AA11-K

DIAGNOSTIC

MD-11-DZAAC-B

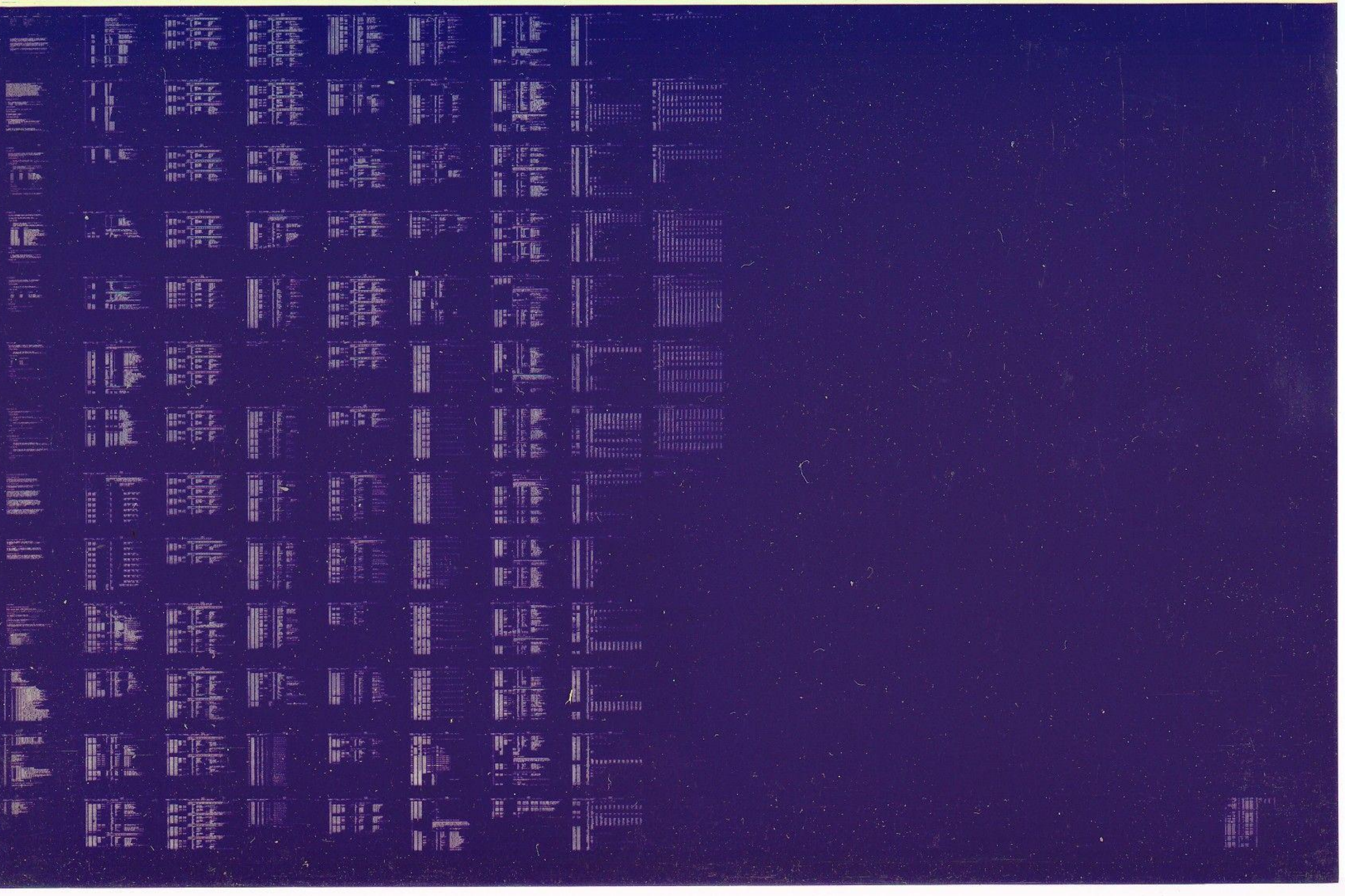
EP DZAAC B DL B

FICHE 1 OF 1

MAR 1977

digital

MADE IN USA



PRODUCT CODE: MAINDEC-11-DZAAC-B

PRODUCT NAME: AA11-K DIAGNOSTIC TEST

DATE: JANUARY 1977

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.

7.0 TEST PROCEDURE

7.1 AUTO LOGIC TEST

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

B. STARTING SEQUENCE

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

C. CONTROL SWITCHES

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

SWITCH	OCTAL	FUNCTION
SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON CURRENT TEST
SW13=1	020000	INHIBIT ERROR TYPEOUT
SW12=1	010000	STORAGE SCOPE CONNECTED
SW11=1	004000	INHIBIT TEST INTERATIONS
SW10=1	002000	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. TYPE 'AD' TO RUN THE AUTO VISUAL DISPLAY TEST. THE PROGRAM WILL THEN EXECUTE THE VISUAL DISPLAY TEST.

C. CONTROL SWITCHES

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

CONSOLE SWITCHES	FUNCTION
CONSOLE SW10=1	SELECTE 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.
THE "AUTO-CALIBRATION (AC)" MUST HAVE BEEN RUN PRIOR TO RUNNING THIS SECTION WHEN ON THE "AUTO HARDWARE TESTER".

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

SEQ 0006

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

3. 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) THEY Y IS INCREMENTED TO POSITIVE FULL SCALE (RIGHT LINE) THEN X IS DECREMENTED TO NEGATIVE FULL SCALE (TOP LINE) AND FINALLY Y IS DECREMENTED TO NEGATIVE FULL SCALE (LEFT LINE). MODE 01 (INTENSIFY ON LOADING X) AND MODE 10 (INTENSIFY ON LOADING Y) ARE USED.

DISPLAY X

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

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.

DISPLAY ALPHA-NUMERIC CHARACTER SET

SEQ 0009

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 DESCENDING 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 PROGRAM MUST BE RESTARTED AT 200.

9.2 XXDP/APT NOTES

THIS DIAGNOSTIC IS CHAINABLE UNDER XXDP.
THIS DIAGNOSTIC HAS THE "AP"" 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-B      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-C2), SEPT 14, 1976.
                                ;*
                                .SBTTL  BASIC DEFINITIONS

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

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

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

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

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
```

```
(1)             020000                  SW13=   20000
(1)             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
```

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

                  000000                        .=0
                                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                  000174                        .=174
000174  000000                         DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                         SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                        .SBTTL  STARTING ADDRESS(ES)
000200  000137  001462                         JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  001500                         JMP     BEGIN1                  ;JUMP TO RESTART ADDRESS
000210  000137  013434                         JMP     DYNCAL                  ;JUMP TO DYNAMIC DAC CALIBRATION
```

```
 27                                  .SBTTL  ACT11 HOOKS
(1)
(2)                                  ;;*****************************************************************
(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                          .=$SVPC                 ;; RESTORE PC
 28          001000                          .=1000
 29                                  .SBTTL  APT PARAMETER BLOCK
(1)
(2)                                  ;;*****************************************************************
(1)                                  ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(2)                                  ;;*****************************************************************
(1)          001000                          .$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)
```

```
                                .SBTTL  COMMON TAGS

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

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

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

```
(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   ;;3ASE 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                             $ERRTB:
 31
 33
 34                                     ;ITEM   1
 35  001256  015750                             EM1                     ;AA11-K STATUS REGISTER IN ERROR
 36  001260  016571                             DH1                     ;ERRPC  IBASE   GOOD    BAD
 37  001262  020236                             DT1                     ;$ERRPC STAT    $GDDAT  $BDDAT
 38  001264  020364                             DF0
 39
 40                                     ;ITEM   2
 41  001266  016001                             EM2                     ; DACO REGISTER IN ERROR
 42  001270  016571                             DH1                     ;ERRPC  IBASE   GOOD    BAD
 43  001272  020250                             DT2                     ;$ERRPC DACO    $GDDAT  $BDDAT
 44  001274  020364                             DF0
 45
 46                                     ;ITEM   3
 47  001276  016030                             EM3                     ; DAC1 REGISTER IN ERROR
 48  001300  016571                             DH1                     ;ERRPC  IBASE   GOOD    BAD
 49  001302  020262                             DT3                     ;$ERRPC DAC1    $GDDAT  $BDDAT
 50  001304  020364                             DF0
 51
 52                                     ;ITEM   4
 53  001306  016057                             EM4                     ;DAC #2 REGISTER IN ERROR
 54  001310  016571                             DH1                     ;ERRPC  IBASE   GOOD    BAD
 55  001312  020274                             DT4                     ;$ERRPC DAC2    $GDDAT  $BDDAT
 56  001314  020364                             DF0
 57
 58                                     ;ITEM   5
 59  001316  016106                             EM5                     ;DAC #3 REGISTER IN ERROR
 60  001320  016571                             DH1                     ;ERRPC  IBASE   GOOD    BAD
 61  001322  020306                             DT5                     ;$ERRPC DAC3    $GDDAT  $BDDAT
 62  001324  020364                             DF0
 63
 64                                     ;ITEM   6
 65  001326  016135                             EM6                     ;AA11-K FAILED TO INTERRUPT
 66  001330  016625                             DH6                     ;ERRPC  IBASE
 67  001332  020320                             DT6                     ;$ERRPC STAT
 68  001334  020364                             DF0
 69
```

```
                                ;ITEM   7
001336  016170                          EM7             ;AA11-K INTERRUPTED IN ERROR
001340  016625                          DH6             ;ERRPC  IBASE
001342  020320                          DT6             ;$ERRPC STAT
001344  020364                          DFO

                                ;ITEM   10
001346  016224                          EM10            ;BUS TIME-OUT ON AA11-K ADDRESSES
001350  016625                          DH6             ;ERRPC  IBASE
001352  020320                          DT6             ;$ERRPC STAT
001354  020364                          DFO

                                ;ITEM   11
001356  016275                          EM11            ;AA11-K READY BIT ERROR
001360  016571                          DH1             ;ERRPC  IBASE   GOOD    BAD
001362  020236                          DT1             ;$ERRPC STAT    $GDDAT  $BDDAT
001364  020364                          DFO

                                ;ITEM   12
001366  016324                          EM12            ;POWER SUPPLY VOLTAGE WAS INCORRECT
001370  016673                          DH14            ;ERRPC  BASE    GOOD    BAD
001372  020340                          DT14            ;$ERRPC $BASE   $GDDAT  $BDDAT
001374  020364                          DFO

                                ;ITEM   13
001376  016367                          EM13            ;A PREVIOUSLY EXISTING AA11-K DOES NOT RESPOND NOW
001400  016641                          DH13            ;ERRPC  BASE    FIRST#  #NOW
001402  020326                          DT13            ;$ERRPC $BASE   EVER    $UNIT
001404  020364                          DFO

                                ;ITEM   14
001406  016410                          EM14            ;AA11-K LOGIC SIGNAL HIGH OUTPUT TO LOW
001410  016673                          DH14            ;ERRPC  BASE    GOOD    BAD
001412  020340                          DT14            ;$ERRPC $BASE   $GDDAT  $BDDAT  SPREAD
001414  020364                          DFO

                                ;ITEM   15
001416  016457                          EM15            ;AA11-K LOGIC LOW OUTPUT TO HIGH
001420  016673                          DH14            ;ERRPC  BASE    GOOD    BAD
001422  020340                          DT14            ;$ERRPC $BASE   $GDDAT  $BDDAT  SPREAD
001424  020364                          DFO

                                ;ITEM   16
001426  016526                          EM16            ;SELECTED DAC HAS A LINEARITY ERROR
001430  016720                          DH16            ;ERRPC  DACADR  GOOD    BAD
001432  020352                          DT16            ;$ERRPC XDACUT  $GDDAT  $BDDAT
001434  020364                          DFO
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
```

```
  125  001454  170426                    DAC3:   170426
  126  001456  000350                    IV:     350
  127  001460  000352                    IVS:    352
  128
  129  001462  005037  020416            BEGIN:  CLR     TEMP
  130  001466  005037  020426                    CLR     EVER            ;RESET POINTER
  131  001472  005037  020430                    CLR     EVER1
  132  001476  000403                            BR      BEG
  133  001500  012737  000001  020416    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  020730  000020            MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
  (1)  001536  012737  000340  000022            MOV     #340,@#IOTVEC+2 ;;LEVEL 7
  (1)  001544  012737  021144  000030            MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
  (1)  001552  012737  000340  000032            MOV     #340,@#EMTVEC+2 ;;LEVEL 7
  (1)  001560  012737  023504  000034            MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)  001566  012737  000340  000036            MOV     #340,@#TRAPVEC+2;LEVEL 7
  (1)  001574  012737  021430  000024            MOV     #$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$:     MOV     R2,(R1)+                ;LOAD .+2
001754  005021                          CLR     (R1)+                   ;LOAD HALT
001756  010102                          MOV     R1,R2                   ;LOAD R2
001760  005722                          TST     (R2)+                   ;BUMP R2
001762  020227  001002                  CMP     R2,#1002                ;TEST FOR LAST
001766  001371                          BNE     5$                      ;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     #DACO,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  020422          MOVB    $VECT1+1,BRLEV1         ;LOAD BR LEVEL
002066  042737  177437  020422          BIC     #177437,BRLEV1
002074  013737  020422  020424          MOV     BRLEV1,BRLEV2
002102  162737  000040  020424          SUB     #40,BRLEV2
002110  000207                          RTS     PC                      ;EXIT
175
```

