCOPE
1410  012436  005077  167002                    CLR     @STAT                   ;CLEAR
1411  012442  005037  001124                    CLR     $GDDAT                  ;CLEAR EXPECTED
1412  012446  012777  010000  166770            MOV     #BIT12,@STAT            ;LOAD EXPECTED
1413  012454  017737  166764  001126            MOV     @STAT,$BDDAT            ;READ REG
1414  012462  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1415  012470  001401                            BEQ     1$              ;;BR IF SET
1416  012472  104001                            ERROR   1                       ;ERASE FAILED TO CLEAR READY, CHECK G5036-
1417                                                                    ;BCO8R CONNECTION: A-VV, VV-A
1418  012474  052777  001000  002076  1$:      BIS     #BIT9,@GSTAT            ;SET CH 2 (GENERATE ERASE RETURN ON GOOD)
1419  012502  005037  001124                    CLR     $GDDAT                  ;LOAD EXPECTED
1420  012506  017737  166732  001126            MOV     @STAT,$BDDAT            ;READ REG
1421  012514  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1422  012522  001401                            BEQ     2$              ;;BR IF CLEARED
1423  012524  104001                            ERROR   1                       ;ERASE BIT FAILED TO CLEAR BY ERASE
1424                                                                    ;RETURN (SETTING OF THE CH 2 BIT)
1425
1426  012526  042777  001000  002044  2$:      BIC     #BIT9,@GSTAT            ;CLEAR CH 2
1427  012534  012737  000200  001124            MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
1428  012542  013737  166676  001126            MOV     @STAT,$BDDAT            ;READ REG
1429  012550  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1430  012556  001401                            BEQ     3$              ;;BR IF EQUAL
1431  012560  104001                            ERROR   1                       ;READY FAILED TO SET ON THE END
1432                                                                    ; OF ERASE RETURN (CLEARING CH 2)
1433
1434  012562                         3$:
 (1)  012562  004537  014146                    JSR     R5,CROSS                ;LOAD THE RELAYS WITH 0
 (1)  012566  000000                            0
```

```
1436                                    ;;*****************************************************************
 (3)                                    ;*TEST 70       ADJUSTMENT ROUTINES FOR DAC 0 - 1
 (3)                                    ;;*****************************************************************
 (2)  012570  000004                    T$T70:  SCOPE
 (1)  012572  012737  000001  001166             MOV     #1,$TIMES       ;;DO 1 ITERATION
1437                                    ;OFFSET ADJUST FOR DAC 0
1438  012600  004537  014054                     JSR     R5,SNOVLT               ;LOAD THE VOLTAGE
1439  012604  020235                             N50000
1440  012606  005077  166634                     CLR     @DAC0                   ;LOAD DAC REGISTER
1441  012612  004537  014146                     JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
 (1)  012616  060000                             BIT14!BIT13
1442  012620  104400                             TYPE
1443  012622  017565                             R38                             ;TELL OPR. TO ADJUST R38
1444  012624  004737  013374                     JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1445
1446                                    ;OFFSET ADJUSTMENT FOR DAC 1
1447  012630  004537  014146                     JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
 (1)  012634  020000                             BIT13
1448  012636  005077  166606                     CLR     @DAC1                   ;LOAD THE DAC REGISTER
1449  012642  104400                             TYPE
1450  012644  017632                             R40                             ;TELL OPR. TO ADJUST R40
1451  012646  004737  013374                     JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1452
1453                                    ;GAIN ADJUST FOR DAC 0
1454  012652  004537  014054                     JSR     R5,SNOVLT
1455  012656  020250                             P49976
1456  012660  012777  007777  166560             MOV     #7777,@DAC0             ;LOAD DAC 0
1457  012666  004537  014146                     JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
 (1)  012672  060000                             BIT14!BIT13
1458  012674  104400                             TYPE
1459  012676  020011                             R37                             ;TELL OPR. TO ADJUST R37
1460  012700  004737  013374                     JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1461
1462                                    ;GAIN ADJUST FOR DAC 1
1463  012704  004537  014146                     JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
 (1)  012710  020000                             BIT13
1464  012712  012777  007777  166530             MOV     #7777,@DAC1             ;LOAD DAC 1
1465  012720  104400                             TYPE
1466  012722  020056                             R39                             ;TELL OPR. TO ADJUST R39
1467  012724  004737  013374                     JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1468
1469                                    ;DAC 0 D/A FUNCTION CHECK
1470  012730  004537  014352                     JSR     R5,FUNDAC               ;GO TO DAC FUNCTION SUB.
1471  012734  001446                             ^AC0
1472  012736  000051                             CH51
1473
1474                                    ;DAC 1 D/A FUNCTION CHECK
1475  012740  004537  014352                     JSR     R5,FUNDAC
1476  012744  001450                             DAC1
1477  012746  000052                             CH52
1478
```

```
1480                                    ;;*****************************************************************
 (3)                                    ;*TEST 71       ADJUSTMENT ROUTINES FOR DAC 2 - 3
 (3)                                    ;;*****************************************************************
 (2)  012750  000004                    TST71:  SCOPE
 (1)  012752  012737  000001  001166            MOV     #1,$TIMES       ;;DO 1 ITERATION
1481                                    ;NOW DO  DAC 2 AND 3
1482  012760  104400                            TYPE
1483  012762  017126                            SELD23
1484                                    ;OFFSET ADJUST TOR DAC 2
1485  012764  004537  014054                    JSR     R5,SNDVLT               ;LOAD E.D.C
1486  012770  020235                            N50000
1487  012772  005077  166454                    CLR     @DAC2                   ;LOAD DAC VALUE
1488  012776  004537  014146                    JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
 (1)  013002  060000                            BIT14!BIT13
1489
1490  013004  104400                            TYPE
1491  013006  000042                            42                              ;TELL OPR. TO ADJUST R42
1492  013010  004737  013374                    JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1493
1494                                    ;OFFSET ADJUST FOR DAC 3
1495  013014  005077  166434                    CLR     @DAC3
1496  013020  004537  014146                    JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
 (1)  013024  020000                            BIT13
1497  013026  005077  166422                    CLR     @DAC3                   ;LOAD THE DAC REGISTER
1498  013032  104400                            TYPE
1499  013034  017744                            R44                             ;TELL OPR. TO ADJUST R44
1500  013036  004737  013374                    JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1501
1502                                    ;GAIN ADJUST FOR DAC 2
1503  013042  004537  014054                    JSR     R5,SNDVLT
1504  013046  020250                            P49976
1505  013050  012777  007777  166374            MOV     #7777,@DAC2             ;LOAD DAC 2
1506  013056  004537  014146                    JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
 (1)  013062  060000                            BIT14!BIT13
1507
1508  013064  104400                            TYPE
1509  013066  020123                            R41                             ;TELL OPR. TO ADJUST R41
1510  013070  004737  013374                    JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1511
1512                                    ;GAIN ADJUST FOR DAC 3
1513  013074  012777  007777  166352            MOV     #7777,@DAC3             ;LOAD DAC 3
1514  013102  004537  014146                    JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
 (1)  013106  020000                            BIT13
1515
1516  013110  104400                            TYPE
1517  013112  020170                            R43                             ;TELL OPR. TO ADJUST R43
1518  013114  004737  013374                    JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1519
```

```
1521                                        ;DAC 2 D/A FUNCTION CHECK
1522
1523  013120  004537  014352                      JSR     R5,FUNDAC
1524  013124  001452                              DAC2
1525  013126  000051                              CHS1
1526
1527                                        ;DAC 3 D/A FUNCTION CHECK
1528
1529  013130  004537  014352                      JSR     R5,FUNDAC
1530  013134  001454                              DAC3
1531  013136  000052                              CHS2
1532
1533  013140  104400                              TYPE
1534  013142  017173                              CALDON
1535  013144  000137  011500                      JMP     AUTCAL
1536
1537  013150  022626              AUTRAP: CMP     (SP)+,(SP)+
1538  013152  012737  000006  000004              MOV     #6,ERRVEC
1539  013160  104400                              TYPE
1540  013162  017437                              AUTOER
1541  013164  000137  002312                      JMP     CTRLC
```

```
1543                                                    .SBTTL  MANUAL LOGIC TEST
1544
1545  013170  012706  001100          MANUL:  MOV     #STACK,SP       ;LOAD STACK
1546  013174  004737  001742                  JSR     PC,LDTRAP       ;LOAD BUS ADDRESS
1547  013200  104400                          TYPE
1548  013202  017232                          MANHED
1549  013204  004737  013332          1$:     JSR     PC,ACTRLC       ;TEST FOR CTRL C
1550  013210  104406                          CKSWR
1551  013212  017700  165722                  MOV     @SWR,R0         ;READ THE SWITCHES
1552  013216  010001                          MOV     R0,R1           ;COPY
1553  013220  042700  017777                  BIC     #17777,R0       ;MASK OFF BITS
1554  013224  001003                          BNE     2$              ;BR IF NOT DAC 0
1555  013226  013702  001446                  MOV     DAC0,R2         ;LOAD DAC 0 BUS ADDRESS
1556  013232  000424                          BR      10$
1557  013234  022700  020000          2$:     CMP     #BIT13,R0       ;TEST FOR DAC 1
1558  013240  001003                          BNE     3$              ;BR IF NOT
1559  013242  013702  001450                  MOV     DAC1,R2         ;LOAD DAC 1 BUS ADDRESS
1560  013246  000416                          BR      10$
1561  013250  022700  040000          3$:     CMP     #BIT14,R0       ;TEST FOR DAC 2
1562  013254  001003                          BNE     4$              ;BR IF NOT
1563  013256  013702  001452                  MOV     DAC2,R2         ;LOAD DAC 2 BUS ADDRESS
1564  013262  000410                          BR      10$
1565  013264  022700  060000          4$:     CMP     #BIT14!BIT13,R0 ;TEST FOR DAC 3
1566  013270  001003                          BNE     5$              ;BR IF NOT
1567  013272  013702  001454                  MOV     DAC3,R2         ;LOAD DAC 3 BUS ADDRESS
1568  013276  000402                          BR      10$
1569  013300  013702  001444          5$:     MOV     STAT,R2         ;LOAD STATUS REG. BUS ADDRESS
1570
1571  013304  010112                  10$:    MOV     R1,(R2)         ;LOAD THE SWITCHES INTO THE SELECTED ADDRESS
1572  013306  013703  001442                  MOV     DELAY,R3        ;PRESET THE DELAY
1573  013312  005303                  11$:    DEC     R3              ;DELAY
1574  013314  001376                          BNE     11$
1575  013316  005012                          CLR     (R2)            ;CLEAR THE REGISTER
1576  013320  013703  001442                  MOV     DELAY,R3        ;LOAD THE PRESET
1577  013324  005303                  12$:    DEC     R3              ;DELAY
1578  013326  001376                          BNE     12$
1579  013330  000725                          BR      1$
```

```
1581
1582                                    ;TEST FOR CTRL C  -- DONT WAIT
1583
1584  013332  105777  165606          ACTRLC: TSTB    @$TKS           ;INPUT FLAG ?
1585  013336  100014                          BPL     1$                      ;NO
1586  013340  017737  165602  013372          MOV     @$TKB,10$               ;READ THE CHARACTER
1587  013346  042737  177600  013372          BIC     #177600,10$
1588  013354  022737  000003  013372          CMP     #3,10$                  ;TEST FOR CTRL C
1589  013362  001002                          BNE     1$                      ;BR IF NOT
1590  013364  000137  002312                  JMP     CTRLC
1591  013370  000207                  1$:     RTS     PC                      ;EXIT
1592  013372  000000                  10$:    0
1593
1594                                    ;WAIT FOR A "SPACE" CHARACTER
1595
1596  013374  105777  165544          CSPACE: TSTB    @$TKS           ;WAIT FOR CHAR
1597  013400  100375                          BPL     CSPACE
1598  013402  017737  165540  013452          MOV     @$TKB,10$               ;READ CHAR
1599  013410  042737  177600  013452          BIC     #177600,10$             ;MASK
1600  013416  022737  000003  013452          CMP     #3,10$                  ;TEST FOR CTRL C
1601  013424  001002                          BNE     1$
1602  013426  000137  002312                  JMP     CTRLC
1603  013432  022737  000040  013452  1$:     CMP     #40,10$                 ;TEST FOR SPACE
1604  013440  001001                          BNE     2$
1605  013442  000207                          RTS     PC                      ;EXIT
1606  013444  104400                  2$:     TYPE
1607  013446  016774                          QMARK
1608  013450  000751                          BR      CSPACE
1609
1610  013452  000000                  10$:    0
```

```
1612
1613                                        .SBTTL  MANUAL DAC CALIBRATION
1614
1615  013454  012706  001100    CALDAC:   MOV     #STACK,SP           ;LOAD STACK POINTER
1616  013460  004737  001742              JSR     PC,LDTRAP           ;LOAD BUS ADDRESSES
1617  013464  104406              1$:     CKSWR                       ;TEST FOR CTRL G
1618  013466  004737  013332              JSR     PC,ACTRLC           ;TEST FOR CTRL C
1619  013472  017700  165442              MOV     @SWR,R0             ;READ SWITCHES
1620  013476  010077  165744              MOV     R0,@DAC0            ;LOAD DAC #0
1621  013502  010077  165742              MOV     R0,@DAC1            ;LOAD DAC #1
1622  013506  010077  165740              MOV     R0,@DAC2            ;LOAD DAC #2
1623  013512  010077  165736              MOV     R0,@DAC3            ;LOAD DAC #3
1624  013516  000762                      BR      1$
1625
1626                                        .SBTTL  DYNAMIC DAC CALIBRATION
1627
1628  013520  012706  001100    DYNCAL:   MOV     #STACK,SP           ;LOAD STACK POINTER
1629  013524  004737  001742              JSR     PC,LDTRAP           ;LOAD BUS ADDRESSES
1630  013530  104406              1$:     CKSWR                       ;TEST FOR CTRL G
1631  013532  004737  013332              JSR     PC,ACTRLC           ;TEST FOR CTRL C
1632  013536  017700  165376              MOV     @SWR,R0             ;READ SWR
1633  013542  004737  013556              JSR     PC,10$              ;LOAD THE SWR VALUE TO ALL DACS
1634  013546  005000                      CLR     R0                  ;CLEAR R0
1635  013550  004737  013556              JSR     PC,10$              ;LOAD ALL DAC'S WITH 0
1636  013554  000765                      BR      1$
1637
1638  013556  010077  165664    10$:      MOV     R0,@DAC0            ;LOAD DAC #0
1639  013562  010077  165662              MOV     R0,@DAC1            ;LOAD DAC #1
1640  013566  010077  165660              MOV     R0,@DAC2            ;LOAD DAC #2
1641  013572  010077  165656              MOV     R0,@DAC3            ;LOAD DAC #3
1642  013576  012700  000020              MOV     #20,R0              ;LOAD DELAY COUNTER
1643  013602  005300              11$:    DEC     R0                  ;DELAY
1644  013604  100376                      BPL     11$                 ;WAIT
1645  013606  000207                      RTS     PC                  ;EXIT
1646
1652
```

```
1654                                                    .SBTTL  SUBROUTINE TO ERASE STORAGE SCOPE SCREEN
1655  013610  104406                            CLRVC:  CKSWR
1656  013612  032777  000040  165320                    BIT     #BIT5,@SWR      ;TEST SR BIT 5
1657  013620  001441                                    BEQ     3$              ;BR IF NOT A STORAGE SCOPE
1658  013622  012777  002000  165614                    MOV     #BIT10,@STAT    ;ERASE THE SCREEN
1659  013630  052777  010000  165606                    BIS     #BIT12,@STAT
1660  013636  012700  000020                          MOV     #20,R0          ;SET UP DELAY
1661  013642  005001                                    CLR     R1
1662  013644  032777  000040  165266  1$:     BIT     #BIT5,@SWR      ;TEST IF NOT STORAGE SCOPE
1663  013652  001424                                    BEQ     3$
1664  013654  105777  165564                          TSTB    @STAT           ;TEST FOR READY
1665  013660  100421                                    BMI     3$              ;BRANCH IF SET
1666  013662  005301                                    DEC     R1              ;DELAY
1667  013664  001367                                    BNE     1$
1668  013666  004737  013332                          JSR     PC,@CTRLC       ;TEST FOR CTRL C
1669  013672  005300                                    DEC     R0              ;DELAY
1670  013674  001363                                    BNE     1$
1671  013676  037727  165236  010000                    BIT     @SWR,#SW12      ;TEST INHIBIT PRINTOUT
1672  013704  001002                                    BNE     2$
1673  013706  104400                                    TYPE
1674  013710  015412                                    MES3
1675  013712  104406                            2$:     CKSWR                   ;TEST FOR CTRL C
1676  013714  005777  165220                          TST     @SWR            ;TEST @SWR
1677  013720  100001                                    BPL     3$
1678  013722  000000                                    HALT                    ;ERASE RETURN FAILED TO SET READY
1679  013724  000207                            3$:     RTS     PC
1680
1681                                                    .SBTTL  SUBROUTINE TO DRAW A HORIZONTAL LINE
1682  013726  005077  165512                  LOADVC: CLR     @STAT           ;CLEAR STATUS
1683  013732  012737  007777  020434                    MOV     #7777,TEMP1
1684  013740  013700  001444                          MOV     STAT,R0
1685  013744  032777  000010  165166                    BIT     #SW03,@SWR      ;TEST SR BIT 3
1686  013752  001405                                    BEQ     4$              ;BR IF DAC #0 AND 1
1687  013754  013701  001452                          MOV     DAC2,R1 ;LOAD X AXIS
1688  013760  013702  001454                          MOV     DAC3,R2 ;LOAD Y AXIS
1689  013764  000404                                    BR      5$
1690  013766  013701  001446            4$:     MOV     DAC0,R1
1691  013772  013702  001450                          MOV     DAC1,R2
1692  013776  012710  002000            5$:     MOV     #BIT10,(0)      ;SET STORE MODE
1693  014002  013712  020434                          MOV     TEMP1,(2)
1694  014006  012711  007777            1$:     MOV     #7777,(1)
1695  014012  000402                                    BR      3$
1696  014014  162711  000010            2$:     SUB     #10,(1)
1697  014020  005210                    3$:     INC     (0)
1698  014022  105710                            TSTB    (0)
1699  014024  100376                            BPL     .-2
1700  014026  022711  000007                          CMP     #7,(1)
1701  014032  001370                                    BNE     2$
1702  014034  104406                                    CKSWR                   ;TEST FOR CTRL G
1703  014036  004737  013332                          JSR     PC,@CTRLC               ;TEST FOR CTRL C
1704  014042  162712  000010                          SUB     #10,(2)
1705  014046  100357                                    BPL     1$
1706  014050  000207                                    RTS     PC
1707
```

```
1709  014052  167772                  FILZ:   167772                  ;DR11-C OUTPUT CONTROL REGISTER
1710
1711                                  ;SUBROUTINE TO LOAD THE EDC VOLTAGE
1712                                  ;DATA FORMAT IS:        STX
1713                                  ;                       P OR N
1714                                  ;                       N VOLTS
1715                                  ;                       0 TENTHS VOLTS
1716                                  ;                       0
1717                                  ;                       0
1718                                  ;                       0
1719                                  ;                       0
1720                                  ;                       V       VOLTS
1721                                  ;                       ETX
1722
1723  014054  012500                  SNDVLT: MOV     (R5)+,R0                ;LOAD POINTER
1724  014056  112001                  2$:     MOVB    (R0)+,R1                ;GET A BYTE
1725  014060  001422                          BEQ     3$                      ;BR IF TERM.
1726  014062  005101                          COM     R1                      ;CONVERT IT
1727  014064  110177  177762                 MOVB    R1,@FILZ                ;LOAD FUNNY DATA BYTE
1728  014070  012701  001000                 MOV     #1000,R1                ;LOAD DELAY
1729  014074  005301                  5$:     DEC     R1                      ;DELAY
1730  014076  001376                          BNE     5$
1731  014100  042777  000200  177744         BIC     #BIT7,@FILZ             ;CLEAR BIT 7
1732  014106  012701  001000                 MOV     #1000,R1                ;LOAD DELAY COUNT
1733  014112  005301                  1$:     DEC     R1                      ;DELAY
1734  014114  001376                          BNE     1$
1735  014116  052777  000200  177726         BIS     #BIT7,@FILZ             ;SET BIT 7
1736  014124  000754                          BR      2$                      ;DO MORE DATA
1737
1738  014126  012701  000000                 MOV     #0,R1                   ;LOAD DELAY
1739  014132  152777  000177  177712         BISB    #177,@FILZ              ;SET BITS 0-6
1740  014140  005301                  4$:     DEC     R1                      ;DELAY
1741  014142  001376                          BNE     4$
1742  014144  000205                          RTS     R5                      ;EXIT
1743
1744  014146  012537  014206          CROSS:  MOV     (R5)+,10$       ;READ REG.
1745  014152  052737  000177  014206         BIS     #177,10$                ;LOAD LOW 7 BITS
1746  014160  013777  014206  177664         MOV     10$,@FILZ               ;LOAD RELAYS
1747  014166  012700  000001                 MOV     #1,R0                   ;LOAD DELA.
1748  014172  005001                          CLR     R1
1749  014174  005301                  1$:     DEC     R1                      ;DELAY
1750  014176  001376                          BNE     1$
1751  014200  005300                          DEC     R0                      ;DELAY
1752  014202  001374                          BNE     1$                      ;DONE ?
1753  014204  000205                          RTS     R5                      ;EXIT
1754  014206  000000                  10$:    0
```

```
                                                .SBTTL  TIMER ROUTINE FOR VISUAL TEST PATTERNS
                                        ; ENTER VIA JSR PC,TIMER
                                        
014210  104406                  TIMER:  CKSWR
014212  004737  013332                  JSR     PC,@CTRLC               ;TEST FOR CTRL C
014216  017737  164716  020426          MOV     @SWR,TIMSV
                                        
014224  032737  000400  020426  TIMERA: BIT     #BIT8,TIMSV
014232  001006                          BNE     TIMER2          ;BIT 8 SET ?
014234  005337  020430                  DEC     TICKS           ;NO, DECREMENT TICKS
014240  001002                          BNE     TIMER1
014242  062716  000002                  ADD     #2,(6)          ;ADD 2 TO STACK POINTER
014246  000207                  TIMER1: RTS     PC              ;RETURN
                                        
                                        ; SWR 8=1 SELECT TEST TO LOCK ON
                                        ; SWR 2-0= TEST NUMBER
                                        
014250  042737  177770  020426  TIMER2: BIC     #177770,TIMSV
014256  006337  020426                  ASL     TIMSV
014262  062737  014304  020426          ADD     #ROUTPT,TIMSV
014270  017737  004132  020426          MOV     @TIMSV,TIMSV
014276  022600                          CMP     (SP)+,R0
014300  000177  004122          TIMER4: JMP     @TIMSV
                                        
014304  006650                  ROUTPT: PIC0                    ;DISPLAY A HORIZONTAL LINE
014306  006724                          PIC1                    ;DISPLAY A VERTICAL LINE
014310  007110                          PIC3                    ;DISPLAY A SQUARE
014312  007340                          PIC4                    ;DISPALY A "X"
014314  007546                          PIC5                    ;DISPLAY SETTLING TIME
014316  010004                          PIC6                    ;DISPLAY CHARACTER SET
014320  011022                          PIC7                    ;DISPLAY CHANNEL TEST
014322  011326                          PIC12                   ;DISPLAY ERASE AND PHOSPHOR TEST
                                        
014324  013737  014350  020436  CHTIME: MOV     CPTYPE,BRLEV1
014332  005337  020436          1$:     DEC     BRLEV1
014336  001403                          BEQ     2$
014340  006337  020430                  ASL     TICKS
014344  000772                          BR      1$
014346  000207                  2$:     RTS     PC
                                        
014350  000001                  CPTYPE: 1
```

```
1798
1799                                    ;DO 32 CONVERSIONS ON EACH PRESET VALUE OF THE DAC'S
1800                                    ;AND SUBTRACT THE FIRST RESULT FROM EACH OF THE RESULTS
1801
1802  014352  013537  014442            FUNDAC: MOV     @(R5)+,XDACUT                   ;GET BUS ADDRESS OF ACUT DAC
1803  014356  012537  014402                    MOV     (R5)+,60$                       ;GET CHANNEL INPUT FOR ARKG
1804  014362  012777  007600  000052            MOV     #7600,@XDACUT                   ;PRESET THE DAC
1805  014370  012737  007600  001124            MOV     #7600,$GDDAT                    ;LOAD EXPECTED
1806
1807  014376  004537  014444            1$:     JSR     R5,CONVRT                       ;CONVERT USING ARKG
1808  014402  000051                    60$:    CHS1                                    ;
1809
1810  014404  012737  000010  014610            MOV     #10,SPREAD                      ;LOAD LIMIT
1811  014412  004737  014612                    JSR     PC,COMPAR                       ;TEST IF WITHIN LIMIT
1812  014416  000401                            BR      2$              ;;BR IF WITHIN
1813  014420  104016                            ERROR   16              ;SELECTED DAC HAS A LINEARITY ERROR
1814
1815  014422  162777  000400  000012  2$:     SUB     #400,@XDACUT                    ;UPDATE DAC
1816  014430  162737  000400  001124            SUB     #400,$GDDAT                     ;UPDATE GOOD VALUE
1817  014436  001357                            BNE     1$              ;;BR IF NOT FINISHED
1818
1819  014440  000000                    10$:    0
1820  014442  000000                    XDACUT: 0
```

```
1822                                        ;SUBROUTINE TO GET AVERAGE OF 32 CONVERSIONS FROM THE KNOWN GOOD AR 11
1823
1824  014444  012537  014564          CONVRT: MOV     (R5)+,10$
1825  014450  013737  014564  014606          MOV     10$,CHANL
1826  014456  000337  014564                  SWAB    10$
1827  014462  012737  000020  014566          MOV     #16.,11$
1828  014470  005037  014602                  CLR     ADEND
1829  014474  013777  014564  000072          MOV     10$,@GADCS
1830  014502  005277  000066          2$:     INC     @GADCS
1831  014506  105777  000062          1$:     TSTB    @GADCS
1832  014512  100375                          BPL     1$
1833  014514  067737  000056  014602          ADD     @GADDBR,ADEND
1834  014522  005337  014566                  DEC     11$
1835  014526  001365                          BNE     2$
1836  014530  006237  014602                  ASR     ADEND
1837  014534  006237  014602                  ASR     ADEND
1838  014540  006237  014602                  ASR     ADEND
1839  014544  006237  014602                  ASR     ADEND
1840  014550  005537  014602                  ADC     ADEND                   ;ROUND UP
1841  014554  013737  014602  001126          MOV     ADEND,$BDDAT
1842  014562  000205                          RTS     R5
1843
1844  014564  000000          10$:    0
1845  014566  000000          11$:    0
1846  014570  001754          V1754:  1754
1847  014572  000024          V24:    24
1848  014574  170460          GADCS:  170460
1849  014576  170462          GADDBR: 170462
1850  014600  170470          GSTAT:  170470
1851  014602  000000          ADEND:  0
1852  014604  000000          FCHANL: 0
1853  014606  000000          CHANL:  0
1854  014610  000000          SPREAD: 0
1855
1856  014612  010046          COMPAR: MOV     R0,-(SP)
1857  014614  010146                  MOV     R1,-(SP)
1858  014616  013700  001124          MOV     $GDDAT,R0               ;LOAD R0
1859  014622  013701  001126          MOV     $BDDAT,R1               ;LOAD R1
1860  014626  160100                  SUB     R1,R0                   ;SUBTRACT
1861  014630  100001                  BPL     8$                      ;
1862  014632  005400                  NEG     R0
1863  014634  020037  014610  8$:     CMP     R0,SPREAD               ;MAGNITUDE OF DIFF. IN R0
1864  014640  003405                  BLE     10$
1865  014642  012601          9$:     MOV     (SP)+,R1
1866  014644  012600                  MOV     (SP)+,R0
1867  014646  062716  000002          ADD     #2,(SP)
1868  014652  000207                  RTS     PC
1869
1870  014654  012601          10$:    MOV     (SP)+,R1
1871  014656  012600                  MOV     (SP)+,R0
1872  014660  000207                  RTS     PC
1873
```

```
                                                .SBTTL  ASCII MESSAGES
                                                
014662 005015 040412 030501  TITLE:  .ASCII  <15><12><12>'AA11-K DIAGNOSTIC TEST, (MAINDEC-11-DZAAC-A0)'<15><12>
014670 026461 020113 044504
014676 043501 047516 052123
014704 041511 052040 051505
014712 026124 024040 040515
014720 047111 042504 026503
014726 030461 042055 040532
014734 041501 040455 024460
014742 005015
014744 042523 042514 052103          .ASCII  /SELECT TESTS BY TYPING A TWO LETTER I.D./<15><12>
014752 052040 051505 051524
014760 041040 020131 054524
014766 044520 043516 040440
014774 052040 047527 046040
015002 052105 042524 020122
015010 027111 027104 005015
015016 046101 035011 052501          .ASCII  /AL     :AUTO LOGIC TEST/<15><12>
015024 047524 046040 043517
015032 041511 052040 051505
015040 006524    012
015043    101 004504 040472          .ASCII  /AD     :AUTO DISPLAY TEST (DISPLAY SCOPE CONNECTED)/<15><12>
015050 052125 020117 044504
015056 050123 040514 020131
015064 042524 052123 024040
015072 044504 050123 040514
015100 020131 041523 050117
015106 020105 047503 047116
015114 041505 042524 024504
015122 005015
015124 041501 035011 052501          .ASCII  /AC     :AUTO CALIBRATION (AA11-K OPTION TEST AREA ONLY)/<15><12>
015132 047524 041440 046101
015140 041111 040522 044524
015146 047117 024040 040501
015154 030461 045455 047440
015162 052120 047511 020116
015170 042524 052123 040440
015176 042522 020101 047117
015204 054514 006451    012
015211    115 004514 046402          .ASCII  /ML     :MANUAL LOGIC TEST (SWR 15-13 = REGISTER  12-0 = DATA)/<15><12>
015216 047101 040525 020104
015224 047514 044507 020103
015232 042524 052123 024040
015240 053523 020122 032461
015246 030455 020063 020075
015254 042522 044507 052123
015262 051105 020040 031061
015270 030055 036440 042040
015276 052101 024501 005015
015304 042115 035011 040515          .ASCII  /MD     :MANUAL DISPLAY (NO DISPLAY SCOPE)/<15><12>
015312 052516 046101 042040
015320 051511 046120 054501
015326 024040 047516 042040
015334 051511 046120 054501
015342 051440 047503 042520
```

```
                015350  006451      012
1884            015353      115  004503  046472          .ASCII  /MC     :MANUAL CALIBRATION LOOP/<15><12>
                015360  047101  040525  020114
                015366  040503  044514  051102
                015374  052101  047511  020116
                015402  047514  050117  005015
1885            015410  000056                          .ASCIZ  /./
1886            015412  005015  051105  051501  MES3:   .ASCIZ  <15><12>"ERASE RETURN FAILED TO SET READY"<15><12>
                015420  020105  042522  052524
                015426  047122  043040  044501
                015434  042514  020104  047524
                015442  051440  052105  051040
                015450  040505  054504  005015
                015456      000
1887            015457      015  051412  047503  MES6:   .ASCIZ  <15><12>'SCOPE ADJUSTMENT TEST'
                015464  042520  040440  045104
                015472  051525  046524  047105
                015500  020124  042524  052123
                015506      000
1888            015507      015  051412  051127  MES15:  .ASCII  <15><12>/SWR8 AND 0 THRU 2 CONTROL PATTERN/<15><12>
                015514  020070  047101  020104
                015522  020060  044124  052522
                015530  031040  041440  047117
                015536  051124  046117  050040
                015544  052101  042524  047122
                015552  005015
1889            015554  053523  020122  044502          .ASCII  'SWR BIT 3 SELECTS DAC 0+1 OR DAC 2+3'<15><12>
                015562  020124  020063  042523
                015570  042514  052103  020123
                015576  040504  020103  025460
                015604  020061  051117  042040
                015612  041501  031040  031453
                015620  005015
1890            015622  053523  020122  044502          .ASCII  /SWR BIT 5 SELECTS STORAGE SCOPE MODE/<15><12>
                015630  020124  020065  042523
                015636  042514  052103  020123
                015644  052123  051117  043501
                015652  020105  041523  050117
                015660  020105  047515  042504
                015666  005015
1891            015670  053523  020122  044502          .ASCII  /SWR BIT 7 ENABLES VERTICAL SETTLING TEST/<15><12>
                015676  020124  020067  047105
                015704  041101  042514  020123
                015712  042526  052122  041511
                015720  046101  051440  052105
                015726  046124  047111  020107
                015734  042524  052123  005015
1892            015742  053523  020122  044502          .ASCIZ  /SWR BIT 9 SELECTS TWO'S COMPLEMENT DISPLAY MODE/<15><12>
                015750  020124  020071  042523
                015756  042514  052103  020123
                015764  053524  023517  020123
                015772  047503  050115  042514
                016000  042515  052116  042040
                016006  051511  046120  054501
                016014  046440  042117  006505
                016022  000012
```