```
                                        .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  020416                  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  104401                          TYPE                    ;CALL MESSAGE PRINTER VIA 'EMT'
002150  014606                          TITLE                   ;TYPE PROGRAM HEADER.
002152  012706  001100          WAITIN: MOV     #STACK,SP       ;RESET STACK
002156  104407                          CKSWR                   ;TEST FOR CTRL G
002160  104411                          RDLIN                   ;READ TWO CHARACTERS
002162  013637  020432                  MOV     @(SP)+,OPRIN    ;READ THE CHARACTER
002166  042737  000000  020432          BIC     #0,OPRIN        ;MASK OTHER CHARACTERS

002174  023737  002112  020432          CMP     AL,OPRIN        ;TEST FOR "AL"
002202  001427                          BEQ     10$             ;BR IF YES
002204  023737  002114  020432          CMP     AD,OPRIN        ;TEST FOR "AD"
002212  001425                          BEQ     11$             ;BR IF YES
002214  023737  002116  020432          CMP     AC,OPRIN        ;TEST FOR "AC"
002222  001423                          BEQ     12$             ;BR IF YES
002224  023737  002120  020432          CMP     ML,OPRIN        ;TEST FOR "ML"
002232  001421                          BEQ     13$             ;BR FI YES
002234  023737  002122  020432          CMP     MD,OPRIN        ;TEST FOR "MD"
002242  001417                          BEQ     14$             ;BR IF YES
002244  023737  002124  020432          CMP     MC,OPRIN        ;TEST FOR "MC"
002252  001415                          BEQ     15$             ;BR IF YES
002254  104401                          TYPE
002256  016746                          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  011506          12$:    JMP     AUTCAL          ;RUN THE AUTO CALIBRATION (FACTORY ONLY)
002276  000137  013104          13$:    JMP     MANUL           ;RUN THE MANUAL LOGIC TEST
002302  000137  011372          14$:    JMP     FULRMP          ;RUN THE RAMP PATTERN ON FOUR DAC'S
002306  000137  013370          15$:    JMP     CALDAC          ;RUN THE MANUAL CAL OF THE DAC

002312  104401                  CTRLC:  TYPE
002314  016754                          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  020426          3$:     TST     EVER            ;TEST IF # HAS BEEN REPORTED
002404  100417                          BMI     4$              ;BR IF IT HAS
002406  104401                          TYPE
002410  016762                          FOUND1                  ;TELL OPERATOR THE # OF AA11-K'S
002412  013746  001206                  MOV     $UNIT,-(SP)
002416  104403                          TYPOS
002420     002                          .BYTE   2
002421     000                          .BYTE   0
002422  104401                          TYPE
002424  017006                          FOUND2
002426  013737  001206  020426          MOV     $UNIT,EVER      ;SAVE THE # OF AA11-K'S FOR LATER
002434  052737  100000  020426          BIS     #BIT15,EVER     ;SET "REPORTED # FLAG"
002442  000405                          BR      5$              ;;
002444  123737  020426  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
```

```
                                        ;;*****************************************************************
                                        ;*TEST 2        TEST THAT THE DACO REGISTER CAN BE CLEARED
                                        ;;*****************************************************************
002546  000004                  TST2:   SCOPE
002550  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED
002554  013777  001124  176664          MOV     $GDDAT,@DACO            ;LOAD REG
002562  017737  176660  001126          MOV     @DACO,$BDDAT            ;READ REG
002570  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
002576  001401                          BEQ     TST3            ;;BR IF EQUAL
002600  104002                          ERROR   2                       ;ERROR, DACO REGISTER NOT = 0

                                        ;;*****************************************************************
                                        ;*TEST 3        TEST THAT THE DACO REGISTER CAN BE LOADED WITH #7777
                                        ;;*****************************************************************
002602  000004                  TST3:   SCOPE
002604  012737  007777  001124          MOV     #7777,$GDDAT            ;LOAD EXPECTED
002612  013777  001124  176626          MOV     $GDDAT,@DACO            ;LOAD REG
002620  017737  176622  001126          MOV     @DACO,$BDDAT            ;READ REG
002626  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
002634  001401                          BEQ     TST4            ;;BR IF EQUAL
002636  104002                          ERROR   2                       ;ERROR,  DACO REGISTER NOT = 7777

                                        ;;*****************************************************************
                                        ;*TEST 4        TEST THAT THE DACO REGISTER CAN HOLD A FLOATING 1 PATTERN
                                        ;;*****************************************************************
002640  000004                  TST4:   SCOPE
002642  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
002650  012737  004000  001124          MOV     #BIT11,$GDDAT           ;LOAD EXPECTED
002656  013777  001124  176562  1$:     MOV     $GDDAT,@DACO            ;LOAD DACO REGISTER
002664  017737  176556  001126          MOV     @DACO,$BDDAT            ;READ THE REGISTER
002672  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE DATA
002700  001401                          BEQ     2$              ;;BR IF SAME
002702  104002                          ERROR   2                       ;ERROR, DACO REGISTER FAILED TO HOLD A FLOATING
002704  006237  001124          2$:     ASR     $GDDAT                  ;CHANGE THE DATA
002710  001362                          BNE     1$                      ;BR AND TEST MORE DATA
```

```
311                                     ;;*****************************************************************
(3)                                     ;*TEST 5         TEST THAT THE DAC #1 REGISTER CAN BE CLEARED
(3)                                     ;;*****************************************************************
(2)  002712  000004                     TST5:   SCOPE
312  002714  005037  001124                        CLR     $GDDAT                          ;LOAD EXPECTED
313  002720  013777  001124  176522                MOV     $GDDAT,@DAC1                    ;LOAD Y REG
314  002726  017737  176516  001126                MOV     @DAC1,$BDDAT                    ;READ REG
315  002734  023737  001124  001126                CMP     $GDDAT,$BDDAT                   ;COMPARE
316  002742  001401                                BEQ     TST6            ;;BR IF EQUAL
317  002744  104003                                ERROR   3                               ;ERROR,  DAC1 REGISTER NOT = 0
318
319                                     ;;*****************************************************************
(3)                                     ;*TEST 6         TEST THAT THE Y REGISTER CAN BE LOADED WITH #7777
(3)                                     ;;*****************************************************************
(2)  002746  000004                     TST6:   SCOPE
320  002750  012737  007777  001124                MOV     #7777,$GDDAT                    ;LOAD EXPECTED
321  002756  013777  001124  176464                MOV     $GDDAT,@DAC1                    ;LOAD Y REG
322  002764  017737  176460  001126                MOV     @DAC1,$BDDAT                    ;READ REG
323  002772  023737  001124  001126                CMP     $GDDAT,$BDDAT                   ;COMPARE
324  003000  001401                                BEQ     TST7            ;;BR IF EQUAL
325  003002  104003                                ERROR   3                               ;ERROR,  DAC1 REGISTER NOT = 7777
326
327                                     ;;*****************************************************************
(3)                                     ;*TEST 7         TEST THAT THE Y REGISTER CAN HOLD A FLOATING 1 PATTERN
(3)                                     ;;*****************************************************************
(2)  003004  000004                     TST7:   SCOPE
(1)  003006  012737  000100  001166            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
328  003014  012737  004000  001124            MOV     #BIT11,$GDDAT                   ;LOAD EXPECTED
329  003022  013777  001124  176420  1$:       MOV     $GDDAT,@DAC1                    ;LOAD THE REGISTER
330  003030  017737  176414  001126            MOV     @DAC1,$BDDAT                    ;READ THE REGISTER
331  003036  023737  001124  001126            CMP     $GDDAT,$BDDAT                   ;COMPARE THE DATA
332  003044  001401                            BEQ     2$              ;.BR IF DATA IS SAME
333  003046  104003                            ERROR   3                               ;ERROR, DAC1 REGISTER FAILED TO HOLD A FLOATING
334  003050  006237  001124          2$:       ASR     $GDDAT                          ;CHANGE THE DATA
335  003054  001362                            BNE     1$                              ;BR AND TEST MORE DATA
```

```
                                ;;*****************************************************************
                                ;*TEST 10       TEST THAT THE DAC #2 REGISTER CAN BE CLEARED
                                ;;*****************************************************************
(2)  003056  000004             TST10:  SCOPE
339  003060  005037  001124            CLR     $GDDAT                      ;LOAD EXPECTED
340  003064  013777  001124  176360    MOV     $GDDAT,@DAC2                ;LOAD REG
341  003072  017737  176354  001126    MOV     @DAC2,$BDDAT                ;READ REG
342  003100  023737  001124  001126    CMP     $GDDAT,$BDDAT               ;COMPARE
343  003106  001401                    BEQ     TST11           ;;BR IF EQUAL
344  003110  104004                    ERROR   4                           ;ERROR, DAC #2 REGISTER NOT = 0
345
346                                 ;;*****************************************************************
                                ;*TEST 11       TEST THAT THE DAC #2 REGISTER CAN BE LOADED WITH #7777
                                ;;*****************************************************************
(2)  003112  000004             TST11:  SCOPE
347  003114  012737  007777  001124    MOV     #7777,$GDDAT                ;LOAD EXPECTED
348  003122  013777  001124  176322    MOV     $GDDAT,@DAC2                ;LOAD REG
349  003130  017737  176316  001126    MOV     @DAC2,$BDDAT                ;READ REG
350  003136  023737  001124  001126    CMP     $GDDAT,$BDDAT               ;COMPARE
351  003144  001401                    BEQ     TST12           ;;BR IF EQUAL
352  003146  104004                    ERROR   4                           ;ERROR, DAC #2 REGISTER NOT = 7777
353
354                                 ;;*****************************************************************
                                ;*TEST 12       TEST THAT THE DAC #2 REGISTER CAN HOLD A FLOATING 1 PATTERN
                                ;;*****************************************************************
(2)  003150  000004             TST12:  SCOPE
(1)  003152  012737  000100  001166    MOV     #100,$TIMES     ;;DO 100 ITERATIONS
355  003160  012737  004000  001124    MOV     #BIT11,$GDDAT               ;LOAD EXPECTED
356  003166  013777  001124  176256  1$:  MOV   $GDDAT,@DAC2               ;LOAD X REGISTER
357  003174  017737  176252  001126    MOV     @DAC2,$BDDAT                ;READ THE REGISTER
358  003202  023737  001124  001126    CMP     $GDDAT,$BDDAT               ;COMPARE THE DATA
359  003210  001401                    BEQ     2$              ;;BR IF SAME
360  003212  104004                    ERROR   4                           ;ERROR, DAC #2 REGISTER FAILED TO HOLD A FLOATIN
361  003214  006237  001124          2$:  ASR   $GDDAT                      ;CHANGE THE DATA
362  003220  001362                    BNE     1$                          ;BR AND TEST MORE DATA
```

```
                                ;;*****************************************************************
                                ;*TEST 13       TEST THAT THE DAC #3 REGISTER CAN BE CLEARED
                                ;;*****************************************************************
(2) 003222  000004              TST13:  SCOPE
367 003224  005037  001124              CLR     $GDDAT                  ;LOAD EXPECTED
368 003230  013777  001124  176216      MOV     $GDDAT,@DAC3            ;LOAD REG
369 003236  017737  176212  001126      MOV     @DAC3,$BDDAT            ;READ REG
370 003244  023737  001124  001126      CMP     $GDDAT,$BDDAT           ;COMPARE
371 003252  001401                      BEQ     TST14           ;;BR IF EQUAL
372 003254  104005                      ERROR   5                       ;ERROR, DAC #3 REGISTER NOT = 0
373
374                             ;;*****************************************************************
                                ;*TEST 14       TEST THAT THE DAC #3 REGISTER CAN BE LOADED WITH #7777
                                ;;*****************************************************************
(2) 003256  000004              TST14:  SCOPE
375 003260  012737  007777  001124      MOV     #7777,$GDDAT            ;LOAD EXPECTED
376 003266  013777  001124  176160      MOV     $GDDAT,@DAC3            ;LOAD REG
377 003274  017737  176154  001126      MOV     @DAC3,$BDDAT            ;READ REG
378 003302  023737  001124  001126      CMP     $GDDAT,$BDDAT           ;COMPARE
379 003310  001401                      BEQ     TST15           ;;BR IF EQUAL
380 003312  104005                      ERROR   5                       ;ERROR, DAC #3 REGISTER NOT = 7777
381
382                             ;;*****************************************************************
                                ;*TEST 15       TEST THAT THE DAC #3 REGISTER CAN HOLD A FLOATING 1 PATTERN
                                ;;*****************************************************************
(2) 003314  000004              TST15:  SCOPE
(1) 003316  012737  000100  001166      MOV     #100,$TIMES     ;;DO 100 ITERATIONS
383 003324  012737  004000  001124      MOV     #BIT11,$GDDAT           ;LOAD EXPECTED
384 003332  013777  001124  176114  1$: MOV     $GDDAT,@DAC3            ;LOAD DAC #3 REGISTER
385 003340  017737  176110  001126      MOV     @DAC3,$BDDAT            ;READ THE REGISTER
386 003346  023737  001124  001126      CMP     $GDDAT,$BDDAT           ;COMPARE THE DATA
387 003354  001401                      BEQ     2$              ;;BR IF SAME
388 003356  104005                      ERROR   5                       ;ERROR, DAC #3 REGISTER FAILED TO HOLD A FLOATIN
389 003360  006237  001124          2$: ASR     $GDDAT                  ;CHANGE THE DATA
390 003364  001362                      BNE     1$                      ;BR AND TEST MORE DATA
```