```
 1893  016024 052123 052101 051525 EM1:   .ASCIZ  /STATUS REGISTER IN ERROR/
       016032 051040 043505 051511
       016040 042524 020122 047111
       016046 042440 051122 051117
       016054    000
 1894  016055    104 041501 020060 EM2:   .ASCIZ  /DAC0 REGISTER IN ERROR/
       016062 042522 044507 052123
       016070 051105 044440 020116
       016076 051105 047522 000122
 1895  016104 040504 030503 051040 EM3:   .ASCIZ  /DAC1 REGISTER IN ERROR/
       016112 043505 051511 042524
       016120 020122 047111 042440
       016126 051122 051117    000
 1896  016133    104 041501 020062 EM4:   .ASCIZ  /DAC2 REGISTER IN ERROR/
       016140 042522 044507 052123
       016146 051105 044440 020116
       016154 051105 047522 000122
 1897  016162 040504 031503 051040 EM5:   .ASCIZ  /DAC3 REGISTER IN ERROR/
       016170 043505 051511 042524
       016176 020122 047111 042440
       016204 051122 051117    000
 1898  016211    101 030501 026461 EM6:   .ASCIZ  /AA11-K FAILED TO INTERRUPT/
       016216 020113 040506 046111
       016224 042105 052040 020117
       016232 047111 042524 051122
       016240 050125 000124
 1899  016244 040501 030461 045455 EM7:   .ASCIZ  /AA11-K INTERRUPTED IN ERROR/
       016252 044440 052116 051105
       016260 052522 052120 042105
       016266 044440 020116 051105
       016274 047522 000122
 1900  016300 052502 020123 044524 EM10:  .ASCIZ  /BUS TIME-OUT WHEN REFERENCING THE AA11-K/
       016306 042515 047455 052125
       016314 053440 042510 020116
       016322 042522 042506 042522
       016330 041516 047111 020107
       016336 044124 020105 040501
       016344 030461 045455    000
 1901  016351    101 030501 026461 EM11:  .ASCIZ  /AA11-K READY BIT ERROR/
       016356 020113 042522 042101
       016364 020131 044502 020124
       016372 051105 047522 000122
 1902  016400 047520 042527 020122 EM12:  .ASCIZ  /POWER SUPPLY VOLTAGE WAS INCORRECT/
       016406 052523 050120 054514
       016414 053040 046117 040524
       016422 042507 053440 051501
       016430 044440 041516 051117
       016436 042522 052103    000
 1903  016443    117 042516 041040 EM13:  .ASCIZ  /ONE BIT THE DUST/
       016450 052111 052040 042510
       016456 042040 051525 000124
 1904  016464 040501 030461 045455 EM14:  .ASCIZ  /AA11-K LOGIC SIGNAL HIGH OUTPUT TO LOW/
       016472 046040 043517 041511
       016500 051440 043511 040516
       016506 020114 044510 044107
       016514 047440 052125 052520
```

```
                016522  020124  047524  046040
                016530  053517     000
1905            016533     101  030501  026461  EM15:   .ASCIZ  /AA11-K LOGIC SIGNAL LOW OUTPUT TO HIGH/
                016540  020113  047514  044507
                016546  020103  044523  047107
                016554  046101  046040  053517
                016562  047440  052125  052520
                016570  020124  047524  044040
                016576  043511  000110
1906            016602  042523  042514  052103  EM16:   .ASCIZ  /SELECTED DAC HAS A LINEARITY ERROR/
                016610  042105  042040  041501
                016616  044040  051501  040440
                016624  046040  047111  040505
                016632  044522  054524  042440
                016640  051122  051117     000
1907            016645     105  051122  041520  DH1:    .ASCIZ  /ERRPC  IBASE   EXPECT    BAD/
                016652  044411  040502  042523
                016660  020040  042440  050130
                016666  041505  020124  020040
                016674  041040  042101     000
1908            016701     105  051122  041520  DH6:    .ASCIZ  /ERRPC  IBASE/
                016706  044411  040502  042523
                016714     000
1909            016715     105  051122  041520  DH13:   .ASCIZ  /ERRPC  IBASE   # FIRST # NOW/
                016722  044411  040502  042523
                016730  021411  043040  051111
                016736  052123  021411  047040
                016744  053517     000
1910            016747     105  051122  041520  DH14:   .ASCIZ  /ERRPC  IBASE   GOOD    BAD/
                016754  044411  040502  042523
                016762  043411  047517  004504
                016770  040502  000104
1911
1912            016774     015     012     077  QMARK:  .BYTE   15,12,77,15,12,0
                016777     015     012     000
1913            017002     136     103     015  CONTC:  .BYTE   136,103,15,12,56,0
                017005     012     056     000
1914            017010     015     012          FOUND1: .BYTE   15,12
1915            017012  051120  043517  040522          .ASCIZ  /PROGRAM DETECTED /
                017020  020115  042504  042524
                017026  052103  042105  000040
1916            017034  034050  020051  020040  FOUND2: .ASCIZ  , 8)   AA11-K(S)  /
                017042  040501  030461  045455
                017050  051450  020051  000040
1917            017056     015     012     000          .BYTE   15,12,0
1918            017061     015     012          SELD01: .BYTE   15,12
1919            017063     123  052105  051440          .ASCIZ  /SET SWITCH TO SELECT DAC 0 AND 1/<15><12>
                017070  044527  041524  020110
                017076  047524  051440  046105
                017104  041505  020124  040504
                017112  020103  020060  047101
                017120  020104  006461  000012
1920            017126     015     012          SELD23: .BYTE   15,12
1921            017130  042523  020124  053523          .ASCIZ  /SET SWITCH TO SELECT DAC 2 AND 3/<15><12>
                017136  052111  044103  052040
                017144  020117  042523  042514
```

```
                017152  052103  042040  041501
                017160  031040  040440  042116
                017166  031440  005015     000
1922            017173     015     012          CALDON: .BYTE   15,12
1923            017175     101  052125  020117          .ASCIZ  /AUTO CALIBRATION COMPLETED/<15><12>
                017202  040503  044514  051102
                017210  052101  047511  020116
                017216  047503  050115  042514
                017224  042524  006504  000012
1924            017232     015     012          MANHED: .BYTE   15,12
1925            017234  053523  030440  026465          .ASCII  /SW 15-13 SELECT THE REGISTER/<15><12>
                017242  031461  051440  046105
                017250  041505  020124  044124
                017256  020105  042522  044507
                017264  052123  051105  005015
1926            017272  053523  030440  026462          .ASCII  /SW 12-00 ARE LOADED INTO THE SELECTED REGISTER/<15><12>
                017300  030060  040440  042522
                017306  046040  040517  042504
                017314  020104  047111  047524
                017322  052040  042510  051440
                017330  046105  041505  042524
                017336  020104  042522  044507
                017344  052123  051105  005015
1927            017352     000                  .BYTE   0
1928            017353     015     012          AUTOCL: .BYTE   15,12
1929            017355     101  052125  020117          .ASCII  /AUTO CALIBRATION (AA11-K OPTION TEST AREA ONLY)/
                017362  040503  044514  051102
                017370  052101  047511  020116
                017376  040450  030501  026461
                017404  020113  050117  044524
                017412  047117  052040  051505
                017420  020124  051101  040505
                017426  047440  046116  024531
1930            017434     015     012     000          .BYTE   15,12,0
1931            017437     015  041012  051525  AUTOER: .ASCII  <15><12>/BUS TRAP WHEN LOADING VOLTAGE OR CHANGING A RELAY/<15><12>
                017444  052040  040522  020120
                017452  044127  047105  046040
                017460  040517  044504  043516
                017466  053040  046117  040524
                017474  042507  047440  020122
                017502  044103  047101  044507
                017510  043516  040440  051040
                017516  046105  054501  005015
1932            017524  037477  040411  030501          .ASCIZ  /??     AA11-K OPTION TEST AREA ??      /<15><12>
                017532  026461  020113  050117
                017540  044524  047117  052040
                017546  051505  020124  051101
                017554  040505  037440  004477
                017562  005015     000
1933
1937            017565     015  040412  045104  R38:    .ASCIZ  <15><12>/ADJUST R38 FOR A NULL ON THE METER/
 (1)            017572  051525  020124  031522
 (1)            017600  020070  047506  020122
 (1)            017606  020101  052516  046114
 (1)            017614  047440  020116  044124
 (1)            017622  020105  042515  042524
```

```
   (1)  017630  000122
1938    017632  005015  042101  052512  R40:    .ASCIZ  <15><12>/ADJUST R40 FOR A NULL ON THE METER/
   (1)  017640  052123  051040  030064
   (1)  017646  043040  051117  040440
   (1)  017654  047040  046125  020114
   (1)  017662  047117  052040  042510
   (1)  017670  046440  052105  051105
   (1)  017676     000
1939    017677     015  040412  045104  R22:    .ASCIZ  <15><12>/ADJUST R22 FOR A NULL ON THE METER/
   (1)  017704  051525  020124  031122
   (1)  017712  020062  047506  020122
   (1)  017720  020101  052516  046114
   (1)  017726  047440  020116  044124
   (1)  017734  020105  042515  042524
   (1)  017742  000122
1940    017744  005015  042101  052512  R44:    .ASCIZ  <15><12>/ADJUST R44 FOR A NULL ON THE METER/
   (1)  017752  052123  051040  032064
   (1)  017760  043040  051117  040440
   (1)  017766  047040  046125  020114
   (1)  017774  047117  052040  042510
   (1)  020002  046440  052105  051105
   (1)  020010     000
1941    020011     015  040412  045104  R37:    .ASCIZ  <15><12>/ADJUST R37 FOR A NULL ON THE METER/
   (1)  020016  051525  020124  031522
   (1)  020024  020067  047506  020122
   (1)  020032  020101  052516  046114
   (1)  020040  047440  020116  044124
   (1)  020046  020105  042515  042524
   (1)  020054  000122
1942    020056  005015  042101  052512  R39:    .ASCIZ  <15><12>/ADJUST R39 FOR A NULL ON THE METER/
   (1)  020064  052123  051040  034463
   (1)  020072  043040  051117  040440
   (1)  020100  047040  046125  020114
   (1)  020106  047117  052040  042510
   (1)  020114  046440  052105  051105
   (1)  020122     000
1943    020123     015  040412  045104  R41:    .ASCIZ  <15><12>/ADJUST R41 FOR A NULL ON THE METER/
   (1)  020130  051525  020124  032122
   (1)  020136  020061  047506  020122
   (1)  020144  020101  052516  046114
   (1)  020152  047440  020116  044124
   (1)  020160  020105  042515  042524
   (1)  020166  000122
1944    020170  005015  042101  052512  R43:    .ASCIZ  <15><12>/ADJUST R43 FOR A NULL ON THE METER/
   (1)  020176  052123  051040  031464
   (1)  020204  043040  051117  040440
   (1)  020212  047040  046125  020114
   (1)  020220  047117  052040  042510
   (1)  020226  046440  052105  051105
   (1)  020234     000
1945            000001                  STX=1
1946            000003                  ETX=3
1947    020235     001          N50000: .BYTE   STX
1948    020236  032516  030060  030060  .ASCII  /N500000V/
        020244  053060
1949    020246     003     000          .BYTE   ETX,0
```

```
1950  020250  001                       P49976: .BYTE   STX
1951  020251  120  034464  033471               .ASCII  /P499760V/
      020256  030066  126
1952  020261  003  000                          .BYTE   ETX,0
1953          020264                           .EVEN
1954  020264  001116  001444  001124  DT1:     $ERRPC,STAT,$GDDAT,$BDDAT,0
      020272  001126  000000
1955  020276  001116  001446  001124  DT2:     $ERRPC,DAC0,$GDDAT,$BDDAT,0
      020304  001126  000000
1956  020310  001116  001450  001124  DT3:     $ERRPC,DAC1,$GDDAT,$BDDAT,0
      020316  001126  000000
1957  020322  001116  001452  001124  DT4:     $ERRPC,DAC2,$GDDAT,$BDDAT,0
      020330  001126  000000
1958  020334  001116  001454  001124  DT5:     $ERRPC,DAC3,$GDDAT,$BDDAT,0
      020342  001126  000000
1959  020346  001116  001444  000000  DT6:     $ERRPC,STAT,0
1960  020354  001116  001444  020442  DT13:    $ERRPC,STAT,EVER,$UNIT,0
      020362  001206  000000
1961  020366  001116  001444  001124  DT14:    $ERRPC,STAT,$GDDAT,$BDDAT,0
      020374  001126  000000
1962  020400  000000  000000  000000  DF0:     0,0,0,0,0,0,0,0
      020406  000000  000000  000000
      020414  000000  000000
1963  020420  000000                    LOW:    0
1964  020422  000000                    HIGH:   0
1965  020424  000010                    INCR:   10
1966  020426  000000                    TIMSV:  0
1967  020430  000000                    TICKS:  0
1968  020432  000000                    TEMP:   0
1969  020434  000000                    TEMP1:  0                ;TEMPORARY STORAGE
1970  020436  000000                    BRLEV1: 0
1971  020440  000000                    BRLEV2: 0
1972  020442  000000                    EVER:   0
1973  020444  000000                    EVER1:  0
1974  020446  000000                    OPRIN:  0
1975  020450  000000                    P7CNT:  0
1976  020452  000000                    P7PNT:  0
1977  020454  010530                    CH01:   C
1978  020456  010561                            H
1979  020460  010516                            A
1980  020462  010617                            N
1981  020464  010617                            N
1982  020466  010542                            E
1983  020470  010605                            L
1984  020472  011002                            SPACEA
1985  020474  010725                            N1
1986  020476  010530                    CH02:   C
1987  020500  010561                            H
1988  020502  010516                            A
1989  020504  010617                            N
1990  020506  010617                            N
1991  020510  010542                            E
1992  020512  010605                            L
1993  020514  011002                            SPACEA
1994  020516  010732                            N2
1995
```

```
 1996                                   .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
  (1)
  (2)                                   ;;**************************************************************
  (1)                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
  (1)                                   ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
  (1)                                   ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
  (1)                                   ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
  (1)                                   ;*REPLACED WITH SPACES.
  (1)                                   ;*CALL:
  (1)                                   ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
  (1)                                   ;*      TYPDS                   ;;GO TO THE ROUTINE
  (1)
  (2)  020520                           $TYPDS:
  (3)  020520  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)  020522  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (3)  020524  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (3)  020526  010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (3)  020530  010546                           MOV     R5,-(SP)        ;;PUSH R5 ON STACK
  (1)  020532  012746  020200                   MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
  (1)  020536  016605  000020                   MOV     20(SP),R5       ;;GET THE INPUT NUMBER
  (1)  020542  100004                           BPL     1$              ;;BR IF INPUT IS POS.
  (1)  020544  005405                           NEG     R5              ;;MAKE THE BINARY NUMBER POS.
  (1)  020546  112766  000055  000001           MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
  (1)  020554  005000                   1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
  (1)  020556  012703  020734                   MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
  (1)  020562  112723  000040                   MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
  (1)  020566  005002                   2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
  (1)  020570  016001  020724                   MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
  (1)  020574  160105                   3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
  (1)  020576  002402                           BLT     4$              ;;BR IF DONE
  (1)  020600  005202                           INC     R2              ;;INCREASE THE BCD DIGIT BY 1
  (1)  020602  000774                           BR      3$
  (1)  020604  060105                   4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
  (1)  020606  005702                           TST     R2              ;;CHECK IF BCD DIGIT=0
  (1)  020610  001002                           BNE     5$              ;;FALL THROUGH IF 0
  (1)  020612  105716                           TSTB    (SP)            ;;STILL DOING LEADING 0'S?
  (1)  020614  100407                           BMI     7$              ;;BR IF YES
  (1)  020616  106316                   5$:     ASLB    (SP)            ;;MSD?
  (1)  020620  103003                           BCC     6$              ;;BR IF NO
  (1)  020622  116663  000001  177777           MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
  (1)  020630  052702  000060           6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
  (1)  020634  052702  000040           7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
  (1)  020640  110223                           MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
  (1)  020642  005720                           TST     (R0)+           ;;JUST INCREMENTING
  (1)  020644  020027  000010                   CMP     R0,#10          ;;CHECK THE TABLE INDEX
  (1)  020650  002746                           BLT     2$              ;;GO DO THE NEXT DIGIT
  (1)  020652  003002                           BGT     8$              ;;GO TO EXIT
  (1)  020654  010502                           MOV     R5,R2           ;;GET THE LSD
  (1)  020656  000764                           BR      6$              ;;GO CHANGE TO ASCII
  (1)  020660  105726                   8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
  (1)  020662  100003                           BPL     9$              ;;BR IF NO
  (1)  020664  116663  177777  177776           MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
  (1)  020672  105013                   9$:     CLRB    (R3)            ;;SET THE TERMINATOR
  (3)  020674  012605                           MOV     (SP)+,R5        ;;POP STACK INTO R5
  (3)  020676  012603                           MOV     (SP)+,R3        ;;POP STACK INTO R3
  (3)  020700  012602                           MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
   (3)  020702  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
   (3)  020704  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
   (1)  020706  104400  020734                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
   (1)  020712  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
   (1)  020720  012616                          MOV     (SP)+,(SP)
   (1)  020722  000002                          RTI                     ;;RETURN TO USER
   (1)  020724  023420                $DTBL:    10000.
   (1)  020726  001750                          1000.
   (1)  020730  000144                          100.
   (1)  020732  000012                          10.
   (1)  020734  000004                $DBLK:    .BLKW   4
  1997                                .SBTTL  SCOPE HANDLER ROUTINE
   (1)
   (2)                                ;;**********************************************************************
   (1)                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
   (1)                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
   (1)                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
   (1)                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
   (1)                                ;*SW14=1        LOOP ON TEST
   (1)                                ;*SW11=1        INHIBIT ITERATIONS
   (1)                                ;*SW08=1        LOOP ON TEST IN SWR<7:0>
   (1)                                ;*CALL
   (1)                                ;*      SCOPE           ;;SCOPE=IOT
   (1)
   (1)  020744                        $SCOPE:
   (1)  020744  104436                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
   (2)  020746  004737  013332                  JSR     PC,@CTRLC
   (1)  020752  032777  040000  160160  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
   (1)  020760  001070                          BNE     $OVER           ;;YES IF SW14=1
   (1)                                ;#####START OF CODE FOR THE XOR TESTER#####
   (1)  020762  000416                $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
   (1)                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
   (1)  020764  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
   (1)  020770  012737  021010  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
   (1)  020776  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
   (1)  021002  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
   (1)  021006  000446                          BR      $SVLAD          ;;GO TO THE NEXT TEST
   (1)  021010  022626                5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
   (1)  021012  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
   (1)  021016  000451                          BR      $OVER           ;;LOOP ON THE PRESENT TEST
   (1)  021020                        6$:;#####END OF CODE FOR THE XOR TESTER#####
   (1)  021020  032777  000400  160112          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
   (1)  021026  001404                          BEQ     2$              ;;BR IF NO
   (1)  021030  127737  160104  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
   (1)  021036  001441                          BEQ     $OVER           ;;BR IF YES
   (1)  021040  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
   (1)  021044  001404                          BEQ     3$              ;;BR IF NO
   (1)  021046  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
   (1)  021052  005037  001166                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
   (1)  021056  032777  004000  160054  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
   (1)  021064  001021                          BNE     1$              ;;BR IF YES
   (1)  021066  005737  001202                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
   (1)  021072  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
   (1)  021074  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
   (1)  021100  023737  001166  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
   (1)  021106  002015                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
```

```
                021110  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
                021116  013737  021156  001166          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
                021124  105237  001102                  INCB    $TSTNM          ;;COUNT TEST NUMBERS
                021130  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
                021136  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
                021142  016777  001102  157772  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
                021150  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
                021154  000002                          RTI                     ;;FIXES PS
                021156  003720                  $MXCNT: 3000.                   ;;MAX. NUMBER OF ITERATIONS
1998                                            .SBTTL  ERROR HANDLER ROUTINE

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

                021160                  $ERROR:
                021160  104406                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
                021162  004737  013332                  JSR     PC,@CTRLC
                021166  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
                021172  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
                021174  013777  001102  157740          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
                021202  005237  001112                  INC     $ERTTL          ;;INC THE ERROR COUNT
                021206  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
                021212  162737  000002  001116          SUB     #2,$ERRPC
                021220  117737  157672  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
                021226  032777  020000  157704          BIT     #BIT13,@$SWR    ;;SKIP TYPEOUT IF SET
                021234  001004                          BNE     20$             ;;SKIP TYPEOUTS
                021236  004737  021310                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
                021242  104400  001171                  TYPE    ,$CRLF
                021246                          20$:
                021246  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
                021254  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
                021256  113737  001114  021270          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
                021264  004737  023270                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
                021270      000                 21$:    .BYTE   0
                021271      003                         .BYTE   3
                021272  000777                  22$:    BR      22$             ;;APT ERROR LOOP
                021274  005777  157640          2$:     TST     @$SWR           ;;HALT ON ERROR
                021300  100002                          BPL     3$              ;;SKIP IF CONTINUE
                021302  000000                          HALT                    ;;HALT ON ERROR!
                021304  104406                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
                021306                          3$:
                021306  000002                          RTI                     ;;RETURN
1999                                            .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

                                                ;;**********************************************************************
                                                ;*THIS POUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
```

```
(1)  021310                          $ERRTYP:
(1)  021310  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021314  010046                          MOV     R0,-(SP)        ;;SAVE R0
(1)  021316  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
(1)  021320  153700  001114                  BISB    @#$ITEMB,R0
(1)  021324  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                     ;;TYPE THE PC OF THE ERROR
(2)  021326  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                  ;;ERROR ADDRESS
(2)  021332  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  021334  000426                          BR      6$              ;;GET OUT
(1)  021336  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(1)  021340  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
(1)  021342  006300                          ASL     R0
(1)  021344  006300                          ASL     R0
(1)  021346  062700  001256                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
(1)  021352  012037  021362                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
(1)  021356  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(1)  021360  104400                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
(1)  021362  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
(1)  021364  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021370  012037  021400          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
(1)  021374  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
(1)  021376  104400                          TYPE                    ;;TYPE THE "DATA HEADER"
(1)  021400  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
(1)  021402  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021406  011000                  5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
(1)  021410  001004                          BNE     7$              ;;GO TYPE THE DATA
(1)  021412  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
(1)  021414  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021420  000207                          RTS     PC              ;;RETURN
(2)  021422                          7$:
(2)  021422  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
(2)  021424  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  021426  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
(1)  021430  001770                          BEQ     6$              ;;BR IF NO
(1)  021432  104400  021440                  TYPE    ,8$             ;;TYPE TWO(2) SPACES
(1)  021436  000771                          BR      7$              ;;LOOP
(1)  021440  020040      000        8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
(1)          021444                          .EVEN
2006                                 .SBTTL  POWER DOWN AND UP ROUTINES
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;POWER DOWN ROUTINE
(1)  021444  012737  02:610  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
(1)  021452  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
(3)  021460  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  021462  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  021464  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  021466  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  021470  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
(3)  021472  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(3)  021474  017746  157440                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
(1)  021500  010637  021614                  MOV     SP,$SAVR6       ;;SAVE SP
(1)  021504  012737  021516  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
(1)  021512  000000                          HALT
```

```
    (1)  021514  000776                          BR      .-2             ;;HANG UP
    (1)
    (2)                                 ;;*****************************************************************
    (1)                                 ;POWER UP ROUTINE
    (1)  021516  012737  021610  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
    (1)  021524  013706  021614          $SAVR6: MOV     $SAVR6,SP       ;;GET SP
    (1)  021530  005037  021614                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
    (1)  021534  005237  021614          1$:     INC     $SAVR5          ;;WAIT FOR THE INC
    (1)  021540  001375                          BNE     1$              ;;OF THE WORD
    (3)  021542  012677  157372                  MOV     (SP)+,@#SWR     ;;POP STACK INTO @#SWR
    (3)  021546  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
    (3)  021550  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
    (3)  021552  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
    (3)  021554  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
    (3)  021556  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
    (3)  021560  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
    (1)  021562  012737  021444  000024          MOV     #$PWRON,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
    (1)  021570  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
    (1)  021576  104400                          TYPE                    ;;REPORT THE POWER FAILURE
    (1)  021600  021616                  $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
    (1)  021602  012716                          MOV     (PC)+,(SP)      ;;RESTART AT BEGIN
    (1)  021604  001462                  $PWRAD: .WORD   BEGIN           ;;RESTART ADDRESS
    (1)  021606  000002                          RTI
    (1)  021610  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
    (1)  021612  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
    (1)  021614  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
 2001    021616  005015  042522  052123  PWRMSG: .ASCIZ  <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12><12>
         021624  051101  044524  043516
         021632  040440  052106  051105
         021640  040440  050040  053517
         021646  051105  043040  044501
         021654  052514  042522  005015
         021662  000012
 2002                                .EVEN
 2003
 2004                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
  (1)
  (2)                                ;;*****************************************************************
  (1)                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
  (1)                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
  (1)                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
  (1)                                ;*CALL:
  (1)                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
  (1)                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
  (1)                                ;*      .BYTE   M               ;;M=1 OR 0
  (1)                                ;*                                      ;;1=TYPE LEADING ZEROS
  (1)                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
  (1)                                ;*
  (1)                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
  (1)                                ;*$TYPOS OR $TYPOC
  (1)                                ;*CALL:
  (1)                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                                ;*      TYPON                   ;;CALL FOR TYPEOUT
  (1)                                ;*
  (1)                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
```

```
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPOC                   ;;CALL FOR TYPEOUT
(1)
(1)  021664  017646  000000            $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(1)  021670  116637  000001  022107            MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(1)  021676  112637  022111                    MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1)  021702  062716  000002                    ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(1)  021706  000406                            BR      $TYPON
(1)  021710  112737  000001  022107    $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
(1)  021716  112737  000006  022111            MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(1)  021724  112737  000005  022106    $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
(1)  021732  010346                            MOV     R3,-(SP)        ;;SAVE R3
(1)  021734  010446                            MOV     R4,-(SP)        ;;SAVE R4
(1)  021736  010546                            MOV     R5,-(SP)        ;;SAVE R5
(1)  021740  113704  022111                    MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  021744  005404                            NEG     R4
(1)  021746  062704  000006                    ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  021752  110437  022110                    MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(1)  021756  113704  022107                    MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(1)  021762  016605  000012                    MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(1)  021766  005003                            CLR     R3              ;;CLEAR THE OUTPUT WORD
(1)  021770  006105                    1$:     ROL     R5              ;;ROTATE MSB INTO "C"
(1)  021772  000404                            BR      3$              ;;GO DO MSB
(1)  021774  006105                    2$:     ROL     R5              ;;FORM THIS DIGIT
(1)  021776  006105                            ROL     R5
(1)  022000  006105                            ROL     R5
(1)  022002  010503                            MOV     R5,R3
(1)  022004  006103                    3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  022006  105337  022110                    DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  022012  100016                            BPL     7$              ;;BR IF NO
(1)  022014  042703  177770                    BIC     #177770,R3      ;;GET RID OF JUNK
(1)  022020  001002                            BNE     4$              ;;TEST FOR 0
(1)  022022  005704                            TST     R4              ;;SUPPRESS THIS 0?
(1)  022024  001403                            BEQ     5$              ;;BR IF YES
(1)  022026  005204                    4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  022030  052703  000060                    BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  022034  052703  000040            5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  022040  110337  022104                    MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  022044  104400  022104                    TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  022050  105337  022106            7$:     DECB    $OCNT           ;;COUNT BY 1
(1)  022054  003347                            BGT     2$              ;;BR IF MORE TO DO
(1)  022056  002402                            BLT     6$              ;;BR IF DONE
(1)  022060  005204                            INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  022062  000744                            BR      2$              ;;GO DO THE LAST DIGIT
(1)  022064  012605                    6$:     MOV     (SP)+,R5        ;;RESTORE R5
(1)  022066  012604                            MOV     (SP)+,R4        ;;RESTORE R4
(1)  022070  012603                            MOV     (SP)+,R3        ;;RESTORE R3
(1)  022072  016666  000002  000004            MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  022100  012616                            MOV     (SP)+,(SP)
(1)  022102  000002                            RTI                     ;;RETURN
(1)  022104     000            8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  022105     000                    .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  022106     000            $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
(1)  022107     000            $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
(1)  022110  000000            $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
  2005                                  .SBTTL  TYPE ROUTINE
   (1)
   (2)                                  ;;**********************************************************************
   (1)                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
   (1)                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
   (1)                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
   (1)                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
   (1)                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
   (1)                                  ;*
   (1)                                  ;*CALL:
   (1)                                  ;*1) USING A TRAP INSTRUCTION
   (1)                                  ;*      TYPE    ,MESADR         ;;MESADR    FIRST ADDRESS OF AN ASCIZ STRING
   (1)                                  ;*OR
   (1)                                  ;*      TYPE
   (1)                                  ;*      MESADR
   (1)                                  ;*
   (1)
   (1)  022112  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
   (1)  022116  100002                          BPL     1$              ;;BR IF YES
   (1)  022120  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
   (1)  022122  000430                          BR      3$              ;;LEAVE
   (1)  022124  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
   (1)  022126  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
   (1)  022132  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
   (1)  022140  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
   (1)  022142  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
   (1)  022150  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
   (1)  022152  010037  022162                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
   (1)  022156  004737  023260                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
   (1)  022162  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
   (1)  022164  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
   (1)  022172  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
   (1)  022174  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
   (1)  022176  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
   (1)  022200  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
   (1)  022202  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
   (1)  022204  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
   (1)  022210  000002                          RTI                     ;;RETURN
   (1)  022212  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
   (1)  022216  001430                          BEQ     8$
   (1)  022220  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
   (1)  022224  001006                          BNE     5$
   (1)  022226  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
   (1)  022230  104400                          TYPE                    ;;TYPE A CR AND LF
   (1)  022232  001171                          $CRLF
   (1)  022234  105037  022370                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
   (1)  022240  000755                          BR      2$              ;;GET NEXT CHARACTER
   (1)  022242  004737  022324          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
   (1)  022246  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
   (1)  022252  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
   (1)  022254  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
   (1)                                                                  ;;AND THE NULL CHAR.
   (1)  022260  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
   (1)  022264  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
   (1)  022266  004737  022324                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
   (1)  022272  105337  022370                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
```

```
(1)  022276  000770                          BR      7$              ;;LOOP
(1)
(1)                                  ;HORIZONTAL TAB PROCESSOR
(1)
(1)  022300  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  022304  004737  022324          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  022310  132737  000007  022370          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  022316  001372                          BNE     9$              ;;TAB STOP
(1)  022320  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
(1)  022322  000724                          BR      2$              ;;GET NEXT CHARACTER
(1)  022324  105777  156620          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  022330  100375                          BPL     $TYPEC
(1)  022332  116677  000002  156612          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  022340  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  022346  001003                          BNE     1$              ;;BRANCH IF NO
(1)  022350  105037  022370                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  022354  000406                          BR      $TYPEX          ;;EXIT
(1)  022356  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  022364  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
(1)  022366  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  022370  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(1)  022372  000207          $TYPEX: RTS     PC
(1)
2006                                 .SBTTL  TTY INPUT ROUTINE
(1)
(2)                                  ;;**********************************************************************
(1)                                  .ENABL  LSB
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
(1)                                  ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
(1)                                  ;*SERVICE THE TEST OR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
(1)                                  ;*WHEN OPERATING IN TTY FLAG MODE.
(1)  022374  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
(1)  022402  001074                          BNE     15$             ;;BRANCH IF NO
(1)  022404  105777  156534                  TSTB    @$TKS           ;;CHAR THERE?
(1)  022410  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
(1)  022412  117746  156530                  MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
(1)  022416  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
(1)  022422  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
(1)  022426  001062                          BNE     15$             ;;NO, RETURN TO USER
(1)  022430  123727  001134  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
(1)  022436  001456                          BEQ     15$             ;;BRANCH IF YES
(1)
(1)  022440  104400  023121                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
(1)  022444  104400  023126          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
(2)  022450  013746  000176                  MOV     $SWREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
(2)  022454  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  022456  104400  023137                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
(1)  022462  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
(1)  022464  005046                          CLR     -(SP)           ;;THE NEW SWR
(1)  022466  105777  156452          7$:     TSTB    @$TKS           ;;CHAR THERE?
(1)  022472  100375                          BPL     7$              ;;IF NOT TRY AGAIN
(1)
(1)  022474  117746  156446                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
(1)  022500  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
```

```
(1)
(1)
(1)
(1)  022504  021627  000025          9$:    CMP     (SP),#25        ;;IS IT A CONTROL-U?
(1)  022510  001005                         BNE     10$             ;;BRANCH IF NOT
(1)  022512  104400  023114                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (↑U)
(1)  022516  062706  000006          20$:   ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
(1)  022522  000757                         BR      19$             ;;LET'S TRY IT AGAIN
(1)
(1)
(1)  022524  021627  000015          10$:   CMP     (SP),#15        ;;IS IT A <CR>?
(1)  022530  001022                         BNE     16$             ;;BRANCH IF NO
(1)  022532  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
(1)  022536  001403                         BEQ     11$             ;;BRANCH IF YES
(1)  022540  016677  000002  156372         MOV     2(SP),@SWR      ;;SAVE NEW SWR
(1)  022546  062706  000006          11$:   ADD     #6,SP           ;;CLEAR UP STACK
(1)  022552  104400  001171          14$:   TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
(1)  022556  123727  001135  000001         CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS
(1)  022564  001003                         BNE     15$             ;;BRANCH IF NOT
(1)  022566  012777  000100  156350         MOV     #100,@#TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1)  022574  000002                 15$:    RTI                     ;;RETURN
(1)  022576  004737  022324          16$:   JSR     PC,$TYPEC       ;;ECHO CHAR
(1)  022602  021627  000060                 CMP     (SP),#60        ;;CHAR < 0?
(1)  022606  002420                         BLT     18$             ;;BRANCH IF YES
(1)  022610  021627  000067                 CMP     (SP),#67        ;;CHAR > 7?
(1)  022614  003015                         BGT     18$             ;;BRANCH IF YES
(1)  022616  042726  000060                 BIC     #60,(SP)+       ;;STRIP-OFF ASCII
(1)  022622  005766  000002                 TST     2(SP)           ;;IS THIS THE FIRST CHAR
(1)  022626  001403                         BEQ     17$             ;;BRANCH IF YES
(1)  022630  006316                         ASL     (SP)            ;;NO, SHIFT PRESENT
(1)  022632  006316                         ASL     (SP)            ;;    CHAR OVER TO MAKE
(1)  022634  006316                         ASL     (SP)            ;;    ROOM FOR NEW ONE.
(1)  022636  005266  000002          17$:   INC     2(SP)           ;;KEEP COUNT OF CHAR
(1)  022642  056616  177776                 BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
(1)  022646  000707                         BR      7$              ;;GET THE NEXT ONE
(1)  022650  104400  001170          18$:   TYPE    ,$QUES          ;;TYPE ?<CR><LF>
(1)  022654  000720                         BR      20$             ;;SIMULATE CONTROL-U
(1)                                  .DSABL  LSB
(1)
(1)
(2)                                  ;;**************    ******************************************
(1)                                  ;*THIS ROUTINE       INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                  ;*CALL:
(1)                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                  ;*      RETURN                  ;;CHARACTER IS ON THE STACK
(1)                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
(1)                                  ;
(1)
(1)  022656  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
(1)  022660  016666  000004  000002         MOV     4(SP),2(SP)     ;;SAVE THE PS
(1)  022666  105777  156252          1$:    TSTB    @$TKS           ;;WAIT FOR
(1)  022672  100375                         BPL     1$              ;;A CHARACTER
(1)  022674  117766  156246  000004         MOVB    @$TKB,4(SP)     ;;READ THE TTY
(1)  022702  042766  177600  000004         BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
(1)  022710  026627  000004  000023         CMP     4(SP),#23       ;;IS IT A CONTROL-S?
(1)  022716  001013                         BNE     3$              ;;BRANCH IF NO
```