```
                                        ;;*****************************************************************
                                        ;*TEST 16       TEST THAT THE FOUR DAC REGISTERS CAN HOLD DIFFERENT DATA
                                        ;;*****************************************************************
003366  000004                  TST16:  SCOPE
003370  012777  001111  176050          MOV     #1111,@DAC0             ;LOAD DAC #0
003376  012777  002222  176044          MOV     #2222,@DAC1             ;LOAD DAC #1
003404  012777  004444  176040          MOV     #4444,@DAC2             ;LOAD DAC #2
003412  012777  007777  176034          MOV     #7777,@DAC3             ;LOAD DAC #3
003420  005077  176020                  CLR     @STAT                   ;CLEAR STATUS
003424  012737  001111  001124          MOV     #1111,$GDDAT            ;LOAD EXPECTED
003432  017737  176010  001126          MOV     @DAC0,$BDDAT            ;READ REG
003440  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003446  001401                          BEQ     1$              ;;BR IF EQUAL
003450  104002                          ERROR   2                       ;ERROR, SELECTED DAC0 IN ERROR

003452  012737  002222  001124  1$:     MOV     #2222,$GDDAT            ;LOAD EXPECTED
003460  017737  175764  001126          MOV     @DAC1,$BDDAT            ;READ REG
003466  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003474  001401                          BEQ     2$              ;;BR IF EQUAL
003476  104003                          ERROR   3                       ;ERROR, SELECTED DAC1 IN ERROR

003500  012737  004444  001124  2$:     MOV     #4444,$GDDAT            ;LOAD EXPECTED
003506  017737  175740  001126          MOV     @DAC2,$BDDAT            ;READ REG
003514  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003522  001401                          BEQ     3$              ;;BR IF SAME
003524  104004                          ERROR   4                       ;ERROR, SELECTED DAC #2 IN ERROR

003526  012737  007777  001124  3$:     MOV     #7777,$GDDAT            ;LOAD EXPECTED
003534  017737  175714  001126          MOV     @DAC3,$BDDAT            ;READ REG
003542  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003550  001401                          BEQ     TST17           ;;BR IF SAME
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,$TIMES      ;;DO 10 ITERATIONS
003564  012737  000200  001124          MOV     #BIT7,$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
```

```
                                ;;*****************************************************************
                                ;*TEST 24       TEST THAT INTEPRUPT ENABLE (BIT 6) CAN BE SET
                                ;;*****************************************************************
004154  000004                  TST24:  SCOPE
004156  012777  000100  175260          MOV     #BIT6,@STAT             ;LOAD DISPLAY STATUS
004164  012737  000300  001124          MOV     #BIT7!BIT6,$GDDAT       ;LOAD EXPECTED
004172  017737  175246  001126          MOV     @STAT,$BDDAT            ;READ REG
004200  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004206  001401                          BEQ     TST25           ;;BR IF EQUAL
004210  104001                          ERROR   1                       ;ERROR, STATUS NOT = 300

                                ;;*****************************************************************
                                ;*TEST 25       TEST THAT CHANNEL (BIT 9) CAN BE SET
                                ;;*****************************************************************
004212  000004                  TST25:  SCOPE
004214  012777  001000  175222          MOV     #BIT9,@STAT             ;LOAD DISPLAY STATUS
004222  012737  001200  001124          MOV     #BIT9!BIT7,$GDDAT       ;LOAD EXPECTED
004230  017737  175210  001126          MOV     @STAT,$BDDAT            ;READ REG
004236  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004244  001401                          BEQ     TST26           ;;BR IF EQUAL
004246  104001                          ERROR   1                       ;ERROR, STATUS NOT = 1200

                                ;;*****************************************************************
                                ;*TEST 26       TEST THAT STORE (BIT 10) CAN BE SET
                                ;;*****************************************************************
004250  000004                  TST26:  SCOPE
004252  012777  002000  175164          MOV     #BIT10,@STAT            ;LOAD DISPLAY STATUS
004260  012737  002200  001124          MOV     #BIT10!BIT7,$GDDAT      ;LOAD EXPECTED
004266  017737  175152  001126          MOV     @STAT,$BDDAT            ;READ REG
004274  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004302  001401                          BEQ     TST27           ;;BR IF EQUAL
004304  104001                          ERROR   1                       ;ERROR, STATUS NOT = 2200

                                ;;*****************************************************************
                                ;*TEST 27       TEST THAT WRITE THRU (BIT 11) CAN BE SET
                                ;;*****************************************************************
004306  000004                  TST27:  SCOPE
004310  012777  004000  175126          MOV     #BIT11,@STAT            ;LOAD DISPLAY STATUS
004316  012737  004200  001124          MOV     #BIT11!BIT7,$GDDAT      ;LOAD EXPECTED
004324  017737  175114  001126          MOV     @STAT,$BDDAT            ;READ REG
004332  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004340  001401                          BEQ     TST30           ;;BR IF EQUAL
004342  104001                          ERROR   1                       ;ERROR, STATUS NOT = 4200
```

```
                                        ;;*****************************************************************
                                        ;*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,$GDDAT  ;LOAD EXPECTED RESULTS
004424  013737  001444  010520          MOV     STAT,CHRCOL                     ;MAKE THE HIGH BYTE ADDRESS
004432  005237  010520                  INC     CHRCOL                          ;MAKE IT ODD
004436  105077  004056                  CLRB    @CHRCOL                         ;CLEAR THE HIGH BYTE
004442  017737  174776  001126          MOV     @STAT,$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
                                ;;*****************************************************************
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
                                ;;*****************************************************************
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,$GDDAT       ;LOAD EXPECTED
004670  017737  174650  001126          MOV     @STAT,$BDDAT            ;READ REG
004676  104011                        ERROR   11                      ;ERROR, READY FAILED TO SET
                                                                        ; AFTER MODE 1 OPERATION
```

```
592                                     ;;*****************************************************************
(3)                                     ;*TEST 34       TEST THAT MODE 2 (INTENSIFY ON DAC1) CLEARS THE READY FLAG
(3)                                     ;;*****************************************************************
(2)  004700  000004                     TST34:  SCOPE
593                                     ; AND THEN SETS IT
594  004702  012700  001000                        MOV     #1000,R0                ;SET UP DELAY
595  004706  012737  000210  001124                MOV     #BIT7!BIT3,$GDDAT       ;LOAD EXPECTED
596  004714  012777  000010  174522                MOV     #BIT3,@STAT             ;LOAD MODE 2
597  004722  017737  174516  001126                MOV     @STAT,$BDDAT            ;READ REG
598  004730  105737  001125                       TSTB    $BDDAT                  ;TEST READY
599  004734  100402                                BMI     2$              ;;BR IF SET
600  004736  104011                                ERROR   11                      ;ERROR, IN MODE 2 READY SHOULD NOT CLEAR
601                                                                             ;UNTIL DAC1 IS LOADED
602  004740  000431                                BR      TST35           ;;BR TO SCOPE
603  004742  012737  000010  001124  2$:           MOV     #BIT3,$GDDAT            ;LOAD EXPECTED
604  004750  005077  174474                       CLR     @DAC1                   ;ADDRESS DAC1
605  004754  017737  174464  001126                MOV     @STAT,$BDDAT            ;READ REG
606  004762  105737  001126                       TSTB    $BDDAT                  ;TEST READY
607  004766  100002                                BPL     1$              ;;BR IF CLEARED
608  004770  104011                                ERROR   11                      ;ERROR, MODE 2 LOAD DAC1 FAILED TO CLEAR READY F
609  004772  000414                                BR      TST35           ;;BR TO SCOPE
610  004774  105777  174444           1$:          TSTB    @STAT                   ;TEST READY
611  005000  100411                                BMI     TST35           ;;NEXT TEST
612  005002  005300                                DEC     R0                      ;DELAY
613  005004  001373                                BNE     1$              ;;TEST READY AGAIN
614  005006  012737  000210  001124                MOV     #BIT7!BIT3,$GDDAT       ;LOAD EXPECTED
615  005014  017737  174424  001126                MOV     @STAT,$BDDAT            ;
616  005022  104011                                ERROR   11                      ;ERROR, READY FAILED TO SET
617                                                                             ; AFTER MODE 2 OPERATION
618                                     ;;*****************************************************************
(3)                                     ;*TEST 35       TEST WHEN ERASE IS SET, READY BIT CLEARS AND SET AFTER DELAY    (SW12=1)
(3)                                     ;;*****************************************************************
(2)  005024  000004                     TST35:  SCOPE
(1)  005026  012737  000001  001166            MOV     #1,$TIMES       ;;DO 1 ITERATION
619  005034  032777  010000  174076            BIT     #BIT12,@SWR             ;TEST BIT 12
620  005042  001430                            BEQ     TST36           ;;BYPASS IF NO STORAGE SCOPE
621  005044  012700  000010                    MOV     #10,R0
622  005050  005037  020416                    CLR     TEMP                    ;CLEAR DELAY
623  005054  012777  002000  174362            MOV     #BIT10,@STAT            ;SET STORE MODE
624  005062  052777  010000  174354            BIS     #BIT12,@STAT            ;SET ERASE BIT
625  005070  105777  174350                    TSTB    @STAT                   ;TEST THAT READY CLEARS
626  005074  100002                            BPL     1$              ;;BR IF CLEARED
627  005076  104011                            ERROR   11                      ;ERROR, READY FAILED TO RESET
628  005100  000411                            BR      TST36           ;;BR TO SCOPE
629  005102  105777  174336           1$:      TSTB    @STAT                   ;TEST FOR READY
630  005106  100406                            BMI     TST36           ;;BR IF SET
631  005110  005337  020416                    DEC     TEMP                    ;DELAY
632  005114  001372                            BNE     1$              ;;BR IF NOT READY
633  005116  005300                            DEC     R0                      ;DECREMENT COUNTER
634  005120  001370                            BNE     1$              ;;BR IF NOT DONE
635  005122  104011                            ERROR   11                      ;ERROR, ERASE CLEARED READY AND FAILED
636                                                                             ;TO SET READY AFTER A DELAY
637                                                             ;SWR 12 = 1 AND NO STORAGE SCOPE CONNECTED **
```

```
                                        ;;*****************************************************************
                                        ;*TEST 36       TEST THAT THE DISPLAY DOES INTERRUPT AT LEVEL INDICATED -1
                                        ;;*****************************************************************
005124  000004                  TST36:  SCOPE
005126  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
005134  012737  000340  177776          MOV     #340,PSW
005142  012777  005204  174306          MOV     #1$,@IV         ;SET INTERRUPT VECTOR
005150  012700  000400                  MOV     #400,R0                 ;SET UP DELAY
005154  013737  020424  177776          MOV     BRLEV2,PSW              ;LOAD CPU PSW WITH DEVICE LEVEL -1
005162  012777  000101  174254          MOV     #BIT6!BIT0,@STAT        ;START DISPLAY
005170  005300                          DEC     R0                      ;DELAY
005172  001376                          BNE     .-2
005174  005077  174244                  CLR     @STAT                   ;DO NOT LET INTERRUPT ENABLE SET
005200  104006                          ERROR   6                       ;ERROR, FAILED TO INTERRUPT
005202  000401                          BR      2$              ;;NEXT TEST
005204  022626                  1$:     CMP     (SP)+,(SP)+             ;POP THE STACK
005206  013777  001460  174242  2$:     MOV     IVS,@IV         ;RESET VECTOR
005214  005077  174240                  CLR     @IVS