```
(1)  022720  105777  156220          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
(1)  022724  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
(1)  022726  117746  156214                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
(1)  022732  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)  022736  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
(1)  022742  001366                          BNE     2$              ;;IF NOT DISCARD IT
(1)  022744  000750                          BR      1$              ;;YES, RESUME
(1)  022746  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
(1)  022754  002407                          BLT     4$              ;;BRANCH IF YES
(1)  022756  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
(1)  022764  003003                          BGT     4$              ;;BRANCH IF YES
(1)  022766  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
(1)  022774  000002                  4$:     RTI                     ;;GO BACK TO USER
(2)                                ;;**********************************************************************
(1)                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1)                                ;*CALL:
(1)                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
(1)                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1)                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
(1)
(1)  022776  010346                $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
(1)  023000  012703  023104          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
(1)  023004  022703  023114          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
(1)  023010  101405                          BLOS    4$              ;;BR IF YES
(1)  023012  104407                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
(1)  023014  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
(1)  023016  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
(1)  023022  001003                          BNE     3$              ;;SKIP IF NOT
(1)  023024  104400  001170          4$:     TYPE    ,$QUES          ;;TYPE A '?'
(1)  023030  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1)  023032  111337  023102          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
(1)  023036  104400  023102                  TYPE    ,9$
(1)  023042  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
(1)  023046  001356                          BNE     2$              ;;LOOP IF NOT RETURN
(1)  023050  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
(1)  023054  104400  001172                  TYPE    ,$LF            ;;TYPE A LINE FEED
(1)  023060  012603                          MOV     (SP)+,R3        ;;RESTORE R3
(1)  023062  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  023064  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
(1)  023072  012766  023104  000004          MOV     #$TTYIN,4(SP)
(1)  023100  000002                          RTI                     ;;RETURN
(1)  023102     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  023103     000                          .BYTE   0               ;;TERMINATOR
(1)  023104  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
(1)  023114  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
(1)  023121     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
(1)  023126  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)  023134  020075     000
(1)  023137     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
(1)  023144  036440  000040
2007                               .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
(1)
(2)                                ;;**********************************************************************
(1)                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
(1)                                ;*CHANGE IT TO BINARY.
(1)                                ;*CALL:
```

```
(1)                                     ;*      RDOCT                   ;;READ AN OCTAL NUMBER
(1)                                     ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
(1)                                     ;*                              ;;HIGH ORDER BITS ARE IN $H OCT
(1)
(1)  023150  011646                     $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
(1)  023152  016666  000004  000002             MOV     4(SP),2(SP)     ;;INPUT NUMBER
(3)  023160  010046                             MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  023162  010146                             MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  023164  010246                             MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(1)  023166  104410                     1$:     RDLIN                   ;;READ AN ASCIZ LINE
(1)  023170  012600                             MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
(1)  023172  005001                             CLR     R1              ;;CLEAR DATA WORD
(1)  023174  005002                             CLR     R2
(1)  023176  112046                     2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
(1)  023200  001412                             BEQ     3$              ;;IF ZERO GET OUT
(1)  023202  006301                             ASL     R1              ;;*2
(1)  023204  006102                             ROL     R2
(1)  023206  006301                             ASL     R1              ;;*4
(1)  023210  006102                             ROL     R2
(1)  023212  006301                             ASL     R1              ;;*8
(1)  023214  006102                             ROL     R2
(1)  023216  042716  177770                     BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
(1)  023222  062601                             ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
(1)  023224  000764                             BR      2$              ;;LOOP
(1)  023226  005726                     3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
(1)  023230  010166  000012                     MOV     R1,12(SP)       ;;SAVE THE RESULT
(1)  023234  010237  023250                     MOV     R2,$HIOCT
(3)  023240  012602                             MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  023242  012601                             MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  023244  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  023246  000002                             RTI                     ;;RETURN
(1)  023250  000000                     $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
2008                                    .SBTTL  APT COMMUNICATIONS ROUTINE
(1)
(2)                                     ;;**************************************************************
(1)  023252  112737  000001  023516     $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
(1)  023260  112737  000001  023514     $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
(1)  023266  000403                             BR      $ATYC
(1)  023270  112737  000001  023516     $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
(2)  023276                             $ATYC:
(3)  023276  010046                             MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  023300  010146                             MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(1)  023302  105737  023514                     TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
(1)  023306  001450                             BEQ     5$              ;;IF NOT:  BR
(1)  023310  122737  000001  001214             CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
(1)  023316  001031                             BNE     3$              ;;IF NOT:  BR
(1)  023320  132737  000100  001215             BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(1)  023326  001425                             BEQ     3$              ;;IF NOT:  BR
(1)  023330  017600  000004                     MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
(1)  023334  062766  000002  000004             ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(1)  023342  005737  001174             1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
(1)  023346  001375                             BNE     1$              ;;IF NOT:  WAIT
(1)  023350  010037  001210                     MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
(1)  023354  105720                     2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
(1)  023356  001376                             BNE     2$
(1)  023360  163700  001210                     SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
```

```
(1)  023364  006200                        ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
(1)  023366  010037  001212                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
(1)  023372  012737  000004  001174        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
(1)  023400  000413                        BR      5$
(1)  023402  017637  000004  023426  3$:   MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
(1)  023410  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
(3)  023416  013746  177776                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
(1)  023422  004737  022112                JSR     PC,$TYPE        ;;CALL TYPE MACRO
(1)  023426  000000                  4$:   .WORD   0
(1)  023430                          5$:
(1)  023430  105737  023516          10$:  TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
(1)  023434  001416                        BEQ     12$             ;;IF NOT:  BR
(1)  023436  005737  001214                TST     $ENV            ;;RUNNING UNDER APT?
(1)  023442  001413                        BEQ     12$             ;;IF NOT:  BR
(1)  023444  005737  001174          11$:  TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
(1)  023450  001375                        BNE     11$             ;;IF NOT:  WAIT
(1)  023452  017637  000004  001176        MOV     @4(SP),$FATAL   ;;GET ERROR #
(1)  023460  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(1)  023466  005237  001174                INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
(1)  023472  105037  023516          12$:  CLRB    $FFLG           ;;CLEAR FATAL FLAG
(1)  023476  105037  023515                CLRB    $LFLG           ;;CLEAR LOG FLAG
(1)  023502  105037  023514                CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
(3)  023506  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  023510  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  023512  000207                        RTS     PC              ;;RETURN
(1)  023514     000                 $MFLG: .BYTE   0               ;;MESSG. FLAG
(1)  023515     000                 $LFLG: .BYTE   0               ;;LOG FLAG
(1)  023516     000                 $FFLG: .BYTE   0               ;;FATAL FLAG
(1)          023520                        .EVEN
(1)          000200                 APTSIZE=200
(1)          000001                 APTENV=001
(1)          000100                 APTSPOOL=100
(1)          000040                 APTCSUP=040
2009
2010                                .SBTTL  TRAP DECODER
(1)
(2)                                 ;;**************************************************************
(1)                                 ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
(1)                                 ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(1)                                 ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1)                                 ;*GO TO THAT ROUTINE.
(1)
(1)  023520  010046                 $TRAP: MOV     R0,-(SP)        ;;SAVE R0
(1)  023522  016600  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
(1)  023526  005740                        TST     -(R0)           ;;BACKUP BY 2
(1)  023530  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
(1)  023532  006300                        ASL     R0              ;;POSITION FOR INDEXING
(1)  023534  016000  023542                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
(1)  023540  000200                        RTS     R0              ;;GO TO ROUTINE
(1)
(3)                                 .SBTTL  TRAP TABLE
(3)
(3)                                 ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(3)                                 ;*BY THE "TRAP" INSTRUCTION.
(3)
(3)                                 ;       ROUTINE
```

```
(3)                                         .       -------
(3)   023542                                $TRPAD:
(3)   023542  022112                                $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
(3)   023544  021710                                $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3)   023546  021664                                $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3)   023550  021724                                $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(3)   023552  020520                                $TYPDS  ;;CALL=TYPDS    TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)
(1)
(3)   023554  022444                                $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING
(1)
(3)   023556  022374                                $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
(3)   023560  022656                                $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
(3)   023562  022776                                $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
(3)   023564  023150                                $RDOCT  ;;CALL=RDOCT    TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
2011
2012  023566  000074                    BUF2:   .BLKW   60.
2013  023756  000074                    BUF3:   .BLKW   60.
2014  024146  000074                    BUFFER: 60.
2015
2016  024150  000240                            NOP
2017          000001                            .END
```

```
A       010516          1043    1117#   1979    1988
ABASE = 170416            16#     30
AC      002116           180#    202
ACDW1 = 000000            30
ACDW2 = 000000            30
ACPUOP= 000000            30
ACTRLC  013332          1234    1549    1584#   1618    1631    1668    1703    1760    1997    1998
AD      002114           179#    200
ADDW0 = 000000            30
ADDW1 = 000000            30
ADDW10= 000000            30
ADDW11= 000000            30
ADDW12= 000000            30
ADDW13= 000000            30
ADDW14= 000000            30
ADDW15= 000000            30
ADDW2 = 000000            30
ADDW3 = 000000            30
ADDW4 = 000000            30
ADDW5 = 000000            30
ADDW6 = 000000            30
ADDW7 = 000000            30
ADDW8 = 000000            30
ADDW9 = 000000            30
ADEND   014602          1301    1311    1325    1336    1347    1356    1365    1390    1401    1828#   1833#   1836#   1837#
                        1838#   1839#   1840#   1841    1851#
ADEVCT= 000000            30
ADEVM = 000000            30
AENV  = 000000            30
AENVM = 000000            30
AFATAL= 000000            30
AL      002112           178#    198
AMADR1= 000000            30
AMADR2= 000000            30
AMADR3= 000000            30
AMADR4= 000000            30
AMAMS1= 000000            30
AMAMS2= 000000            30
AMAMS3= 000000            30
AMAMS4= 000000            30
AMSGAD= 000000            30
AMSGLG= 000000            30
AMSGTY= 000000            30
AMTYP1= 000000            30
AMTYP2= 000000            30
AMTYP3= 000000            30
AMTYP4= 000000            30
APASS = 000000            30
APRIOR= 000200           18#     30
APTCSU= 000040          2005    2008#
APTENV= 000001          1998    2005    2008#
APTSIZ= 000200           136    2008#
APTSPD= 000100          2005    2008#
ASWREG= 000000            30
ATESTN= 000000            30
AUNIT = 000000            30
```

```
AUSWR = 000000            30
AUTCAL  011500           216    1241#   1535
AUTOCL  017353          1244    1928#
AUTOER  017437          1540    1931#
AUTRAP  013150          1250    1537#
AVECT1= 100360            17#     30
AVECT2= 000000            30
B       010523          1118#
BEG     001506           132     134#
BEGIN   001462            23     129#   2000
BEGIN1  001500            24     133#
BIT0  = 000001            14#     547     644     659
BIT00 = 000001            14#
BIT01 = 000002            14#
BIT02 = 000004            14#
BIT03 = 000010            14#
BIT04 = 000020            14#
BIT05 = 000040            14#
BIT06 = 000100            14#
BIT07 = 000200            14#
BIT08 = 000400            14#    1997
BIT09 = 001000            14#
BIT1  = 000002            14#     431     432     676    1086
BIT10 = 002000            14#     506     507     623     684    1332    1343    1361    1377    1378    1397    1658    1692
BIT11 = 004000            14#     302     328     355     383     514     515     523     524     533     684    1343    1361
                         1386    1997
BIT12 = 010000            14#     619     624    1332    1341    1386    1412    1659
BIT13 = 020000            14#    1295    1317    1441    1447    1457    1463    1488    1496    1506    1514    1557    1565
                         1998
BIT14 = 040000            14#    1441    1457    1488    1506    1561    1565    1997
BIT15 = 100000            14#     252
BIT2  = 000004            14#     445     446     523     533     534     564     565     573     586     676
BIT3  = 000010            14#     461     462     523     533     534     595     596     603     614     676
BIT4  = 000020            14#     476     477     676     745     753     762     769     841     884     942     979    1083
                         1219    1370    1371    1378
BIT5  = 000040            14#    1656    1662
BIT6  = 000100            14#     490     491     644     659     684
BIT7  = 000200            14#     423     432     438     446     452     462     469     477     483     491     499     507
                          515     524     534     557     564     586     595     614     677     685     769    1378    1427
                         1731    1735
BIT8  = 000400            14#    1763
BIT9  = 001000            14#     498     499     523     524     533     684    1361    1397    1418    1426
BPTVEC= 000014            14#
BRLEV1  020436           164#    165#    166     658     739#   1790#   1970#
BRLEV2  020440           166#    167#    643    1371#
BUFFER  024146          2014#
BUF2    023566          2012#
BUF3    023756          2013#
C       010530          1119#    1977    1986
CALDAC  013454           219    1615#
CALLUN  017173          1534    1922#
CHANL   014606          1825#   1853#
CHBR    010324          1069    1077#   1174    1188
CHBR1   010412          1089#   1104
CHBR3   010420          1091#   1100
CHBR3   010442          1092    1096#   1097
```

```
CHAR4   010370           1082   1084#  1085
CHRCOL  010512            535#   536#   537#  1067#  1070#  1112#
CHTIME  014324            821    833    864    921    970   1035   1161   1789#
CH0     = 000000         1257#
CH01    020454           1172   1977#
CH02    020476           1186   1966#
CH1     = 000001         1258#
CH10    = 000010         1265#
CH11    = 000011         1266#
CH12    = 000012         1267#
CH13    = 000013         1268#
CH14    = 000014         1269#
CH15    = 000015         1270#
CH16    = 000016         1271#
CH17    = 000017         1272#
CH2     = 000002         1259#
CH3     = 000003         1260#  1300
CH4     = 000004         1261#  1310
CH40    = 000040         1276#
CH41    = 000041         1277#
CH42    = 000042         1278#
CH43    = 000043         1279#
CH44    = 000044         1280#
CH45    = 000045         1281#
CH46    = 000046         1282#
CH47    = 000047         1283#
CH5     = 000005         1262#
CH50    = 000050         1284#
CH51    = 000051         1285#  1472   1525   1808
CH52    = 000052         1286#  1477   1531
CH53    = 000053         1287#  1324   1335   1346
CH54    = 000054         1288#
CH55    = 000055         1289#  1355   1364
CH56    = 000056         1290#  1389   1400
CH57    = 000057         1291#
CH6     = 000006         1263#
CH7     = 000007         1264#
CKSWR   = 104406          193   1550   1617   1630   1655   1675   1702   1759   1997   1998   2010#
CLRVC   013610           1199   1203   1655#
COMPAR  014612           1303   1313   1811   1856#
CONTC   017002            222   1913#
CONVRT  014444           1299   1309   1323   1334   1345   1354   1363   1388   1399   1807   1824#
CPTYPE  014350           1789   1796#
CR      = 000015           14#  2005
CRLF    = 000200           14#  2005
CROSS   014146           1294   1317   1434   1441   1447   1457   1463   1488   1496   1506   1514   1744#
CSPACE  013374           1444   1451   1460   1467   1492   1500   1510   1518   1596#  1597   1608
CTRLC   002312            187    221#  1541   1590   1602
D       010535           1120#
DAC0    001446            122#   154    276    287#   288    295#   296    303#   304    393#   399    574#   693#
                          696    738#   781#   814    827    665    922    973#   986   1036   1162   1220   1440#
                         1456#  1471   1555   1620#  1638#  1690   1955
DAC1    001450            123#   277    313#   314    321#   322    329#   330    394#   405    604#   702#   705
                          739#   782#   815    826    866    923    974#   985   1036   1162   1222   1448#  1464#
                         1476   1559   1621#  1639#  1691   1955
DAC2    001452            124#   278    340#   341    348#   349    356#   357    395#   411    712#   715    740#
```

```
                        783*    818     831     869     926     975*    983    1036    1162    1224    1487*   1505*   1524
                       1563    1622*   1640*   1687    1957
DAC3    001454          125*    279     368*    369     376*    377     384*    385     396*    417     720*    723     741*
                        784*    819     830     870     927     976*    982    1036    1162    1226    1495*   1497*   1513*
                       1530    1567    1623*   1641*   1688    1958
DDISP = 177570          148     30      136
DELAY   001442          120*   1572    1576
DFD     020400           38     44       50      56      62      68      75     81      87      93      99      105     111
                        117    1962*
DH1     016645           36     42       48      54      60      85    1907*
DH13    016715           97    1909*
DH14    016747           91    103      109     115    1910*
DH6     016701           66     73       79    1908*
DISPLA  001142           30*    136*    1997*   1998*
DISPRE  000174           23*    136
DSWR  = 177570           14*     30      136
DT1     020264           37     86      1954*
DT13    020354           95    1960*
DT14    020366           92    104      110     116    1961*
DT2     020276           43    1955*
DT3     020310           49    1956*
DT4     020322           55    1957*
DT5     020334           61    1958*
DT6     020346           67     74       80    1959*
DYNCAL  013520           25    1628*
E       010542         1121*   1982    1991
EMTVEC= 000030           14*    136*
EM1     016024           35    1893*
EM10    016300           78    1900*
EM11    016351           84    1901*
EM12    016400           90    1902*
EM13    016443           96    1903*
EM14    016464          102    1904*
EM15    016533          108    1905*
EM16    016602          114    1906*
EM2     016055           41    1894*
EM3     016104           47    1895*
EM4     016133           53    1896*
EM5     016162           59    1897*
EM6     016211           65    1898*
EM7     016244           72    1899*
ERRVEC= 000004           14*    136*    231*    274*    283*   1250*   1538*   1997*
ETX   = 000003         1946*   1949    1952
EVER    020442          130*    241     251*    252*    254     778    1960    1972*
EVER1   020444          131*    806     812*   1973*
F       010547         1122*
FCHANL  014604         1852*
FILZ    014052         1709*   1727*   1731*   1735*   1739*   1746*
FOUND1  017010          244    1914*
FOUND2  017034          250    1916*
FULRMP  011364          218    1215*
FUNDAC  014352         1470    1475    1523    1529   1802*
G       010554         1123*
GADCS   014574         1829*   1830*   1831    1848*
GADDBP  014575         1833    1849*
GEN1    010310         1069*   1071
```

```
GNS     = ****** J          23    1318    2010
GSTAT     014600          1418*   1426*   1850*
GTSWR   = 104405          2010#
H         010561          1124#   1978    1987
HIGH      020422          1964#
HT      = 000011            14#   2005
I         010566          1125#
INCR      020424          1965#
INIT1     002126           138     163#
IOTVEC= 000020              14=    136*
IV        001456           126#    152     160*    161*    162     641*    651*    656*    666*    672*    785*
IVS       001460           127#    162*    163*    651     652*    672     673*    786*
J         010573          1126#
K         010600          1127#
L         010605          1128#   1983    1992
LDTRAP    001742           141#    185     258     1216    1242    1546    1616    1629
LF      = 000012            14#   2005
LOADVC    013726          1200    1682#
LOOPNT    007754           995     997     1000    1002    1010#
LOW       020420          1963#
LTEST     002466           260#    788
M         010612          1129#
MANMED    017232          1548    1924#
MANUL     013170           217    1545#
MC        002124           183#    208
MD        002122           182#    205
MES15     015507           811    1889#
MES3      015412          1674    1886#
MES6      015457           809    1887#
ML        002120           181#    204
MTEST     002324           189     214     229#    791
N         010617          1052    1130#   1980    1981    1989    1990
NSUPLY    011622          1304    1308#
N0        010720          1061    1143#
N1        010725          1144#   1985
N2        010732          1145#   1994
N3        010737          1146#
N4        010744          1147#
N5        010751          1148#
N50000    020235          1439    1486    1947#
N6        010756          1149#
N7        010763          1150#
N8        010770          1151#
N9        010775          1152#
O         010624          1131#
OPRIN     020446           195=    196*    198     200     202     204     206     208     1974#
P         010631          1132#
P88       007002           822     834     838#
PC      =%000007            14#    168*    185*    258*    791*    821*    822*    823*    833*    834*    835*    855*    864*
                           915*    921*    963*    970*   1006*   1035*   1045*   1054*   1063*   1064*   1069*   1072*   1136*
                          1161*   1174*   1188*   1192*   1199*   1230*   1201*   1203*   1216*   1221*   1223*   1225*   1227*
                          1234*   1235*   1242*   1303*   1313*   1444*   1451*   1460*   1467*   1492*   1500*   1510*   1518*
                          1546*   1549*   1591*   1605*   1616*   1618*   1629*   1631*   1633*   1635*   1645*   1668*   1679*
                          1703*   1706*   1760*   1768*   1794*   1811*   1868*   1872*   1997*   1998*   1999*   2000    2005*
                          2006*   2008*
PICO      006650           807     814*   1205    1780
```

```
PIC1    006724          826#   1781
PIC12   011326         1194    1198#   1787
PIC3    007110          837     862#   1782
PIC4    007340          919#   1783
PIC4B   007356          922#
PIC5    007546          969#   1784
PIC6    010004         1008    1034#   1785
PIC6A   010060         1037#   1065
PIC6B   010276         1045    1054    1063    1067#
PIC7    011022         1066    1160#   1786
PIC7A   011064         1173#   1177
PIC7AA  011102         1164#   1193
PIC7B   011272         1187#   1191
PIRQ   = 177772          14#
PIRQVE= 000240           14#
PR0    = 000000          14#
PR1    = 000040          14#
PR2    = 000100          14#
PR3    = 000140          14#
PR4    = 000200          14#
PR5    = 000240          14#
PR6    = 000300          14#
PR7    = 000340          14#
PS     = 177776          14#    137*
PSUPLY  011560         1252    1253    1298#
PS%    = 177776          14#    640*    643*    655*    658*    667*
PWRMSG  021616         2000    2001#
PWRVEC= 000024           14#    136*   2000*
P3      007166          873#    916
P3A     007236          883     885#    890
P3B     007256          893#    898
P3C     007276          901#    906
P3D     007316          909#    914
P4      007416          930#    964
P4A     007466          941     943#    949
P4B     007522          956#    962
P49976  020250         1455    1504    1950#
P7CNT   020450         1171*   1176*   1185*   1190*   1975#
P7PNT   020452         1172*   1173    1175*   1186*   1187    1189*   1976#
Q       010636         1133#
QMARK   016774          211    1607    1912#
R       010643         1134#
RDCHR  = 104407         2006    2010#
RDLIN  = 104410          194    2007    2010#
RDOCT  = 104411         2010#
RESVEC= 000010           14#
ROUTPT  014304         1775    1780#
R0     =%000000          14#    150*    151*    152     154*    156*    545*    555*    563*    584*    594*    612*    621*
                        633*    642*    645*    657*    660*    747*    750*    763*    767*    791*    814*    818*    826*
                        830*    857*    881*    889*    892*    897*    900*    905*    908*    913*    939*    948*    955*
                        961*    982*    985*    989     990*   1004    1015*   1088*   1103*   1220*   1222*   1224*   1226*
                       1230*   1231*   1232    1551*   1552    1553*   1557    1561    1565    1619*   1620    1621    1622
                       1623    1632*   1634*   1638    1639    1640    1641    1642*   1643*   1660*   1669*   1684*   1723*
                       1724    1747*   1751*   1777    1856    1858*   1860*   1862*   1863    1866*   1871*   1996*   1999*
                       2000*   2005*   2007*   2008*   2010*
R1     =%000001          14#    142*    143*    144*    145     155*    156     157     158     815*    819*    827*    831*
```

```
                        856*  865*  869*  875*  878*  922*  926*  933*  936*  983*  986*  990   991*
                       1011*  1089*  1099*  1552*  1571   1661*  1666*  1687*  1690*  1724*  1726*  1727   1728*
                       1729*  1732*  1733*  1738*  1740*  1748*  1749*  1857   1859*  1860   1865*  1870*  1996*
                       2000*  2007*  2008*
R2      =%000002         14#   141*  143    145*  146    147    844*  849    866*  870    876*  879*  923*
                        927*  932*  935*  1010*  1016*  1043*  1052*  1061*  1173*  1187*  1555*  1559*  1563*
                       1567*  1569*  1571*  1575*  1688*  1691*  1996*  2000*  2007*
R22       017677       1939#
R3      =%000003         14#   843*  853    871*  886*  894*  902*  910*  928*  943*  956*  989*  991
                        992*  1012*  1013   1090*  1091*  1572*  1573*  1576*  1577*  1996*  2000*  2004*  2006*
R37       020011       1459   1941#
R38       017565       1443   1937#
R39       020056       1466   1942#
R4      =%000004         14#   842*  872*  885    893    901    909    929*  946    947    959    960    1087*
                       1098   1102   1105   2000*  2004*
R40       017632       1450   1938#
R41       020123       1509   1943#
R43       020170       1517   1944#
R44       017744       1499   1940#
R5      =%000005         14#   995*  997*  1000*  1002*  1011   1018   101?*  1068*  1095*  1096   1163*  1294*
                       1299*  1309*  1317*  1323*  1334*  1345*  1354*  136.*  1388*  1399*  1434*  1438*  1441*
                       1447*  1454*  1457*  1463*  1470*  1475*  1485*  1488*  1496*  1503*  1506*  1514*  1523*
                       1529*  1723   1742*  1744   1753*  1802   1803   1807*  1824   1842*  1996*  2000*  2004*
R6      =%000006         14#   136*
R7      =%000007         14#
S         010650       1135#
SELD01    017061       1246   1918#
SELD23    017126       1483   19_0#
SNOVLT    014054       1438   1454   1485   1503   1723#
SP      =%000006         14#   136*  192*  195    236    245*  281    650    671    791*  805*  1215*  1241*
                       1537   1545*  ...*  1628*  1777   1856*  1857*  1865   1866   1867*  1870   1871   1996*
                       1997*  1998   1999*  2000*  2004*  2005*  2006*  2007*  2008*  2010*
SPACEA    011002       1153#  1984   1993
SPREAD    014610       1302*  1312*  1810*  1854#  1863
STACK =   001100         14#   136    192    805    1215   1241   1545   1615   1628
STAT      001444        121#   150    275    397*   425    431*   433    437*   439    445*   447    451*   453
                        461*   463    468*   470    476*   478    482*   484    490*   492    498*   500    506*
                        508    514*   516    523*   525*   526    533*   535    538    547*   548    553    558
                        565*   566    575    582    587    596*   597    605    610    615    623*   624*   625
                        629    644*   647*   659*   663*   668*   676*   679    684*   687    745*   746*   748
                        754    762*   764*   765    770    780*   838*   841*   842    862*   871    884*   919*
                        928    942*   972*   979*   992    1044*  1053*  1062*  1068   1083*  1084   1163   1164*
                       1178*  1219*  1321*  1331*  1332*  1343*  1352*  1361*  1370*  1372   1373   1377*  1379
                       1383*  1386*  1397*  1406*  1410*  1412*  1413   1420   1428   1569   1659*  1659*  1664
                       1682*  1684   1954   1959   1960   1961
STKLMT=   177774         14#
STX     =   000001     1945#  1947   1950
SUPOK     011664       1314   1317#
SWR       001140         30#   136*  619    692    701    711    719    738    739    740    741    743    760
                        816    828    839    845    867    873    882    924    930    940    977    980    987
                        993   1036   1039   1048   1057   1081   1162   1167   1181   1217   1551   1619   1632
                       1656   1662   1671   1676   1685   1761   1997   1998   2000*  2006*
SWREG     000176         23#   136    2006
SWO     =   000001         14#
SWOO    =   000001         14#
SWO1    =   000002         14#
```

DZAACA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
SW02  = 000004          14#
SW03  = 000010          14#   816   828   867   924   980  1036  1162  1685
SW04  = 000020          14#
SW05  = 000040          14#
SW06  = 000100          14#
SW07  = 000200          14#   987
SW08  = 000400          14#
SW09  = 001000          14#   692   701   711   719   738   739   740   741   845   873   930   993
                       1039  1048  1057  1167  1181
SW1   = 000002          14#
SW10  = 002000          14#   743   760   839   882   940   977  1081  1217
SW11  = 004000          14#
SW12  = 010000          14#  1671
SW13  = 020000          14#
SW14  = 040000          14#
SW15  = 100000          14#
SW2   = 000004          14#
SW3   = 000010          14#
SW4   = 000020          14#
SW5   = 000040          14#
SW6   = 000100          14#
SW7   = 000200          14#
SW8   = 000400          14#
SW9   = 001000          14#
T       010655         1136#
TBITVE= 000014           14#
TEMP    020432          129#   133#   186   622#   631#  1968#
TEMP1   020434         1683#  1693  1969#
TICKS   020430          820#   832#   863#   920#   969#  1034#  1160#  1198#  1765#  1792#  1967#
TIMER   014210          823    835    915    963   1006  1064  1192  1201  1759#
TIMERA  014224         1763#
TIMER1  014246         1766   1768#
TIMER2  014250         1764   1773#
TIMER4  014300         1778#
TIMSV   020426         1761#  1763   1773#  1774#  1775#  1776#  1778  1966#
TITLE   014662          191   1877#
TKVEC = 000060           14#
TPVEC = 000064           14#
TRAPVE= 000034           14#   136#
TRTVEC= 000014           14#
TST1    002470          240    272#
TST10   003056          338#
TST11   003112          343    346#
TST12   003150          351    354#
TST13   003222          366#
TST14   003256          371    374#
TST15   003314          379    382#
TST16   003366          392#
TST17   003554          419    422#
TST2    002546          285#
TST20   003614          427    430#
TST21   003704          441    444#
TST22   003774          455    460#
TST23   004064          472    475#
TST24   004154          486    489#
TST25   004212          494    497#
```

DZAACA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
TST26   004250            502     505*
TST27   004306            510     513*
TST3    002602            290     293*
TST30   004344            518     522*
TST31   004406            528     532*
TST32   004462            540     543*
TST33   004552            552     554     561*
TST34   004700            571     580     583     592*
TST35   005024            602     609     611     618*
TST36   005124            620     628     630     639*
TST37   005220            654*
TST4    002640            298     301*
TST40   005340            665     675*
TST41   005406            681     683*
TST42   005454            689     691*
TST43   005530            692     698     700*
TST44   005604            701     707     710*
TST45   005652            711     716     718*
TST46   005720            719     724     738*
TST47   005776            738     739*
TST5    002712            311*
TST50   006054            739     740*
TST51   006132            740     741*
TST52   006210            741     742*
TST53   006300            744     752     759*
TST54   006370            761     775*
TST55   011556           1297*
TST56   011620           1307*
TST57   011730           1316    1320*
TST6    002746            316     319*
TST60   011774           1327    1330*
TST61   012044           1338    1342*
TST62   012110           1349    1351*
TST63   012152           1358    1360*
TST64   012216           1367    1369*
TST65   012320           1385*
TST66   012364           1392    1396*
TST67   012434           1409*
TST7    003004            324     327*
TST70   012570           1436*
TST71   012750           1480*
TYPDS = 104404            791    2010*
TYPE  = 104400            190     210     221     243     249     791     808     810    1243    1245    1318    1442    1449
                         1458    1465    1482    1490    1498    1508    1516    1533    1539    1547    1606    1673    1996
                         1998    1999    2000    2004    2005    2006    2010*
TYPOC = 104401           1999    2006    2010*
TYPON = 104403           2010*
TYPOS = 104402            246    2010*
U       010662           1137*
V       010667           1138*
VADDR   001136            118*    233     780     781     782     783     784
VCXTMP  010502           1036*   1079*   1093*   1108*   1162*
VCYTMP  010504           1036*   1080*   1094*   1109*   1162*
VISUAL  006622            215     805*
VVECT   001440            119*    785     786
V1754   014570           1298    1846*
```