                                        ;;*****************************************************************
                                        ;*TEST 37       TEST THAT THE DISPLAY DOES NOT INTERRUPT AT LEVEL INDICATED
                                        ;;*****************************************************************
005220  000004                  TST37:  SCOPE
005222  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
005230  012737  000340  177776          MOV     #340,PSW
005236  012777  005272  174212          MOV     #1$,@IV         ;SET INTERRUPT VECTOR
005244  012700  004000                  MOV     #4000,R0                ;SET UP DELAY
005250  013737  020422  177776          MOV     BRLEV1,PSW              ;LOAD CPU PSW WITH DEVICE LEVEL
005256  012777  000101  174160          MOV     #BIT6!BIT0,@STAT        ;START DISPLAY
005264  005300                          DEC     R0                      ;DELAY
005266  001376                          BNE     .-2
005270  000404                          BR      2$              ;;
005272  005077  174146          1$:     CLR     @STAT                   ;DO NOT LET INTERRUPT ENABLE SET
005276  104007                          ERROR   7                       ;ERROR INTERRUPTED IN ERROR
005300  000417                          BR      TST40           ;;NEXT TEST
005302  012777  005324  174146  2$:     MOV     #3$,@IV         ;LOAD RETURN VECTOR
005310  005037  177776                  CLR     PSW                     ;LOWER PSW
005314  005077  174124                  CLR     @STAT                   ;CLEAR INT ENABLE
005320  104007                          ERROR   7                       ;LOWERING THE PRIORITY FAILED TO ALLOW INTERRUPT
005322  000401                          BR      4$              ;;NEXT TEST
005324  022626                  3$:     CMP     (SP)+,(SP)+
005326  013777  001460  174122  4$:     MOV     IVS,@IV         ;RESET VECTOR
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 DAC0 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,@DAC0
005502  005037  001124                CLR     $GDDAT                  ;LOAD EXPECTED
005506  000005                        RESET
005510  017737  173732  001126          MOV     @DAC0,$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 0 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)
                                        ;;*****************************************************************
                                        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)
                                        ;;*****************************************************************
                                        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 DACO TO 4000        (SW09=1)
                                        ;;*****************************************************************
                                        TST46:  SCOPE
005722  012737  000040  001166                MOV     #40,$TIMES      ;;DO 40 ITERATIONS
005730  032777  001000  173202                BIT     #SW09,@SWR              ;TEST BIT 9
005736  001417                                BEQ     TST47           ;;BR IF CLEARED
005740  012777  003777  173500                MOV     #3777,@DACO             ;LOAD DACO
005746  012737  004000  001124                MOV     #4000,$GDDAT            ;LOAD EXPTECTED
005754  000005                                RESET
005756  017737  173464  001126                MOV     @DACO,$BDDAT            ;READ THE DACO REGISTER
005764  023737  001124  001126                CMP     $GDDAT,$BDDAT           ;COMPARE
005772  001401                                BEQ     TST47           ;;BR IF SAME
005774  104002                                ERROR   2                       ;RESET FAILED TO SET DACO TO 4000
                                        ;;*****************************************************************
                                        ;*TEST 47        TEST THAT RESET SET DAC1 TO 4000        (SW09=1)
                                        ;;*****************************************************************
                                        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
```

```
740                                     ;;*****************************************************************
(4)                                     ;*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,$BDDAT            ;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  001166             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  020426             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             ADD     VVECT,IVS
787  006470  005037  001102                     CLR     $TSTNM
788  006474  000137  002466                     JMP     LTEST                   ;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     $PASS           ;;INCREMENT THE PASS NUMBER
006516  042737  100000  001202          BIC     #100000,$PASS   ;;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                  $ENDCT: .WORD   1
006536  006526                          $EOPCT
006540  104401  006605                   TYPE    ,$ENDMG         ;;TYPE "END PASS #"
006544  013746  001202                   MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
006550  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
006552  104401  006602                   TYPE    ,$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
```

```
 805  006622  012706  001100          VISJAL: MOV     #STACK,SP               ;LOAD SP
 806  006626  005737  020430                  TST     EVER1                   ;TEST IF ON AUTO-HARDWARE TESTER
 807  006632  100003                         BPL     1$                      ;BR IF NOT
 808  006634  004537  014062                 JSR     R5,CROSS                ;CHANGE THE RELAYS
 809  006640  120000                         BIT15!BIT13
 810  006642  105737  020430          1$:     TSTB    EVER1                   ;TEST IF TYPED ONCE
 811  006646  001006                         BNE     PIC0                    ;BR IF YES
 812  006650  104401                         TYPE
 813  006652  015403                         MES6                    ;HEADER ABOUT SCOPE ADJ.
 814  006654  104401                         TYPE
 815  006656  015433                         MES15                   ;TEXT ABOUT SWR
 816  006660  105237  020430                 INCB    EVER1                   ;SET FLAG
 820  006664  013700  001446          PIC0:   MOV     DAC0,R0
 821  006670  013701  001450                 MOV     DAC1,R1
 822  006674  032777  000010  172236         BIT     #SW03,@SWR              ;TEST SR BIT 3
 823  006702  001404                         BEQ     2$                      ;BR IF DAC #0 AND #1
 824  006704  013700  001452                 MOV     DAC2,R0         ;USE DAC #2 AND 3
 825  006710  013701  001454                 MOV     DAC3,R1
 826  006714  012737  000030  020414  2$:     MOV     #30,TICKS               ;LOAD TIMER FOR CP TYPE
 827  006722  004737  014240                 JSR     PC,CHTIME               ;CHANGE TIMER FOR CP TYPE
 828  006726  004737  007016          1$:     JSR     PC,PBB
 829  006732  004737  014124                 JSR     PC,TIMER                ;DONE ?
 930  006736  000773                         BR      1$
 834  006740  013700  001450          PIC1:   MOV     DAC1,R0
 835  006744  013701  001446                 MOV     DAC0,R1
 836  006750  032777  000010  172162         BIT     #SW03,@SWR              ;TEST SR BIT 3
 837  006756  001404                         BEQ     2$                      ;BR IF DAC #0 AND 1
 838  006760  013700  001454                 MOV     DAC3,R0         ;USE DAC #2 AND 3
 839  006764  013701  001452                 MOV     DAC2,R1
 840  006770  012737  000030  020414  2$:     MOV     #30,TICKS               ;LOAD TIME
 841  006776  004737  014240                 JSR     PC,CHTIME               ;CHANGE TIMER FOR CP TYPE
 842  007002  004737  007016          1$:     JSR     PC,PBB
 843  007006  004737  014124                 JSR     PC,TIMER                ;DONE ?
 844  007012  000773                         BR      1$
 845  007014  000440                         BR      PIC3
 846  007016  005077  172422          PBB:    CLR     @STAT
 847  007022  032777  002000  172110         BIT     #SW10,@SWR              ;TEST IS SW10 =1
 848  007030  001403                         BEQ     10$                     ;BR IF NOT
 849  007032  052777  000020  172404         BIS     #BIT4,@STAT             ;ENABLE EXT. DELAY
 850  007040  013704  001444          10$:    MOV     STAT,R4
 851  007044  012703  007777                 MOV     #7777,R3        ;SET HIGH LIMIT
 852  007050  012702  000001                 MOV     #1,R2           ;INITIALIZE INCREMENTS BETWEEN POINTS
 853  007054  032777  001000  172356         BIT     #SW09,@SWR              ;TEST SW 9 = 1
 854  007062  001012                         BNE     2$                      ;BR IF YES
 855  007064  012711  004000                 MOV     #4000,(R1)
 856  007070  005010                         CLR     (R0)
 857  007072  060210          1$:     ADD     R2,(R0)         ;INCREMENT
 858  007074  005214                 INC     (R4)            ;INTENSIFY
 859  007076  105714                 TSTB    (R4)
 860  007100  100376                 BPL     .-2
 861  007102  021003                 CMP     (R0),R3         ;DONE ALL POINTS?
 862  007104  001372                 BNE     1$              ;NO
 863  007106  000207                 RTS     PC
 864  007110  005011          2$:     CLR     (R1)                    ;LOAD TWO'S COMP. ORIGIN
```

```
   865  007112  005010                    CLR     (R0)
   866  007114  000766                    BR      1$
```

```
                                        .SBTTL  PINCUSHION TEST (DISPLAY SQUARE)

007116  005077  172322        PIC3:   CLR     @STAT
007122  012737  000140  020414        MOV     #140,TICKS
007130  004737  014240                JSR     PC,CHTIME
007134  013701  001446                MOV     DAC0,R1
007140  013702  001450                MOV     DAC1,R2
007144  032777  000010  171766        BIT     #SW03,@SWR      ;TEST SR BIT 3
007152  001404                        BEQ     1$              ;BR IF DAC #0 AND 1
007154  013701  001452                MOV     DAC2,R1         ;USE DAC #2 AND 3
007160  013702  001454                MOV     DAC3,R2
007164  013703  001444        1$:     MOV     STAT,R3
007170  012704  000020                MOV     #20,R4
007174  032777  001000  171736  P3:    BIT     #SW09,@SWR      ;TEST BIT 9
007202  001405                        BEQ     1$
007204  012711  004000                MOV     #4000,(R1)      ;LOAD TWO'S COMP ORIGIN
007210  012712  004000                MOV     #4000,(R2)
007214  000402                        BR      2$
007216  005011        1$:     CLR     (R1)            ;CLEAR AXIS
007220  005012                CLR     (R2)
                        ;DRAW BOTTOM LINE
007222  012700  000377        2$:     MOV     #377,R0
007226  032777  002000  171704        BIT     #SW10,@SWR              ;TEST IF SW10 =1
007234  001403                        BEQ     P3A                     ;BR IF NOT
007236  052777  000020  172200        BIS     #BIT4,@STAT             ;ENABLE EXT. DELAY
007244  060411        P3A:    ADD     R4,(1)
007246  005213                INC     (R3)            ;INTENSIFY
007250  105713                TSTB    (3)
007252  100376                BPL     .-2             ;WAIT FOR READY
007254  005300                DEC     R0
007256  001372                BNE     P3A             ;NO
                        ;DRAW RIGHT LINE
007260  012700  000377        MOV     #377,R0
007264  060412        P3B:    ADD     R4,(2)
007266  005213                INC     (R3)            ;INTENSIFY
007270  105713                TSTB    (3)
007272  100376                BPL     .-2             ;WAIT FOR READY
007274  005300                DEC     R0
007276  001372                BNE     P3B             ;NO
                        ;DRAW TOP LINE
007300  012700  000377        MOV     #377,R0
007304  160411        P3C:    SUB     R4,(1)
007306  005213                INC     (R3)            ;INTENSIFY
007310  105713                TSTB    (3)
007312  100376                BPL     .-2             ;WAIT FOR READY
007314  005300                DEC     R0
007316  001372                BNE     P3C             ;NO
                        ;DRAW LEFT LINE
007320  012700  000377        MOV     #377,R0
007324  160412        P3D:    SUB     R4,(2)
007326  005213                INC     (R3)            ;INTENSIFY
007330  105713                TSTB    (3)
007332  100376                BPL     .-2             ;WAIT FOR READY
007334  005300                DEC     R0
```

```
                                        BNE     P3D             ;NO
                                        JSR     PC,TIMER
                                        BR      P3
                                        .SBTTL  PLOT AN X

007346  005077  172072        PIC4:     CLR     @STAT
007352  012737  000100  020414          MOV     #100,TICKS
007360  004737  014240                  JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
007364  013701  001446        PIC4B:    MOV     DAC0,R1
007370  013702  001450                  MOV     DAC1,R2
007374  032777  000010  171536          BIT     #SW03,@SWR      ;TEST SWR BIT 3
007402  001404                          BEQ     1$              ;BR IF DAC #0 AND 1
007404  013701  001452                  MOV     DAC2,R1         ;USE DAC #2 AND 3
007410  013702  001454                  MOV     DAC3,R2
007414  013703  001444        1$:       MOV     STAT,R3
007420  012704  000004                  MOV     #4,R4
007424  032777  001000  171506  P4:     BIT     #SW09,@SWR      ;TEST BIT 9
007432  001405                          BEQ     1$
007434  012712  004000                  MOV     #4000,(R2)
007440  012711  004003                  MOV     #4000,(R1)
007444  000402                          BR      2$
007446  005012        1$:               CLR     (R2)
007450  005011                          CLR     (R1)

                                ;PLOT LINE BEGINNING IN LOWER LEFT CORNER
007452  012700  001777        2$:       MOV     #1777,R0
007456  032777  002000  171454          BIT     #SW10,@SWR              ;TEST IF SW10 =1
007464  001403                          BEQ     P4A                     ;BR IF NOT
007466  052777  000020  171750          BIS     #BIT4,@STAT             ;ENABLE EXT. DELAY
007474  005213        P4A:              INC     (R3)            ;INTENSIFY
007476  105713                          TSTB    (3)
007500  100376                          BPL     .-2
007502  060412                          ADD     R4,(2)          ;+4 TO Y
007504  060411                          ADD     R4,(1)          ;+4 TO X
007506  005300                          DEC     R0
007510  001371                          BNE     P4A             ;NO
007512  105713                          TSTB    (3)
007514  100376                          BPL     .-2

                                ;PLOT LINE BEGINNING IN UPPER LEFT CORNER
007516  012712  007774                  MOV     #7774,(2)
007522  005011                          CLR     (1)
007524  012700  001777                  MOV     #1777,R0
007530  005213        P4B:              INC     (R3)            ;INTENSIFY
007532  105713                          TSTB    (3)
007534  100376                          BPL     .-2
007536  160412                          SUB     R4,(2)          ;-4 TO Y
007540  060411                          ADD     R4,(1)          ;+4 TO X
007542  005300                          DEC     R0
007544  001371                          BNE     P4B             ;NO
007546  004737  014124                  JSR     PC,TIMER
007552  000724                          BR      P4
```

```
                                                .SBTTL  SCOPE SETTLING TIME TEST
                
007554  012737  000060  020414  PIC5:   MOV     #60,TICKS       ;LOAD TIMER
007562  004737  014240                  JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
                
007566  005077  171652          1$:     CLR     @STAT           ;CLEAR STATUS
007572  005077  171650                  CLR     @DAC0           ;CLEAR X AXIS
007576  005077  171646                  CLR     @DAC1           ;CLEAR Y AXIS
007602  005077  171644                  CLR     @DAC2
007606  005077  171642                  CLR     @DAC3
007612  032777  002000  171320          BIT     #SW10,@SWR      ;TEST SW BIT 10
007620  001403                          BEQ     10$             ;BR IF NOT
007622  052777  000020  171614          BIS     #BIT4,@STAT     ;ENABLE EXT. DELAY
007630  032777  000010  171302  10$:    BIT     #SW03,@SWR      ;TEST SR BIT 3
007636  001405                          BEQ     2$              ;BR IF DAC #0 AND 1
007640  013700  001454                  MOV     DAC3,R0         ;LOAD Y DAC
007644  013701  001452                  MOV     DAC2,R1         ;LOAD X DAC
007650  000404                          BR      3$
007652  013700  001450          2$:     MOV     DAC1,R0         ;LOAD Y DAC
007656  013701  001446                  MOV     DAC0,R1         ;LOAD X DAC
007662  032777  000200  171250  3$:     BIT     #SW07,@SWR      ;TEST X OR Y TEST BIT
007670  001403                          BEQ     4$              ;NORMAL
007672  010003                          MOV     R0,R3           ;REVERSE THE AXIS
007674  010100                          MOV     R1,R0
007676  010301                          MOV     R3,R1
007700  013703  001444          4$:     MOV     STAT,R3         ;LOAD STATUS REG
007704  032777  001000  171226  5$:     BIT     #SW09,@SWR      ;TEST TWO'S COMP SWITCH
007712  001007                          BNE     6$
007714  004537  007762                  JSR     R5,LODPNT       ;LOAD A LINE
007720  000000                          0
007722  004537  007762                  JSR     R5,LODPNT       ;LOAD A LINE
007726  007777                          7777
007730  000406                          BR      7$
007732  004537  007762          6$:     JSR     R5,LODPNT       ;LOAD A LINE
007736  004000                          4000
007740  004537  007762                  JSR     R5,LODPNT
007744  003777                          3777
007746  005710          7$:     TST     (R0)            ;END
007750  001355                          BNE     5$              ;BR IF NOT
007752  004737  014124                  JSR     PC,TIMER        ;TEST TIME
007756  000703                          BR      1$
007760  000414                          BR      PIC6
                
007762  012702  000200  LODPNT: MOV     #200,R2
007766  011511          1$:     MOV     (R5),(R1)       ;LOAD AXIS
007770  005213                  INC     (R3)            ;INTENSIFY
007772  105713          2$:     TSTB    (R3)            ;DONE
007774  100376                  BPL     2$              ;WAIT
007776  062710  000002          ADD     #2,(R0)         ;UPDATE AXIS
010002  005302                  DEC     R2              ;DONE
010004  001370                  BNE     1$              ;BR IF NOT
010006  005725                  TST     (R5)+           ;UPDATE
010010  000205                  RTS     R5              ;EXIT
```

```
 1040                                                   .SBTTL  PLOT CHARACTER SET
 1041
 1042  010012  012737  000100  020414  PIC6:   MOV     #100,TICKS
 1043  010020  004737  014240                  JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
 1044  010024  013737  001446  010510          MOV     DAC0,VCXTMP     ;LOAD THE X AXIS
  (1)  010032  013737  001450  010512          MOV     DAC1,VCYTMP     ;LOAD THE Y AXIS
  (1)  010040  032777  000010  171072          BIT     #SW03,@SWR      ;TEST SR BIT 3
  (1)  010046  001406                          BEQ     1$              ;BR IF DAC #0 AND 1
  (1)  010050  013737  001452  010510          MOV     DAC2,VCXTMP     ;LOAD THE  X AXIS
  (1)  010056  013737  001454  010512          MOV     DAC3,VCYTMP
  (1)  010064  000240                  1$:     NOP
 1045  010066  012737  001400  010516  PIC6A:  MOV     #1400,XPOS
 1046  010074  012737  005400  010514          MOV     #5400,YPOS
 1047  010102  032777  001000  171030          BIT     #SW09,@SWR      ;TEST SW 9
 1048  010110  001406                          BEQ     1$
 1049  010112  012737  005400  010516          MOV     #5400,XPOS      ;LOAD 2'S COMP ORIGIN
 1050  010120  012737  001400  010514          MOV     #1400,YPOS
 1051  010126  012702  010524          1$:     MOV     #A,R2           ;LOAD STARTING CHARACTER
 1052  010132  005077  171306                  CLR     @STAT
 1053  010136  004737  010304                  JSR     PC,PIC6B
 1054  010142  012737  001400  010516          MOV     #1400,XPOS      ;LOAD X POS
 1055  010150  012737  004000  010514          MOV     #4000,YPOS      ;LOAD Y POS
 1056  010156  032777  001000  170754          BIT     #SW09,@SWR      ;TEST SW 9
 1057  010164  001406                          BEQ     2$              ;BR IF 0
 1058  010166  012737  005400  010516          MOV     #5400,XPOS      ;LOAD 2'S COMP ORIGIN
 1059  010174  012737  000000  010514          MOV     #0,YPOS
 1060  010202  012702  010625          2$:     MOV     #N,R2           ;LOAD STARTING CHARACTER
 1061  010206  005077  171232                  CLR     @STAT
 1062  010212  004737  010304                  JSR     PC,PIC6B        ;DISPLAY CHARACTERS
 1063  010216  012737  001400  010516          MOV     #1400,XPOS      ;LOAD X POS
 1064  010224  012737  002400  010514          MOV     #2400,YPOS      ;LOAD Y POS
 1065  010232  032777  001000  170700          BIT     #SW09,@SWR      ;TEST SW 9
 1066  010240  001406                          BEQ     3$
 1067  010242  012737  005400  010516          MOV     #5400,XPOS
 1068  010250  012737  006400  010514          MOV     #6400,YPOS      ;LOAD 2'S COMP ORIGIN
 1069  010256  012702  010726          3$:     MOV     #NO,R2          ;LOAD STARTING CHARACTER
 1070  010262  005077  171156                  CLR     @STAT
 1071  010266  004737  010304                  JSR     PC,PIC6B        ;DISPLAY CHARACTERS
 1072  010272  004737  014124                  JSR     PC,TIMER
 1073  010276  000673                          BR      PIC6A
 1074  010300  000137  011030                  JMP     PIC7
 1075  010304  012737  177763  010520  PIC6B:  MOV     #-13.,CHRCOL    ;CHARACTERS PER ROW
 1076  010312  013705  001444                  MOV     STAT,R5
 1077  010316  004737  010332          GEN1:   JSR     PC,CHAR
 1078  010322  005237  010520                  INC     CHRCOL          ;
 1079  010326  001373                          BNE     GEN1
 1080  010330  000207                          RTS     PC
 1081
```

```
1083
1084                                        ;PLOT CHARACTER
1085  010332  013737  010514  010522  CHAR:  MOV     YPOS,YPT
1086  010340  042715  000C16                BIC     #16,(5)
1087  010344  013777  010516  000136        MOV     XPOS,@VCXTMP
1088  010352  013777  010514  000132        MOV     YPOS,@VCYTMP
1089  010360  032777  002000  170552        BIT     #SW10,@SWR      ;TEST SR BIT 10
1090  010366  001403                        BEQ     CHAR4           ;BR IF NOT
1091  010370  052777  000020  171046        BIS     #BIT4,@STAT     ;ENABLE EXT. DELAY
1092  010376  105777  171042          CHAR4: TSTB    @STAT
1093  010402  100375                        BPL     CHAR4
1094  010404  052715  000002                BIS     #BIT1,(5)       ;ENABLE FAST INTENSIFY ON LOADING Y
1095  010410  012704  000040                MOV     #40,R4
1096  010414  012700  177773                MOV     #-5,R0          ;INITIALIZE COLUMN COUNT
1097  010420  012701  177771          CHAR1: MOV     #-7,R1          ;INITIALIZE ROW COUNT
1098  010424  112203                        MOVB    (2)+,R3         ;PUT CHARACTER POINTS IN R3
1099  010426  106103          CHAR2:        ROLB    R3
1100  010430  100007                        BPL     CHAR3           ;NO
1101  010432  013777  010516  000050        MOV     XPOS,@VCXTMP
1102  010440  013777  010514  000044        MOV     YPOS,@VCYTMP
1103  010446  005215                        INC     (R5)            ;INTENSIFY
1104  010450  105715          CHAR3:        TSTB    (R5)
1105  010452  100376                        BPL     CHAR3
1106  010454  060437  010514                ADD     R4,YPOS
1107  010460  005201                        INC     R1              ;+1 TO ROW
1108  010462  001361                        BNE     CHAR2           ;FINISH ROW
1109  010464  013737  010522  010514        MOV     YPT,YPOS        ;REINITIALIZE ROW FOR NEXT COLUMN
1110  010472  060437  010516                ADD     R4,XPOS
1111  010476  005200                        INC     R0              ;+1 TO COLUMN COUNT
1112  010500  001347                        BNE     CHAR1
1113  010502  060437  010516                ADD     R4,XPOS
1114  010506  000207                        RTS     PC              ;EXIT
1115
1116  010510  000000          VCXTMP: 0
1117  010512  000000          VCYTMP: 0
1118  010514  000000          YPOS:   0               ;CONTAINS Y POSITION AT ANY GIVEN TIME
1119  010516  000000          XPOS:   0               ;CONTAINS X POSITION AT ANY GIVEN TIME
1120  010520  000000          CHRCOL: 0
1121  010522  000000          YPT:    0
1122
```

```
1124
1125  010524  176  021  021  A:   .BYTE 176,21,21,21,176
      010527  021  176
1126  010531  177  111  111  B:   .BYTE 177,111,111,111,66
      010534  111  066
1127  010536  076  101  101  C:   .BYTE 76,101,101,101,42
      010541  101  042
1128  010543  177  101  101  D:   .BYTE 177,101,101,101,76
      010546  101  076
1129  010550  177  111  111  E:   .BYTE 177,111,111,111,101
      010553  111  101
1130  010555  177  011  011  F:   .BYTE 177,11,11,11,1
      010560  011  001
1131  010562  076  101  121  G:   .BYTE 76,101,121,121,62
      010565  121  062
1132  010567  177  010  010  H:   .BYTE 177,10,10,10,177
      010572  010  177
1133  010574  000  101  177  I:   .BYTE 0,101,177,101,0
      010577  101  000
1134  010601  060  100  100  J:   .BYTE 60,100,100,100,77
      010604  100  077
1135  010606  177  010  024  K:   .BYTE 177,10,24,42,101
      010611  042  101
1136  010613  177  100  100  L:   .BYTE 177,100,100,100,100
      010616  100  100
1137  010620  177  004  010  M:   .BYTE 177,4,10,4,177
      010623  004  177
1138  010625  177  004  010  N:   .BYTE 177,4,10,20,177
      010630  020  177
1139  010632  076  101  101  O:   .BYTE 76,101,101,101,76
      010635  101  076
1140  010637  177  011  011  P:   .BYTE 177,11,11,11,6
      010642  011  006
1141  010644  076  101  121  Q:   .BYTE 76,101,121,141,176
      010647  141  176
1142  010651  177  011  031  R:   .BYTE 177,11,31,51,106
      010654  051  106
1143  010656  046  111  111  S:   .BYTE 46,111,111,111,62
      010661  111  062
1144  010663  001  001  177  T:   .BYTE 1,1,177,1,1
      010666  001  001
1145  010670  077  100  100  U:   .BYTE 77,100,100,100,77
      010673  100  077
1146  010675  037  040  100  V:   .BYTE 37,40,100,40,37
      010700  040  037
1147  010702  177  020  010  W:   .BYTE 177,20,10,20,177
      010705  020  177
1148  010707  143  024  010  X:   .BYTE 143,24,10,24,143
      010712  024  143
1149  010714  003  004  170  Y:   .BYTE 3,4,170,4,3
      010717  004  003
1150  010721  141  121  111  Z:   .BYTE 141,121,111,105,103
      010724  105  103
1151  010726  076  121  111  NO:  .BYTE  76,121,111,105,76
```