```
V24     014572          1302    1309    1312    1847#
W       010674          1139#
WAITIN  002152           192#    212     224
X       010701          1140#
XDACUT  014442          1802*   1804*   1815*   1820#
XPOS    010510          1037*   1041*   1046*   1050*   1055*   1059*   1079    1093    1102*   1105*   1111#   1165*   1169*
                        1179*   1183*
Y       010706          1141#
YPOS    010506          1038*   1042*   1047*   1051*   1056*   1060*   1077    1080    1094    1098*   1101*   1110#   1166*
                        1170*   1180*   1184*
YPT     010514          1077*   1101    1113#
Z       010713          1142#
$APTHD  001000            29#
$ASTAT= ****** U        2008
$ATYC   023276          2008#
$ATY1   023252          2008#
$ATY3   023260          2005    2008#
$ATY4   023270          1998    2008#
$AUTOB  001134            30#   2006
$BASE   001250            30#    151     229
$BDADR  001122            30#
$BDDAT  001126            30#    229*    232     233*    288*    289     296*    297     304*    305     314*    315     322*
                         323    330*    331     341*    342     349*    350     357*    358     369*    370     377*    378
                         385*    386    399*    400     405*    406     411*    412     417*    418     425*    426     433*
                         434    439*    440     447*    448     453*    454     463*    464     470*    471     478*    479
                         484*    485    492*    493     500*    501     508*    509     516*    517     526*    527     538*
                         539    548*    549     558*    566*    567     575*    576     587*    597*    598     605*    606
                         615*    679*    680     687*    688     696*    697     705*    706     715*    723*    738*    739*
                         740*    741*    754*    770*   1301*   1311*   1325*   1326    1336*   1337    1347*   1348    1356*
                        1357    1365*   1366    1373*   1374    1379*   1380    1390*   1391    1401*   1402    1413*   1414
                        1420*   1421    1428*   1429    1841*   1859    1954    1955    1956    1957    1958    1961
$CDW1   001254            30#
$CHARC  022370          2005#*
$CKSWR  022374          2006#   2010
$CMTAG  001100            30#    136
$CM1    =  000002         30#
$CM2    =  000004         30#
$CM3    =  000002         30#
$CNTLG  023121          2006#
$CNTLU  023114          2006#
$CPUOP  001222            30#
$CRLF   001171            30#   1998    1999    2005    2006
$DBLK   020734          1996#
$DEVCT  001204            30#
$DEVM   0C.252            30#
$DORGN  006576            791#
$DTBL   020724          1996#
$ENDAD  006566            27     791#
$ENDCT  006534          791#
$ENDMG  006605          791#
$ENULL  006602          791#
$ENV    001214            30#   1998    2005    2008
$ENVM   001215            30#    136    2005    2008
$EOP    006500          779     791#
$EOPCT  006526          791#
$ERFLG  001103            30#   1997*   1998*
```

DZAACA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
$ERMAX  001115          30#  1997
$ERROR  021160         136   1998#
$ERRPC  001116          30#  1954   1955   1956   1957   1958   1959   1960   1961   1998*  1999
$ERRTB  001256          30#  1999
$ERRTY  021310        1998   1999#
$ERTTL  001112          30#  1998*
$ETABL  001214          30#
$ETEND  001256          29    30#
$FATAL  001176          30#  2008*
$FFLG   023516        2008#*
$FILLC  001156          30#  2005
$FILLS  001155          30#  2005
$GOADR  001120          30#
$GODAT  001124          30#   286*   287    289    294*   295    297    302*   303    305    308*   312*   313
                       315    320*   321    323    328*   329    331    334*   339*   340    342    347*   348
                       350    355*   356    358    361*   367*   368    370    375*   376    378    383*   384
                       386    389*   398*   400    404*   406    410*   412    416*   418    423*   426    432*
                       434    438    440    446*   448    452*   454    462*   464    469*   471    477*   479
                       483*   485    491*   493    499*   501    507*   509    515*   517    524*   527    534*
                       539    546*   557*   564*   573*   586*   595*   603*   614*   677*   680    685*   688
                       694*   697    703*   706    713*   721*   738*   739*   740*   741*   753*   769*  1298*
                      1308*  1322*  1326   1333*  1337   1344*  1348   1353*  1357   1362*  1366   1371*  1374
                      1378*  1380   1387*  1391   1398*  1402   1411*  1414   1419*  1421   1427*  1429   1805*
                      1816*  1858   1954   1955   1956   1957   1958   19€!
$GET42  006556         791#
$GTSWR  022444        2006#  2010
$HD   = 000000          13
$HIBTS  001000          29#
$HIOCT  023250        2007#*
$ICNT   001104          30#  1997*
$ILLUP  021610        2000#
$INTAG  001135          30#  2006
$ITEMB  001114          30#  1998*  1999
$LF     001172          30#  1998   2005   2006
$LFLG   023515        2008#*
$LPADR  001106          30#   136*   273*  1253*  1997*
$LPERR  001110          30#  1252*
$MADR1  001226          30#
$MADR2  001232          30#
$MADR3  001236          30#
$MADR4  001242          30#
$MAIL   001174          29    30#    136    1997   1998   2005
$MAMS1  001224          30#
$MAMS2  001230          30#
$MAMS3  001234          30#
$MAMS4  001240          30#
$MBADR  001002          29#
$MFLG   023514        2008#*
$MNEW   023137        2006#
$MSGAD  001210          30#  2008*
$MSGLG  001212          30#  2008*
$MSGTY  001174          30#  2008*
$MSWR   023126        2006#
$MTYP1  001225          30#
$MTYP2  001231          30#
$MTYP3  001235          30#
```

```
$MTYP4  001241            30#
$MXCNT  021156          1997#
$NULL   001154            30#   2005
$NWTST= 000001           272#    285#    293#    301#    311#    319#    327#    338#    346#    354#    366#    374#    382#
                         392#    422#    430#    444#    460#    475#    489#    497#    505#    513#    522#    532#    543#
                         561#    592#    618#    639#    654#    675#    683#    691#    700#    710#    718#    738#    739#
                         740#    741#    742#    759#    775#   1297#   1307#   1320#   1330#   1342#   1351#   1360#   1369#
                        1385#   1396#   1409#   1436#   1480#
$OCNT   022106          2004#*
$OMODE  022110          2004#*
$OVER   021142          1997#
$PASS   001202            30#    136*    223*    791*   1997
$PASTM  001006            29#
$PWRAD  021604          2000#
$PWRDN  021444           136    2000#
$PWRMG  021600          2000#
$PWRUP  021516          2000#
$QUES   001170            30#   1998    2005    2006
$RDCHR  022656          2006#   2010
$RDDEC= ****** U        2010
$RDLIN  022776          2006#   2010
$RDOCT  023150          2007#   2010
$RDSZ = 000010          2006#
$REGAD  001160            30#
$REGO   001162            30#
$REG1   001164            30#
$RTNAD  006600           791#
$R2A  = ****** U        2010
$SAVRE= ****** U        2010
$SAVR6  021614          2000#*
$SCOPE  020744           136    1997#
$SETUP= 000117            32#    135#    136     791    1997    1998    2006
$STUP = 177777            32#    135#
$SVLAD  021124          1997#
$SVPC = 000214            27#
$SWR  = 164400             5#     13      21      30     136     272     285     293     301     311     319     327     338
                         346     354     366     374     382     392     422     430     444     460     475     489     497
                         505     513     522     532     543     561     592     618     639     654     675     683     691
                         700     710     718     738     739     740     741     742     759     775     791    1297    1307
                        1320    1330    1342    1351    1360    1369    1385    1396    1409    1436    1480    1997    1998
                        2000
$SWREG  001216            30#   136
$SWRMK= 000000            21    1997
$TSTN   001200            30#   1997#
$TIMES  001166            30#    136#    301#    327#    354#    382#    422#    618#    639#    654#    675#    683#    691#
                         700#    710#    718#    738#    739#    740#    741#    742#    759#    775#    791#   1436#   1480#
                        1997#
$TKB    001146            30#   1586    1598    2006
$TKS    001144            30#   1584    1596    2006#
$TN   = 000072             6#     13     240     272#    285#    290     293#    298     301#    311#    316     319#    324
                         327#    338#    343     346#    351     354#    366#    371     374#    385     392#    392#    419
                         422#    427     430#    441     444#    455     460#    472     475#    486     489#    494     497#
                         502     505#    510     513#    518     523#    528     533#    540     543#    552     554     561#
                         571     580     583     592#    602     609     611     618#    620     628     630     639#    654#
                         665     675#    681     683#    689     691#    692     698     700#    701     707     710#    711
                         716     718#    719     724     738#    739#    740#    741#    742#    744     752     759#    761
```

```
                        775#  1251   1297#  1307#  1316   1320#  1327   1330#  1338   1342#  1349   1351#  1358
                       1360#  1367   1369#  1385#  1392   1396#  1409#  1436#  1480#
$TPB    001152           30#  2005*
$TPFLG  001157           30#  2005
$TPS    001150           30#  2005
$TRAP   023520          136   2010#
$TRP  = 000012         2010#
$TRPAD  023542         2010#
$TSTM   001004           29#
$TSTNM  001102           30#   787#   791#  1251#  1997#  1998
$TTYIN  023104         2006#
$TYPBN= ****** U       2010
$TYPOS  020520         1996#  2010
$TYPE   022112         2005#  2008   2010
$TYPEC  022324         2005#  2006
$TYPEX  022372         2005#
$TYPOC  021710         2004#  2010
$TYPON  021724         2004#  2010
$TYPOS  021664         2004#  2010
$UNIT   001206           30#   230#   234#   237    245    251    254    257#   777#   778   1960
$UNITM  001010           29#
$USWR   001220           30#
$VECT1  001244           30#   160    164
$VECT2  001246           30#
$XTSTR  020762         1997#
$$GET4= 000000          791#
$GFILL  022107         2004**
$4DCAT= ****** U       1997   1998
.     = 024152           23#    27#    28#    29#    30#   136    646    661    791    852    888    896    904
                         912    945    951    958   1157#  1318#  1699   1953#  1996#  1997   1998   1999#  2000
                        2005   2006#  2008#  2012#  2013#
.$ASTA= ****** U       2008
.$X   = 001000           29#
```

```
COMMEN   14#
CROSS   264#  1317  1434  1441  1447  1457  1463  1488  1496  1506  1514
CSPACE   26.#  1444  1451  1460  1467  1492  1500  1510  1518
ENDCOM   14#
ERROR    14#   239   256   282   291   299   307   317   325   333   344   352   360   372   380
            388   402   408   414   420   428   436   442   450   456   466   473   481   487   495
            503   511   519   529   541   551   559   569   579   588   600   608   616   627   635
            648   664   669   682   690   699   708   717   725   738   739   740   741   755   771
           1305  1315  1328  1339  1350  1359  1368  1376  1382  1393  1404  1416  1423  1431  1813
ESCAPE   14#
FIXAD1 1022#  1036  1162
GETPRI   14#
GETSWR   14#
LRXX   1934#  1937  1938  1939  1940  1941  1942  1943  1944
MULT     14#
NEWONE   726#  738   739   740   741
NEWTST   14#   272   285   293   301   311   319   327   338   346   354   366   374   382   392
            422   430   444   460   475   489   497   505   513   522   532   543   561   592   618
            639   654   675   683   691   700   710   718   738   739   740   741   742   759   775
           1297  1307  1320  1330  1342  1351  1360  1369  1385  1396  1409  1436  1480
POP      14#  1996  2000  2007  2008
PUSH     14#  1996  2000  2007  2008
REPORT   14#
SCOPE    14#   272   285   293   301   311   319   327   338   346   354   366   374   382   392
            422   430   444   460   475   489   497   505   513   522   532   543   561   592   618
            639   654   675   683   691   700   710   718   738   739   740   741   742   759   775
            791  1297  1307  1320  1330  1342  1351  1360  1369  1385  1396  1409  1436  1480
SETPRI   14#
SETTRA 2010#
SETUP    14#   136
SKIP     14#   235   240   253   255   280   290   298   306   316   324   332   343   351   359
            371   379   387   401   407   413   419   427   435   441   449   455   465   472   480
            486   494   502   510   518   528   540   550   552   554   568   571   578   580   583
            585   599   602   607   609   611   613   620   626   628   630   632   634   649   662
            665   670   681   689   692   698   701   707   711   716   719   724   738   739   740
            741   744   752   761   779  1304  1314  1316  1327  1338  1349  1358  1367  1375  1381
           1392  1403  1415  1422  1430  1812  1817
SLASH    14#
SPACE    14#
STARS    14#    27    29    30   272   285   293   301   311   319   327   338   346   354   366
            374   382   392   422   430   444   460   475   489   497   505   513   522   532   543
            561   592   618   639   654   675   683   691   700   710   718   738   739   740   741
            742   759   775   791  1297  1307  1320  1330  1342  1351  1360  1369  1385  1396  1409
           1436  1480  1996  1997  1998  1999  2000  2004  2005  2006  2007  2008  2010
SWRSU    14#  1368
TRMTRP 2010#
TSTSW9   268#   692   701   711   719
TYPBIN   14#
TYPDEC   14#   791
TYPNAM   14#
TYPNUM   14#
TYPOCS   14#
TYPOCT   14#  1999  2006
TYPTXT   14#  1318
$SCMRE   30#
$SCMTM   30#
```

N08

DZAACA.P11 CROSS REFERENCE TABLE -- MACRO NAMES

```
$$ESCA     14#
$$NEWT     14#   272   285   293   301   311   319   327   338   346   354   366   374   382   392
          422    430   444   460   475   489   497   505   513   522   532   543   561   592   618
          639    654   675   683   691   700   710   718   738   739   740   741   742   759   775
         1297   1307  1320  1330  1342  1351  1360  1369  1385  1396  1409  1436  1480
$$SET     2010#
$$SETM     136#
$$SKIP     14#   240   290   298   316   324   343   351   371   379   419   427   441   455   472
          486    494   502   510   518   528   540   552   554   571   580   583   602   609   611
          620    628   630   665   681   689   692   698   701   707   711   716   719   724   738
          739    740   741   744   752   761  1316  1327  1338  1349  1358  1367  1392
.EQUAT      7#    14
.HEADE      7#    13
.SETUP      8#    32   135
.SWRHI      9#    21
.SWRLO     21#
.$ACTI     11#    27
.$APTB     11#    30#
.$APTH     11#    29
.$APTY     11#  2008
.$CATC      7#    23
.$CMTA      7#    30
.$EOP       7#  1995
.$ERRO      7#  1998
.$ERRT      9#  1999
.$PARM      8#
.$POWE      8#  2000
.$POOC     10#  2007
.$READ      8#  2006
.$SAVE      8#
.$SCOP      9#  1997
.$SPAC      8#
.$SWOO      6#
.$TRAP      8#  2010
.$TYPD      9#  1996
.$TYPE      7#    8#  2005
.$TYPO      7#  2004
```

```
ADC     1840
AGO      156  163  233  780  781  782  783  784  785  786  849  885  893  946  947
         360 1015 1098 1102 1105 1175 1199 1231 1767 1775 1833 1867 1996 1999 2004
        2005 2006 2007 2008
ASL     1774 1792 1999 2006 2007 2010
ASLB    1996
ASR      308  334  361  389 1836 1837 1838 1839 2008
BCC     1996
BEQ      136  199  201  203  205  207  209  255  290  298  306  316  324  332  343
         351  359  371  379  387  401  407  413  419  427  435  441  449  455  465
         472  480  486  494  502  510  518  528  540  620  681  689  698  707  716
         724  738  739  740  741  761  779  791  817  829  840  868  874  883  925
         931  941  978  981  988 1036 1040 1049 1058 1082 1162 1168 1182 1218 1375
        1381 1415 1422 1430 1657 1663 1686 1725 1791 1997 1998 1999 2004 2005 2006
        2007 2008
BGE     1338 1349 1358 1403 1997
BGT      791 1996 2004 2006
BIC      161  165  196  791 1078 1426 1553 1587 1599 1731 1773 2004 2006 2007
BIS      252  624  841  884  942  979 1083 1086 1219 1377 1418 1659 1735 1745 1996
         2004 2006
BISB    1739 1999
BIT      619  692  701  711  719  738  739  740  741  743  760  816  828  839  845
         867  882  924  930  940  977  980  967  993 1036 1039 1048 1057 1081
        1162 1181 1217 1656 1662 1671 1685 1763 1997 1998
BITB     136  200  2008
BLE     1327 1347 1392 1864
BLOS    2006
BLT     1996 2004 2005 2006
BMI      242  554  568  583  599  611  630  749  766  865 1996
BNE      136  148  153  159  187  189  238  309  335  362  390  556  585  613  632
         634  646  661  692  701  711  719  744  751  768  807  846  854  890  998
         906  914  945  962  994 1005 1017 1071 1100 1104 1177 1191 1233 1554 1558
        1562 1566 1574 1578 1589 1601 1604 1667 1670 1672 1701 1730 1734 1741 1750
        1752 1764 1766 1817 1835 1996 1997 1998 1999 2000 2004 2005 2006 2008
BPL      550  578  607  626  852  888  896  904  912  945  951  958 1014 1085 1092
        1097 1585 1597 1644 1677 1699 1705 1832 1861 1996 1998 2004 2005 2006
BR       132  136  212  224  235  240  253  280  552  571  580  602  609  628  649
         662  665  670  752  824  836  837  858  877  916  934  964  984  995 1007
        1008 1065 1193 1194 1202 1228 1304 1314 1316 1318 1556 1560 1564 1568 1579
        1608 1624 1636 1689 1695 1736 1793 1812 1996 1997 1998 1999 2000 2004 2005
        2006 2007 2008
CLR      129  130  131  136  137  147  223  230  257  286  332  339  367  397  437
         451  468  482  546  574  604  622  647  652  663  667  668  673  694  703
         713  721  787  791  838  862  878  879  919  935  936  954  972  973  974
         975  976 1044 1053 1062 1230 1331 1352 1383 1406 1410 1411 1419 1440 1448
        1487 1495 1497 1575 1634 1661 1682 1748 1828 1996 1997 1999 2000 2004 2036
        2007
CLRB     525  537 1996 1997 2005 2006 2008
CMP      136  147  152  158  198  200  202  204  206  208  236  281  289  297  305
         315  323  331  342  350  358  372  378  386  390  406  412  418  426  434
         440  448  454  464  471  479  482  493  501  509  517  527  539  650  671
         680  688  697  706  738  739  740  741  853 1326 1337 1348 1357 1366 1374
        1380 1391 1402 1414 1421 1429 1537 1557 1561 1565 1588 1600 1603 1700 1777
        1863 1995 1997 2006
CMPF     254  778 1997 1998 2005 2006 2008
CO      1726
```

DEC 555 584 612 631 633 645 660 750 767 791 889 897 905 313 948
961 1016 1175 1190 1573 1577 1643 1666 1669 1729 1733 1740 1749 1751 1765
1790 1834 1999
DECB 2004 2005
EMT 14
HALT 23 1678 1998 2000 2005
INC 234 536 777 791 812 850 886 894 902 910 943 956 1012 1070 1095
1099 1103 1372 1697 1830 1996 1997 1998 2000 2004 2006 2008
INCB 746 764 1997 1998 2005
JOT 14
JMP 23 24 25 138 214 215 216 217 218 219 788 791 1066 1205 1535
1541 1590 1602 1778
JSR 185 258 791 821 822 823 833 834 835 864 915 921 963 970 995
997 1000 1002 1006 1035 1045 1054 1063 1064 1069 1161 1174 1188 1192 1199
1200 1201 1203 1216 1221 1223 1225 1227 1234 1242 1294 1299 1303 1309 1313
1317 1323 1334 1345 1354 1363 1388 1399 1434 1438 1441 1444 1447 1451 1454
1457 1460 1463 1467 1470 1475 1485 1488 1492 1496 1500 1503 1506 1510 1514
1518 1523 1529 1546 1549 1616 1618 1629 1631 1633 1635 1668 1703 1760 1807
1811 1997 1998 2005 2006 2008
MOV 133 136 141 142 143 145 150 151 154 155 160 162 166 192 195
229 231 245 251 273 274 283 287 288 294 295 298 301 302 303
304 313 314 320 321 322 327 328 329 330 340 341 347 348 349
354 355 356 357 368 369 375 376 377 382 383 384 385 393 394
395 396 398 399 404 405 410 411 416 417 422 423 425 431 432
433 438 439 445 446 447 452 453 461 462 463 469 470 476 477
478 483 484 490 491 492 498 499 500 506 507 508 514 515 516
523 524 526 533 534 535 538 545 547 548 557 558 563 564 565
566 573 575 586 587 594 595 596 597 603 605 614 615 618 621
623 639 640 641 642 643 644 651 654 655 656 657 658 659 666
672 675 676 677 679 683 684 685 687 691 693 696 700 702 705
710 712 715 718 720 723 738 739 740 741 742 745 747 753 754
759 762 763 769 770 775 791 803 814 815 818 819 820 826 827
830 831 832 842 843 844 847 848 856 857 863 865 866 869 870
871 872 875 876 881 892 900 908 920 922 923 926 927 928 929
932 933 939 953 955 969 982 983 985 986 989 990 991 992 1010
1011 1034 1036 1037 1038 1041 1042 1043 1046 1047 1050 1051 1052 1055 1056
1059 1060 1061 1067 1068 1077 1079 1080 1087 1088 1089 1093 1094 1101 1160
1162 1163 1164 1165 1166 1169 1170 1171 1172 1177 1178 1179 1180 1183 1184
1185 1186 1187 1198 1215 1220 1222 1224 1226 1241 1250 1251 1252 1253 1298
1301 1302 1308 1311 1312 1321 1322 1325 1332 1333 1336 1343 1344 1347 1353
1356 1361 1362 1365 1370 1371 1373 1378 1379 1386 1387 1390 1397 1398 1401
1412 1413 1420 1427 1428 1436 1456 1464 1480 1505 1513 1538 1545 1551 1552
1555 1559 1563 1567 1569 1571 1572 1576 1586 1598 1615 1619 1620 1621 1622
1623 1628 1632 1638 1639 1640 1641 1642 1658 1660 1683 1684 1687 1688 1690
1691 1692 1693 1694 1723 1728 1732 1738 1744 1746 1747 1761 1776 1789 1802
1803 1804 1805 1810 1824 1825 1827 1829 1841 1856 1857 1858 1859 1865 1866
1870 1871 1996 1997 1998 1999 2000 2004 2005 2006 2007 2008 2010
MOVB 1084 1090 1724 1727 1996 1997 1999 2004 2005 2006 2007 2008 2010
MTS 1862 1996 2004
MUL 260 577 791 1036 1162 1236 1237 2016
RESET 134 424 678 686 695 704 714 722 738 739 740 741 773 776 791
1204
ROL 2004 2007
ROLB 1091
RTI 136 1996 1997 1998 2000 2004 2005 2006 2007
RTS 168 855 1019 1072 1106 1235 1591 1605 1645 1679 1706 1742 1753 1768 1794

```
        1842  1868  1872  1999  2005  2008  2010
SUB      167   901   909   959  1696  1704  1815  1816  1860  1996  1996  20uu
SWAB    1826
TRAP    2010
TST      146   157   186   188   232   237   241   275   276   277   278   279   806  1004  1018
        1232  1676  1996  1997  1998  1999  2004  2005  2006  2007  2008  2010
TSTB     549   553   567   576   582   598   606   610   625   629   748   765   851   887   895
         903   911   944   950   957  1013  1084  1096  1584  1596  1664  1698  1831  1996  1997
        2005  2006  2008
.ASCII    30   178   179   180   181   182   183  1877  1878  1879  1880  1881  1882  1883  1884
        1888  1889  1890  1891  1925  1926  1929  1931  1948  1951
.ASCIZ    30   791  1318  1985  1886  1887  1892  1893  1894  1895  189e  1897  1898  1899  1900
        1901  1902  1903  1904  1905  1906  1907  1908  1909  1910  1915  1916  1919  1921  1923
        1932  1937  1938  1939  1940  1941  1942  1943  1944  1999  2001  2006
.BLKB   2006
.BLKW   1996  2012  2013
.BYTE     30   247   248   791  1117  1118  1119  1120  1121  1122  1123  1124  1125  1126  1127
        1128  1129  1130  1131  1132  1133  1134  1135  1136  1137  1138  1139  1140  1141  1142
        1143  1144  1145  1146  1147  1148  1149  1150  1151  1152  1153  1154  1155  1912  1913
        1914  1917  1918  1920  1922  1924  1927  1928  1930  1947  1949  1950  1952  1998  2004
        2006  2008
.DSABL  2006
.ENABL     4  2006
.END    2017
.ENDC     13    14    21    23    27    29    30    32   135   136   235   240   253   255   272
         280   285   290   293   298   301   306   311   316   319   324   327   332   338   343
         346   351   354   359   366   371   374   379   382   387   392   401   407   413   419
         422   427   430   435   441   444   449   455   460   465   472   475   480   486   489
         494   497   502   505   510   513   518   522   528   532   540   543   550   552   554
         561   568   571   578   580   583   585   592   599   602   607   609   611   613   618
         620   626   628   630   632   634   639   649   654   662   665   670   675   681   683
         689   691   692   698   700   701   707   710   711   716   718   719   724   738   739
         740   741   742   744   752   759   761   775   779   791  1297  1304  1307  1314  1316
        1318  1320  1327  1330  1338  1342  1349  1351  1358  1360  1367  1369  1375  1381  1385
        1392  1396  1403  1409  1415  1422  1430  1436  1480  1812  1817  1996  1997  1998  1999
        2000  2004  2005  2006  2007  2008  2010
.EQUIV    14
.EVEN     30  1157  1318  1953  1999  2002  2008
.IF       13    14    21    23    27    29    30    32   135   136   235   240   253   255   272
         280   285   290   293   298   301   306   311   316   319   324   327   332   338   343
         346   351   354   359   366   371   374   379   382   387   392   401   407   413   419
         422   427   430   435   441   444   449   455   460   465   472   475   480   486   489
         494   497   502   505   510   513   518   522   528   532   540   543   550   552   554
         561   568   571   578   580   583   585   592   599   602   607   609   611   613   618
         620   626   628   630   632   634   639   649   654   662   665   670   675   681   683
         689   691   692   698   700   701   707   710   711   716   718   719   724   738   739
         740   741   742   744   752   759   761   775   779   791  1297  1304  1307  1314  1316
        1318  1320  1327  1330  1338  1342  1349  1351  1358  1360  1367  1369  1375  1381  1385
        1392  1396  1403  1409  1415  1422  1430  1436  1480  1812  1817  1996  1997  1998  1999
        2000  2004  2005  2006  2007  2008  2010
.IFF      14    21    27    29    30   136   235   240   253   255   272   280   285   290   233
         298   301   306   311   316   319   324   327   332   338   343   346   351   354   359
         366   371   374   379   382   387   392   401   407   413   419   422   427   430   435
         441   444   449   455   460   465   472   475   480   486   489   494   497   502   505
         510   513   518   522   528   532   540   543   550   552   554   561   568   571   579
         580   583   585   592   599   602   607   609   611   613   619   620   626   628   630
```

E09


```
            632   634   639   649   654   662   665   670   675   681   683   689   691   692   698
            700   701   707   710   711   716   718   719   724   738   739   740   741   742   744
            752   759   761   775   779   791  1297  1304  1307  1314  1316  1320  1327  1330  1338
           1342  1349  1351  1358  1360  1367  1369  1375  1381  1385  1392  1396  1403  1409  1415
           1422  1430  1436  1480  1812  1817  1996  1997  1998  1999  2000  2004  2005  2006  2007
           2008  2010
.IFT       1318  1997  1998  2006  2007
.IFTF      1319  1997  1998  2006  2007
.IIF         13    21    23    30   136   791  1997  1998  1999  2005  2006  2010
.IRP         32   135   272   285   293   301   311   319   327   338   346   354   366   374   382
            392   422   430   444   460   475   489   497   505   513   522   532   543   561   592
            618   639   654   675   683   691   700   710   718   738   739   740   741   742   759
            775  1297  1307  1320  1330  1342  1351  1360  1369  1385  1396  1409  1436  1480  1996
           1997  1998  2000  2007  2008
.LIST         2    12    14    21    23    30    32   135   136   174   272   285   293   301   311
            319   327   338   346   354   366   374   382   392   422   430   444   460   475   489
            497   505   513   522   532   543   561   592   618   639   654   675   683   691   700
            710   718   738   739   740   741   742   759   775   791   804  1031  1212  1292  1297
           1307  1318  1320  1330  1342  1351  1360  1369  1385  1396  1409  1436  1480  1651  1997
           1998  2006  2010
.MACRO       21    30   136   261   264   269   726  1022  1934  2010
.MCALL        7     8     9    10    11    14    30   136
.MEXIT       30
.NLIST        1     3    14    21    23    30    32   135   136   169   272   285   293   301   311
            319   327   338   346   354   366   374   382   392   422   430   444   460   475   489
            497   505   513   522   532   543   561   592   618   639   654   675   683   691   700
            710   718   738   739   740   741   742   759   775   791   793  1021  1207  1254  1297
           1307  1318  1320  1330  1342  1351  1360  1369  1385  1396  1409  1436  1480  1647  1997
           1998  2006  2010
.PAGE        30
.REPT        23    30
.SBTTL       14    21    23    27    29    30   136   140   149   170   171   172   173   177   227
            272   285   293   301   311   319   327   338   346   354   366   374   382   392   422
            430   444   460   475   489   497   505   513   522   532   543   561   592   618   639
            654   675   683   691   700   710   718   738   739   740   741   742   759   775   791
            794   795   796   797   798   799   800   801   802   803   813   825   860   917   967
           1032  1158  1196  1208  1209  1210  1211  1213  1240  1297  1307  1320  1330  1342  1351
           1360  1369  1385  1396  1409  1436  1480  1543  1613  1626  1648  1649  1650  1654  1681
           1756  1875  1996  1997  1998  1999  2000  2004  2005  2006  2007  2008  2010
.TITLE       13
.WORD        23    27    29    30   791  1999  2000  2004  2005  2007  2008


ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*DZAACA,DZAACA.ASS/CRF/PAGNUM=DSKZ:DZAACA.P11
RUN-TIME:  76 43 8 SECONDS
RUN-TIME RATIO:  416/129=3.2
CORE USED:  27K  (53 PAGES)
```

F09

Spooler [illegible] 14 Seconds, [illegible] MCR, [illegible] disk reads, 2 disk writes, 10[illegible] pages

[illegible]

[illegible] 11111111111111111111111111110

[illegible] 111111111122222222223331

[illegible] 567890123456789012 [illegible]

[illegible] 1111111111111111111111111111110

[illegible] 000000000111111111122222222223331

```
                THE SOFT  ...B1    1205  011360  000137   ...B5
        OVER THE PROGRAM  ...C1    1249                   ...C5
                1.  TYPE  ...D1    1326  011762  023737   ...D5
                    AND   ...E1    1348  012076  023737   ...E5
                SWITCH    ...F1    1391  012352  023737   ...F5
             -------- ---  ...G1    1415  012470  001401   ...G5
                1.  TYPI  ...H1    1441  012612  004537   ...H5
        IS INCREMENTED W  ...I1    1485  012764  004537   ...I5
        PHOSPHOR AND ERA  ...J1    1530  013134  001454   ...J5
        WHICH TIME THE P  ...K1    1552  013216  010001   ...K5
136             INITIALI  ...L1    1590  013364  000137   ...L5
796             END OF P  ...M1    1621  013502  010077   ...M5
(3)             TRAP TAB  ...N1    1663  013652  001424   ...N5

(1)                       ...B2    1718                   ...B6
              000004      ...C2    1765  014234  005337   ...C6
 16           170416      ...D2    1807  014376  004537   ...D6
(1)                       ...E2    1831  014506  105777   ...E6
(1)           000214      ...F2          014734  041501   ...F6
(1)  C01102      000      ...G2          015426  047122   ...G6
(2)                       ...H2    1895  016104  040504   ...H6
(1)                       ...I2    1906  016602  042523   ...I6
 80  001352  020346       ...J2    1924  017232     015   ...J6
(1)                       ...K2      .1) 017704  051525   ...K6
149                       ...L2    1956  020310  001116   ...L6
186  002132  005737       ...M2     (1)                   ...M6
236  002364  022626       ...N2     (1)  020732  000012   ...N6

291  002600  104002       ...B3    1998                   ...B7
317  002744  104003       ...C3     (2)  021332  104401   ...C7
343  003106  001401       ...D3     .3)  021542  012677   ...D7
370  003244  023737       ...E3     (1)  021710  112737   ...E7
398  003424  012737       ...F3     (1)                   ...F7
427  003610  001401       ...G3     (1)  022322  000724   ...G7
465  004026  001401       ...H3     (1)  022522  000757   ...H7
495  004210  104001       .. '3     (1)  022756  026627   ...I7
528  004402  001401       ...J3     (1)  023166  104410   ...J7
549  004510  105737       ...K3     (1)  023430           ...K7
598  004730  105737       ...L3     (1)  023554  022444   ...L7
644  005162  012777       ...M3  ADDW1 = 000000           ...M7
680  005374  023737       ...N3  BEGIN   001462           ...N7

714  005636  000005       ...B4  CH12  = 000012           ...B8
(1)  006120  023737       ...C4  DH1     016645           ...C8
764  006332  105277       ...D4  IOTVEC= 000020           ...D8
(1)  006500               ...E4  PIC7    011022           ...E8
814  006650  013700       ...F4                           ...F8
869  007146  013701       ...G4  SW1   = 000002           ...G8
925  007374  001404       ...H4  TST36   005124           ...H8
975  007574  005077       ...I4                           ...I8
(1)  010050  013737       ...J4  $FFLG   023516           ...J8
1084 010370  105777       ...K4  $OMODE  022110           ...K8
1121 010542     177       ...L4  $TSTNM  001102           ...L8
1149 010756      076      ...M4  ESCAPE     14*   1315    ...M8
1163 011076  013705       ...N4                620    628 ...N8

IOT              14      764     ...C9
.ASCII           30      178     ...D9
.IRP             32      135     ...E9
 **END** USER DAVIES,TOM         ...F9
```