```
                010731  105  076
1152            010733  000  102     177  N1:    .BYTE 0,102,177,100,0
                010736  100  000
1153            010740  142  121     111  N2:    .BYTE 142,121,111,105,102
                010743  105  102
1154            010745  042  101     111  N3:    .BYTE 42,101,111,111,66
                010750  111  066
1155            010752  030  024     022  N4:    .BYTE 30,24,22,177,20
                010755  177  020
1156            010757  047  105     105  N5:    .BYTE 47,105,105,105,71
                010762  105  071
1157            010764  076  111     111  N6:    .BYTE 76,111,111,111,62
                010767  111  062
1158            010771  101  041     021  N7:    .BYTE 101,41,21,11,7
                010774  011  007
1159            010776  066  111     111  N8:    .BYTE 66,111,111,111,66
                011001  111  066
1160            011003  046  111     111  N9:    .BYTE 46,111,111,111,76
                011006  111  076
1161            011010  000  000     000  SPACEA: .BYTE 0,0,0,0,0
                011013  000  000
1162            011015  000  000     000          .BYTE   0,0,0,0,0
                011020  000  000
1163            011022  000  000     000          .BYTE   0,0,0,0,0
                011025  000  000
1164
1165                    011030                   .EVEN
1166                                             .SBTTL  CHANNEL 1 AND CHANNEL 2 TEST
```

```
1168  011030  012737  000200  020414  PIC7:   MOV     #200,TICKS      ;SET UP A TIMER
1169  011036  004737  014240                  JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
1170  011042  013737  001446  010510          MOV     DAC0,VCXTMP     ;LOAD THE X AXIS
(1)   011050  013737  001450  010512          MOV     DAC1,VCYTMP     ;LOAD THE Y AXIS
(1)   011056  032777  000010  170054          BIT     #SW03,@SWR      ;TEST SR BIT 3
(1)   011064  001406                          BEQ     1$              ;BR IF DAC #0 AND 1
(1)   011066  013737  001452  010510          MOV     DAC2,VCXTMP     ;LOAD THE  X AXIS
(1)   011074  013737  001454  010512          MOV     DAC3,VCYTMP
(1)   011102  000240                  1$:     NOP
1171  011104  013705  001444                  MOV     STAT,R5         ;SET UP R5 FOR STATUS REGISTER POINTER
1172  011110  012777  000000  170326  PIC7AA: MOV     #0,@STAT        ;SET UP SCOPE CONTROL
1173  011116  012737  002000  010516          MOV     #2000,XPOS      ;LOAD X POSITION
1174  011124  012737  005000  010514          MOV     #5000,YPOS      ;LOAD Y POSITION
1175  011132  032777  001000  170000          BIT     #SW09,@SWR      ;TEST BIT 9
1176  011140  001406                          BEQ     1$
1177  011142  012737  006000  010516          MOV     #6000,XPOS      ;LOAD 2'S COMP ORIGIN
1178  011150  012737  001000  010514          MOV     #1000,YPOS
1179  011156  012737  000011  020434  1$:     MOV     #9.,P7CNT       ;SAVE THE NUMBER OF CHARACTERS
1180  011164  012737  020440  020436          MOV     #CH01,P7PNT     ;SAVE CHANNEL 1 POINTER
1181  011172  017702  007240          PIC7A:  MOV     @P7PNT,R2       ;MOVE MESSAGE POINTER INTO R2 FOR DISPLAY ROUTINE
1182  011176  004737  010332                  JSR     PC,CHAR         ;DISPLAY A CHARACTER
1183  011202  062737  000002  020436          ADD     #2,P7PNT        ;ADD 2 TO THE MESSAGE POINTER
1184  011210  005337  020434                  DEC     P7CNT           ;DECREMENT CHARACTER COUNT
1185  011214  001366                          BNE     PIC7A           ;NOT FINISHED WITH ALL CHARACTERS
1186  011216  012777  001000  170220          MOV     #1000,@STAT
1187  011224  012737  002000  010516          MOV     #2000,XPOS      ;SET UP X POS FOR CHANNEL 2
1188  011232  012737  003000  010514          MOV     #3000,YPOS      ;SET UP Y
1189  011240  032777  001000  167672          BIT     #SW09,@SWR      ;TEST BIT 9
1190  011246  001406                          BEQ     1$
1191  011250  012737  006000  010516          MOV     #6000,XPOS      ;LOAD 2'S COMP ORIGIN
1192  011256  012737  007000  010514          MOV     #7000,YPOS
1193  011264  012737  000011  020434  1$:     MOV     #9.,P7CNT       ;SET UP CHARACTER COUNT
1194  011272  012737  020462  020436          MOV     #CH02,P7PNT     ;SET UP CHANNEL 2 POINTER
1195  011300  017702  007132          PIC7B:  MOV     @P7PNT,R2       ;SET UP
1196  011304  004737  010332                  JSR     PC,CHAR         ;DISPLAY A CHARACTER
1197  011310  062737  000002  020436          ADD     #2,P7PNT        ;ADD 2 TO THE POINTER
1198  011316  005337  020434                  DEC     P7CNT           ;DECREMENT COUNT
1199  011322  001366                          BNE     PIC7B           ;NOT FINISHED
1200  011324  004737  014124                  JSR     PC,TIMER        ;CHECK THE RUNTIME OF THIS ROUTINE
1201  011330  000667                          BR      PIC7AA          ;NOT FINISHED
1202  011332  000400                          BR      PIC12
```

```
1204                                                    .SBTTL  PHOSPHOR TEST
1205
1206  011334  012737  000002  020414  PIC12:  MOV     #2,TICKS
1207  011342  004737  013524          1$:     JSR     PC,CLRVC                ;ERASE THE SCREEN
1208  011346  004737  013642                  JSR     PC,LOADVC               ;LOAD THE SCREEN
1209  011352  004737  014124                  JSR     PC,TIMER                ;CHECK THE TIME
1210  011356  000771                          BR      1$
1211  011360  004737  013524                  JSR     PC,CLRVC
1212  011364  000005                          RESET
1213  011366  000137  006664                  JMP     PIC0
1214
1221                                                    .SBTTL  MANUAL DISPLAY ROUTINE
1222
1223  011372  012706  001100          FULRMP: MOV     #STACK,SP               ;LOAD POINTER
1224  011376  004737  001742                  JSR     PC,LDTRAP               ;LOAD BUS ADDRESS
1225  011402  032777  002000  167530          BIT     #SW10,@SWR      ;TEST SR 10=1
1226  011410  001403                          BEQ     1$              ;BR IF NOT
1227  011412  052777  000020  170024          BIS     #BIT4,@STAT     ;ENABLE EXT. DELAY
1228  011420  013700  001446          1$:     MOV     DAC0,R0         ;GET BUS ADDRESS
1229  011424  004737  011462                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #1
1230  011430  013700  001450                  MOV     DAC1,R0         ;GET BUS ADDRESS
1231  011434  004737  011462                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #1
1232  011440  013700  001452                  MOV     DAC2,R0         ;GET BUS ADDRESS
1233  011444  004737  011462                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #2
1234  011450  013700  001454                  MOV     DAC3,R0         ;GET THE BUS ADDRESS
1235  011454  004737  011462                  JSR     PC,10$                  ;LOAD THE RAMP ON DAC #3
1236  011460  000757                          BR      1$                      ;BR BACK
1237
1238  011462  005010          10$:    CLR     (R0)                    ;CLEAR DAC
1239  011464  062710  000010          11$:    ADD     #10,(R0)                ;UPDATE THE DATA
1240  011470  005710                          TST     (R0)                    ;TEST IF DONE
1241  011472  001374                          BNE     11$                     ;BR IF NOT
1242  011474  004737  013246                  JSR     PC,ACTRLC               ;TEST FOR CTRL C
1243  011500  000207                          RTS     PC                      ;EXIT
1244  011502  000240                          NOP
1245  011504  000240                          NOP
1246
```

```
1248                                        .SBTTL  AUTO CALIBRATION
1249  011506  012706  001100          AUTCAL: MOV     #STACK,SP               ;LOAD STACK POINTER
1250  011512  004737  001742                  JSR     PC,LDTRAP               ;LOAD TRAN AND ADDRESSES
1251  011516  052737  100000  020430          BIS     #BIT15,EVER1            ;INDICATE PROG. IS RUNNING ON THE TESTER HARD
1252  011524  104401                          TYPE
1253  011526  017325                          AUTOCL                          ;TELL OPERATOR
1254  011530  104401                          TYPE
1255  011532  017033                          SELD01
1256
1257                                        ;LOAD BUS TRAP FOR OPERTOR FALSE SELECTION
1258
1259  011534  012737  013064  000004          MOV     #AUTRAP,ERRVEC
1260  011542  012737  000054  001102          MOV     #$TN-1,$TSTNM           ;LOAD TEST NUMBER
1261  011550  012737  011574  001110          MOV     #PSUPLY,$LPERR          ;LOAD LOOP RETURN
1262  011556  012737  011574  001106          MOV     #PSUPLY,$LPADR          ;LOAD LOOP ADDRESS
1302                                          ;@@@@@@@@@@@@@@@@@@@@@@@@@@@@
1303  011564  004537  014062                  JSR     R5,CROSS                ;CHANGE RELAYS
1304  011570  020000                          BIT13                           ;USING BIT 13 FOR CRISS-CROSS WRAP MODE
1305                                          ;@@@@@@@@@@@@@@@@@@@@@@@@@@@@
1306                                        ;;**************************************************************
 (3)                                        ;*TEST 55        TEST THAT CH 3 IS AT +2.5 BIPOLAR
 (3)                                        ;;**************************************************************
 (2)  011572  000004                        TST55:  SCOPE
1307  011574  013737  014514  001124  PSUPLY: MOV     V1754,$GDDAT            ;LOAD EXPECTED VALUE
1308  011602  004537  014370                  JSR     R5,CONVRT               ;CONVERT
1309  011605  000003                          CH3                             ;CH 3
1310  011610  013737  014526  001126          MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1311  011616  013737  014516  014534          MOV     V24,SPREAD              ;LOAD + OR - COUNT SPREAD
1312  011624  004737  014536                  JSR     PC,COMPAR               ;COMPARE $GDDAT AND $BDDAT
1313  011630  000402                          BR      NSUPLY          ;;BR IF WITHIN TOLERANCE
1314  011632  104012                          ERROR   12                      ;CH 3 FAILED TO EQUAL EXPECTED VALUE
1315
1316                                        ;;**************************************************************
 (3)                                        ;*TEST 56        TEST THAT CH 4 IS AT -2 5 BIPOLAR
 (3)                                        ;;**************************************************************
 (2)  011634  000004                        TST56:  SCOPE
1317  011636  013737  014516  001124  NSUPLY: MOV     V24,$GDDAT              ;LOAD EXPECTED VALUE
1318  011644  004537  014370                  JSR     R5,CONVRT               ;CONVERT
1319  011650  000004                          CH4                             ;CH 4
1320  011652  013737  014526  001126          MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1321  011660  013737  014516  014534          MOV     V24,SPREAD              ;LOAD + OR - COUNT SPREAD
1322  011666  004737  014536                  JSR     PC,COMPAR               ;COMPARE $GDDAT AND $BDDAT
1323  011672  000401                          BR      TST57           ;;BR IF WITHIN TOLERANCE
1324  011674  104012                          ERROR   12                      ;CH 4 FAILED TO EQUAL EXPECTED VALUE
```

```
1326                                    ;;*****************************************************************
 (3)                                    ;*TEST 57       TEST THAT ERASE L (CH 53) IS GREATER THAN +3 VOLTS WHEN HIGH
 (3)                                    ;;*****************************************************************
 (2)  011676  000004                    TST57:  SCOPE
1327  011700  012777  000000  167536             MOV     #0,@STAT                ;LOAD STATUS
1328  011706  012737  001146  001124             MOV     #1146,$GDDAT            ;LOAD EXPECTED VALUE
1329  011714  004537  014370                     JSR     R5,CONVRT               ;CONVERT
1330  011720  000053                             CH53                            ;CH 53
1331  011722  013737  014526  001126             MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1332  011730  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
1333  011736  003401                             BLE     TST60           ;;BR IF LESS THAN OR EQUAL
1334  011740  104014                             ERROR   14                      ;CH 53 FAILED TO EQUAL EXPECTED VALUE
1335
1336                                    ;;*****************************************************************
 (3)                                    ;*TEST 60       TEST THAT ERASE L (CH 53) IS LESS THAN +400 MVOLTS WHEN LOW
 (3)                                    ;;*****************************************************************
 (2)  011742  000004                    TST60:  SCOPE
1337  011744  005077  167474                     CLR     @STAT                   ;CLEAR STATUS
1338  011750  012777  012000  167466             MOV     #BIT12!BIT10,@STAT      ;SET ERASE
1339  011756  012737  000120  001124             MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
1340  011764  004537  014370                     JSR     R5,CONVRT               ;CONVERT
1341  011770  000053                             CH53                            ;CH 53
1342  011772  013737  014526  001126             MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1343  012000  023737  001124  001126             CMP     $GDDAT,$BDDAT           ;COMPARE
1344  012006  002001                             BGE     TST61           ;;BR IF GREATER THAN OR EQUAL
1345  012010  104015                             ERROR   15                      ;CH 53 FAILED TO EQUAL EXPECTED VALUE
```

```
1348                                    ;;*****************************************************************
 (3)                                    ;*TEST 61        TEST THAT WRITE-THRU (CH 53) IS LESS THAN +400 MVOLTS WHEN LOW
 (3)                                    ;;*****************************************************************
 (2)  012012  000004                    TST61:  SCOPE
1349  012014  012777  006000  167422            MOV     #BIT11!BIT10,@STAT      ;SET WRITE-THRU
1350  012022  012737  000120  001124            MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
1351  012030  004537  014370                    JSR     R5,CONVRT               ;CONVERT
1352  012034  000053                            CH53                            ;CH 54
1353  012036  013737  014526  001126            MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1354  012044  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1355  012052  002001                            BGE     TST62           ;;BR IF GREATER THAN OR EQUAL
1356  012054  104015                            ERROR   15                      ;CH 54 FAILED TO EQUAL EXPECTED VALUE
1357                                    ;;*****************************************************************
 (3)                                    ;*TEST 62        TEST THAT NON-STORE (CH 55) IS LESS THAN +400 MVOLTS WHEN LOW
 (3)                                    ;;*****************************************************************
 (2)  012056  000004                    TST62:  SCOPE
1358  012060  005077  167360                    CLR     @STAT                   ;CLEAR STORE MODE
1359  012064  012737  000120  001124            MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
1360  012072  004537  014370                    JSR     R5,CONVRT               ;CONVERT
1361  012076  000055                            CH55                            ;CH 55
1362  012100  013737  014526  001126            MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1363  012106  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1364  012114  002001                            BGE     TST63           ;;BR IF GREATER THAN OR EQUAL
1365  012116  104015                            ERROR   15                      ;CH 53 FAILED TO EQUAL EXPECTED VALUE
1366                                    ;;*****************************************************************
 (3)                                    ;*TEST 63        TEST THAT NON-STORE L (CH 55) IS GREATER THAN +3 VOLTS WHEN HIGH
 (3)                                    ;;*****************************************************************
 (2)  012120  000004                    TST63:  SCOPE
1367  012122  012777  017000  167314            MOV     #BIT12!BIT11!BIT10!BIT9,@STAT           ;SET STORE MODE
1368  012130  012737  001146  001124            MOV     #1146,$GDDAT            ;LOAD EXPECTED VALUE
1369  012136  004537  014370                    JSR     R5,CONVRT               ;CONVERT
1370  012142  000055                            CH55                            ;CH 55
1371  012144  013737  014526  001126            MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1372  012152  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1373  012160  003401                            BLE     TST64           ;;BR IF LESS THAN OR EQUAL
1374  012162  104014                            ERROR   14                      ;CH 55 FAILED TO EQUAL EXPECTED VALUE
1375                                    ;;*****************************************************************
 (3)                                    ;*TEST 64        TEST THAT DELAY RETURN SETS READY
 (3)                                    ;;*****************************************************************
 (2)  012164  000004                    TST64:  SCOPE
1376  012166  012777  000020  167250            MOV     #BIT4,@STAT             ;LOAD EXT DELAY BIT
1377  012174  012737  000020  001124            MOV     #BIT4,$GDDAT            ;LOAD EXPECTED
1378  012202  005277  167236                    INC     @STAT                   ;MAKE READY GO AWAY
1379  012206  017737  167232  001126            MOV     @STAT,$BDDAT            ;READ STAT
1380  012214  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1381  012222  001401                            BEQ     1$              ;;BR IF EXPECTED
1382  012224  104001                            ERROR   1                       ;READY FAILED TO CLEAR
1383  012226  052777  002000  167210  1$:       BIS     #BIT10,@STAT            ;LOAD NON-STORE L
1384  012234  012737  002220  001124            MOV     #BIT10!BIT7!BIT4,$GDDAT ;LOAD EXPECTED
1385  012242  017737  167176  001126            MOV     @STAT,$BDDAT            ;READ THE REGISTER
1386  012250  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1387  012256  001401                            BEQ     2$              ;;BR IF SAME
1388  012260  104001                            ERROR   1                       ;EXT. DELAY RETURN FAILED TO SET READY
1389  012262  005077  167156          2$:       CLR     @STAT
```

```
1391                                    ;;*****************************************************************
 (3)                                    ;*TEST 65       TEST THAT CHANNEL 02 L (CH 56) IS GREATER THAN +3 VOLTS WHEN HIGH
 (3)                                    ;;*****************************************************************
 (2)  012266  000004                    TST65:  SCOPE
1392  012270  012777  014000  167146            MOV     #BIT12!BIT11,@STAT      ;LOAD STATUS
1393  012276  012737  001100  001124            MOV     #1100,$GDDAT            ;LOAD EXPECTED VALUE
1394  012304  004537  014370                    JSR     R5,CONVRT               ;CONVERT
1395  012310  000056                            CH56                            ;CH 56
1396  012312  013737  014526  001126            MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1397  012320  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1398  012326  003401                            BLE     TST66           ;;BR IF LESS THAN OR EQUAL
1399  012330  104014                            ERROR   14                      ;CH 56 FAILED TO EQUAL EXPECTED
1400                                                                            ; VALUE
1401
1402                                    ;;*****************************************************************
 (3)                                    ;*TEST 66       TEST THAT CHANNEL 02 L (CH 56) IS LESS THAN +400 MVOLTS WHEN LOW
 (3)                                    ;;*****************************************************************
 (2)  012332  000004                    TST66:  SCOPE
1403  012334  012777  003000  167102            MOV     #BIT9!BIT10,@STAT       ;SET CHANNEL 2
1404  012342  012737  000120  001124            MOV     #120,$GDDAT             ;LOAD EXPECTED VALUE
1405  012350  004537  014370                    JSR     R5,CONVRT               ;CONVERT
1406  012354  000056                            CH56                            ;CH 56
1407  012356  013737  014526  001126            MOV     ADEND,$BDDAT            ;LOAD VALUE READ
1408  012364  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1409  012372  002001                            BGE     1$              ;;BR IF GREATER THAN
1410  012374  104015                            ERROR   15                      ;CH 56 FAILED TO EQUAL EXPECTED
1411                                                                            ; VALUE
1412  012376  005077  167042           1$:     CLR     @STAT                   ;CLEAR BIT 9
1413
```

```
1415                                    ;;*****************************************************************
 (3)                                    ;*TEST 67       TEST THAT "ERASE" AND DONE CAN BE CLEARED AND SET
 (3)                                    ;;*****************************************************************
 (2)  012402  000004                    TST67:  SCOPE
1416  012404  005077  167034                      CLR     @STAT                   ;CLEAR
1417  012410  005037  001124                      CLR     $GDDAT                  ;CLEAR EXPECTED
1418  012414  012777  010000  167022              MOV     #BIT12,@STAT            ;LOAD EXPECTED
1419  012422  017737  167016  001126              MOV     @STAT,$BDDAT            ;READ REG
1420  012430  023737  001124  001126              CMP     $GDDAT,$BDDAT           ;COMPARE
1421  012436  001401                              BEQ     1$              ;;BR IF SET
1422  012440  104001                              ERROR   1                       ;ERASE FAILED TO CLEAR READY, CHECK G5C36-
1423                                                                          ;BC08R CONNECTION: A-VV, JV-A
1424  012442  052777  001000  002054  1$:         BIS     #BIT9,@GSTAT            ;SET CH 2 (GENERATE ERASE RETURN OF GOOD AR11)
1425  012450  042777  001000  002046              BIC     #BIT9,@GSTAT            ;CLEAR CH 2
1426  012456  012737  000200  001124              MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
1427  012464  017737  166754  001126              MOV     @STAT,$BDDAT            ;READ REG
1428  012472  023737  001124  001126              CMP     $GDDAT,$BDDAT           ;COMPARE
1429  012500  001401                              BEQ     TST70           ;;BR IF EQUAL
1430  012502  104001                              ERROR   1                       ;READY FAILED TO SET ON THE END
1431                                                                          ; OF ERASE RETURN (CLEARING CH 2)
```

```
1433                                    ;;*****************************************************************
 (3)                                    ;*TEST 70       ADJUSTMENT ROUTINES FOR DAC 0 - 1
 (3)                                    ;;*****************************************************************
 (2) 012504  000004                     TST70:  SCOPE
 (1) 012506  012737  000001  001166             MOV     #1,$TIMES       ;;DO 1 ITERATION
1434                                    ;OFFSET ADJUST FOR DAC 0
1435  012514  004537  013770                     JSR     R5,SNDVLT               ;LOAD THE VOLTAGE
1436  012520  020207                             N50000
1437  012522  005077  166720                     CLR     @DACO                   ;LOAD DAC REGISTER
1438  012526  004537  014062                     JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
 (1)  012532  060000                             BIT14!BIT13
1439  012534  104401                             TYPE
1440  012536  017537                             R38                             ;TELL OPR. TO ADJUST R38
1441  012540  004737  013310                     JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1442
1443                                    ;OFFSET ADJUSTMENT FOR DAC 1
1444  012544  004537  014062                     JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
 (1)  012550  020000                             BIT13
1445  012552  005077  166672                     CLR     @DAC1                   ;LOAD THE DAC REGISTER
1446  012556  104401                             TYPE
1447  012560  017604                             R40                             ;TELL OPR. TO ADJUST R40
1448  012562  004737  013310                     JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1449
1450                                    ;GAIN ADJUST FOR DAC 0
1451  012566  004537  013770                     JSR     R5,SNDVLT
1452  012572  020222                             P49976
1453  012574  012777  007777  166644             MOV     #7777,@DACO             ;LOAD DAC 0
1454  012602  004537  014062                     JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
 (1)  012606  060000                             BIT14!BIT13
1455  012610  104401                             TYPE
1456  012612  017763                             R37                             ;TELL OPR. TO ADJUST R37
1457  012614  004737  013310                     JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1458
1459                                    ;GAIN ADJUST FOR DAC 1
1460  012620  004537  014062                     JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
 (1)  012624  020000                             BIT13
1461  012626  012777  007777  166614             MOV     #7777,@DAC1             ;LOAD DAC 1
1462  012634  104401                             TYPE
1463  012636  020030                             R39                             ;TELL OPR. TO ADJUST R39
1464  012640  004737  013310                     JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1465
1466                                    ;DAC 0 D/A FUNCTION CHECK
1467  012644  004537  014266                     JSR     R5,FUNDAC               ;GO TO DAC FUNCTION SUB.
1468  012650  001446                             DACO
1469  012652  000051                             CH51
1470
1471                                    ;DAC 1 D/A FUNCTION CHECK
1472  012654  004537  014266                     JSR     R5,FUNDAC
1473  012660  001450                             DAC1
1474  012662  000052                             CH52
1475
```

```
1477                                    ;;*****************************************************************
 (3)                                    ;*TEST 71       ADJUSTMENT ROUTINES FOR DAC 2 - 3
 (3)                                    ;;*****************************************************************
 (2) 012664  000004                     TST71:  SCOPE
 (1) 012666  012737  000001  001166             MOV     #1,$TIMES       ;;DO 1 ITERATION
1478                                    ;NOW DO  DAC 2 AND 3
1479  012674  104401                           TYPE
1480  012676  017100                           SELD23
1481                                    ;OFFSET ADJUST TOR DAC 2
1482  012700  004537  013770                   JSR     R5,SNDVLT               ;LOAD E.D.C
1483  012704  020207                           N50000
1484  012706  005077  166540                   CLR     @DAC2                   ;LOAD DAC VALUE
1485  012712  004537  014062                   JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
 (1)  012716  060000                           BIT14!BIT13
1486
1487  012720  104401                           TYPE
1488  012722  017651                           R42                             ;TELL OPR. TO ADJUST R42
1489  012724  004737  013310                   JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1490
1491                                    ;OFFSET ADJUST FOR DAC 3
1492  012730  005077  166520                   CLR     @DAC3
1493  012734  004537  014062                   JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
 (1)  012740  020000                           BIT13
1494  012742  005077  166506                   CLR     @DAC3                   ;LOAD THE DAC REGISTER
1495  012746  104401                           TYPE
1496  012750  017716                           R44                             ;TELL OPR. TO ADJUST R44
1497  012752  004737  013310                   JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1498
1499                                    ;GAIN ADJUST FOR DAC 2
1500  012756  004537  013770                   JSR     R5,SNDVLT
1501  012762  020222                           P49976
1502  012764  012777  007777  166460           MOV     #7777,@DAC2             ;LOAD DAC 2
1503  012772  004537  014062                   JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT14!BIT13
 (1)  012776  060000                           BIT14!BIT13
1504
1505  013000  104401                           TYPE
1506  013002  020075                           R41                             ;TELL OPR. TO ADJUST R41
1507  013004  004737  013310                   JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1508
1509                                    ;GAIN ADJUST FOR DAC 3
1510  013010  012777  007777  166436           MOV     #7777,@DAC3             ;LOAD DAC 3
1511  013016  004537  014062                   JSR     R5,CROSS                ;LOAD THE RELAYS WITH BIT13
 (1)  013022  020000                           BIT13
1512
1513  013024  104401                           TYPE
1514  013026  020142                           R43                             ;TELL OPR. TO ADJUST R43
1515  013030  004737  013310                   JSR     PC,CSPACE               ;WAIT FOR OPR. TO TYPE "SPACE"
1516
```

```
1518                                    ;DAC 2 D/A FUNCTION CHECK
1519
1520  013034  004537  014266                      JSR     R5,FUNDAC
1521  013040  001452                              DAC2
1522  013042  000051                              CH51
1523
1524                                    ;DAC 3 D/A FUNCTION CHECK
1525
1526  013044  004537  014266                      JSR     R5,FUNDAC
1527  013050  001454                              DAC3
1528  013052  000052                              CH52
1529
1530  013054  104401                              TYPE
1531  013056  017145                              CALDON
1532  013060  000137  011506                      JMP     AUTCAL
1533
1534  013064  022626                  AUTRAP:     CMP     (SP)+,(SP)+
1535  013066  012737  000006  000004              MOV     #6,ERRVEC
1536  013074  104401                              TYPE
1537  013076  017411                              AUTOER
1538  013100  000137  002312                      JMP     CTRLC
```

```
1540                                            .SBTTL  MANUAL LOGIC TEST
1541
1542  013104  012706  001100          MANUL:    MOV     #STACK,SP               ;LOAD STACK
1543  013110  004737  001742                    JSR     PC,LDTRAP               ;LOAD BUS ADDRESS
1544  013114  104401                            TYPE
1545  013116  017204                            MANHED
1546  013120  004737  013246          1$:       JSR     PC,ACTRLC               ;TEST FOR CTRL C
1547  013124  104407                            CKSWR
1548  013126  017700  166006                    MOV     @SWR,R0                 ;READ THE SWITCHES
1549  013132  010001                            MOV     R0,R1                   ;COPY
1550  013134  042700  017777                    BIC     #17777,R0               ;MASK OFF BITS
1551  013140  001003                            BNE     2$                      ;BR IF NOT DAC 0
1552  013142  013702  001446                    MOV     DAC0,R2                 ;LOAD DAC 0 BUS ADDRESS
1553  013146  000424                            BR      10$
1554  013150  022700  020000          2$:       CMP     #BIT13,R0               ;TEST FOR DAC 1
1555  013154  001003                            BNE     3$                      ;BR IF NOT
1556  013156  013702  001450                    MOV     DAC1,R2                 ;LOAD DAC 1 BUS ADDRESS
1557  013162  000416                            BR      10$
1558  013164  022700  040000          3$:       CMP     #BIT14,R0               ;TEST FOR DAC 2
1559  013170  001003                            BNE     4$                      ;BR IF NOT
1560  013172  013702  001452                    MOV     DAC2,R2                 ;LOAD DAC 2 BUS ADDRESS
1561  013176  000410                            BR      10$
1562  013200  022700  060000          4$:       CMP     #BIT14!BIT13,R0         ;TEST FOR DAC 3
1563  013204  001003                            BNE     5$                      ;BR IF NOT
1564  013206  013702  001454                    MOV     DAC3,R2                 ;LOAD DAC 3 BUS ADDRESS
1565  013212  000402                            BR      10$
1566  013214  013702  001444          5$:       MOV     STAT,R2                 ;LOAD STATUS REG. BUS ADDRESS
1567
1568  013220  010112                  10$:      MOV     R1,(R2)                 ;LOAD THE SWITCHES INTO THE SELECTED ADDRESS
1569  013222  013703  001442                    MOV     DELAY,R3                ;PRESET THE DELAY
1570  013226  005303                  11$:      DEC     R3                      ;DELAY
1571  013230  001376                            BNE     11$
1572  013232  005012                            CLR     (R2)                    ;CLEAR THE REGISTER
1573  013234  013703  001442                    MOV     DELAY,R3                ;LOAD THE PRESET
1574  013240  005303                  12$:      DEC     R3                      ;DELAY
1575  013242  001376                            BNE     12$
1576  013244  000725                            BR      1$
```

```
1578
1579                                    ;TEST FOR CTRL C  -- DONT WAIT
1580
1581  013246  105777  165672          ACTRLC: TSTB    @$TKS           ;INPUT FLAG ?
1582  013252  100014                          BPL     1$                      ;NO
1583  013254  017737  165666  013306          MOV     @$TKB,10$               ;READ THE CHARACTER
1584  013262  042737  177600  013306          BIC     #177600,10$
1585  013270  022737  000003  013306          CMP     #3,10$                  ;TEST FOR CTRL C
1586  013276  001002                          BNE     1$                      ;BR IF NOT
1587  013300  000137  002312                  JMP     CTRLC
1588  013304  000207                  1$:     RTS     PC                      ;EXIT
1589  013306  000000                  10$:    0
1590
1591                                    ;WAIT FOR A "SPACE" CHARACTER
1592
1593  013310  105777  165630          CSPACE: TSTB    @$TKS           ;WAIT FOR CHAR
1594  013314  100375                          BPL     CSPACE
1595  013316  017737  165624  013366          MOV     @$TKB,10$               ;READ CHAR
1596  013324  042737  177600  013366          BIC     #177600,10$             ;MASK
1597  013332  022737  000003  013366          CMP     #3,10$                  ;TEST FOR CTRL C
1598  013340  001002                          BNE     1$
1599  013342  000137  002312                  JMP     CTRLC
1600  013346  022737  000040  013366  1$:     CMP     #40,10$                 ;TEST FOR SPACE
1601  013354  001001                          BNE     2$
1602  013356  000207                          RTS     PC                      ;EXIT
1603  013360  104401                  2$:     TYPE
1604  013362  016746                          QMARK
1605  013364  000751                          BR      CSPACE
1606
1607  013366  000000                  10$:    0
```

```
1609
1610                                            .SBTTL  MANUAL DAC CALIBRATION
1611
1612  013370  012706  001100          CALDAC: MOV     #STACK,SP               ;LOAD STACK POINTER
1613  013374  004737  001742                  JSR     PC,LDTRAP               ;LOAD BUS ADDRESSES
1614  013400  104407                  1$:     CKSWR                           ;TEST FOR CTRL G
1615  013402  004737  013246                  JSR     PC,ACTRLC               ;TEST FOR CTRL C
1616  013406  017700  165526                  MOV     @SWR,R0                 ;READ SWITCHES
1617  013412  010077  166030                  MOV     R0,@DAC0                ;LOAD DAC #0
1618  013416  010077  166026                  MOV     R0,@DAC1                ;LOAD DAC #1
1619  013422  010077  166024                  MOV     R0,@DAC2                ;LOAD DAC #2
1620  013426  010077  166022                  MOV     R0,@DAC3                ;LOAD DAC #3
1621  013432  000762                          BR      1$
1622
1623                                            .SBTTL  DYNAMIC DAC CALIBRATION
1624
1625  013434  012706  001100          DYNCAL: MOV     #STACK,SP               ;LOAD STACK POINTER
1626  013440  004737  001742                  JSR     PC,LDTRAP               ;LOAD BUS ADDRESSES
1627  013444  104407                  1$:     CKSWR                           ;TEST FOR CTRL G
1628  013446  004737  013246                  JSR     PC,ACTRLC               ;TEST FOR CTRL C
1629  013452  017700  165462                  MOV     @SWR,R0                 ;READ SWR
1630  013456  004737  013472                  JSR     PC,10$                  ;LOAD THE SWR VALUE TO ALL DAC'S
1631  013462  005000                          CLR     R0                      ;CLEAR R0
1632  013464  004737  013472                  JSR     PC,10$                  ;LOAD ALL DAC'S WITH 0
1633  013470  000765                          BR      1$
1634
1635  013472  010077  165750          10$:    MOV     R0,@DAC0                ;LOAD DAC #0
1636  013476  010077  165746                  MOV     R0,@DAC1                ;LOAD DAC #1
1637  013502  010077  165744                  MOV     R0,@DAC2                ;LOAD DAC #2
1638  013506  010077  165742                  MOV     R0,@DAC3                ;LOAD DAC #3
1639  013512  012700  000020                  MOV     #20,R0                  ;LOAD DELAY COUNTER
1640  013516  005300                  11$:    DEC     R0                      ;DELAY
1641  013520  100376                          BPL     11$                     ;WAIT
1642  013522  000207                          RTS     PC                      ;EXIT
1643
1644
```

```
1651                                                    .SBTTL  SUBROUTINE TO ERASE STORAGE SCOPE SCREEN
1652  013524  104407                          CLRVC:    CKSWR
1653  013526  032777  000040  165404                    BIT     #BIT5,@SWR      ;TEST SR BIT 5
1654  013534  001441                                    BEQ     3$              ;BR IF NOT A STORAGE SCOPE
1655  013536  012777  002000  165700                    MOV     #BIT10,@STAT    ;ERASE THE SCREEN
1656  013544  052777  010000  165672                    BIS     #BIT12,@STAT
1657  013552  012700  000020                          MOV     #20,R0          ;SET UP DELAY
1658  013556  005001                                    CLR     R1
1659  013560  032777  000040  165352  1$:               BIT     #BIT5,@SWR      ;TEST IF NOT STORAGE SCOPE
1660  013566  001424                                    BEQ     3$
1661  013570  105777  165650                          TSTB    @STAT           ;TEST FOR READY
1662  013574  100421                                    BMI     3$              ;BRANCH IF SET
1663  013576  005301                                    DEC     R1              ;DELAY
1664  013600  001367                                    BNE     1$
1665  013602  004737  013246                          JSR     PC,ACTRLC       ;TEST FOR CTRL C
1666  013606  005300                                    DEC     R0              ;DELAY
1667  013610  001363                                    BNE     1$
1668  013612  037727  165322  010000                    BIT     @SWR,#SW12      ;TEST INHIBIT PRINTOUT
1669  013620  001002                                    BNE     2$
1670  013622  104401                                    TYPE
1671  013624  015336                                    MES3
1672  013626  104407                          2$:       CKSWR                   ;TEST FOR CTRL C
1673  013630  005777  165304                          TST     @SWR            ;TEST @SWR
1674  013634  100001                                    BPL     3$
1675  013636  000000                                    HALT                    ;ERASE RETURN FAILED TO SET READY
1676  013640  000207                          3$:       RTS     PC
1677
1678                                                    .SBTTL  SUBROUTINE TO DRAW A HORIZONTAL LINE
1679  013642  005077  165576                LOADVC:   CLR     @STAT           ;CLEAR STATUS
1680  013646  012737  007777  020420                    MOV     #7777,TEMP1
1681  013654  013700  001444                          MOV     STAT,R0
1682  013660  032777  000010  165252                    BIT     #SW03,@SWR      ;TEST SR BIT 3
1683  013666  001405                                    BEQ     4$              ;BR IF DAC #0 AND 1
1684  013670  013701  001452                          MOV     DAC2,R1 ;LOAD X AXIS
1685  013674  013702  001454                          MOV     DAC3,R2 ;LOAD Y AXIS
1686  013700  000404                                    BR      5$
1687  013702  013701  001446          4$:               MOV     DAC0,R1
1688  013706  013702  001450                          MOV     DAC1,R2
1689  013712  012710  002000          5$:               MOV     #BIT10,(0)      ;SET STORE MODE
1690  013716  013712  020420                          MOV     TEMP1,(2)
1691  013722  012711  007777          1$:               MOV     #7777,(1)
1692  013726  000402                                    BR      3$
1693  013730  162711  000010          2$:               SUB     #10,(1)
1694  013734  005210                  3$:               INC     (0)
1695  013736  105710                                    TSTB    (0)
1696  013740  100376                                    BPL     .-2
1697  013742  022711  000007                          CMP     #7,(1)
1698  013746  001370                                    BNE     2$
1699  013750  104407                                    CKSWR                   ;TEST FOR CTRL G
1700  013752  004737  013246                          JSR     PC,ACTRLC               ;TEST FOR CTRL C
1701  013756  162712  000010                          SUB     #10,(2)
1702  013762  100357                                    BPL     1$
1703  013764  000207                                    RTS     PC
1704
```

```
1706  013766  167772                    FILZ:   167772                  ;DR11-C OUTPUT CONTROL REGISTER
1707
1708                                    ;SUBROUTINE TO LOAD THE EDC VOLTAGE
1709                                    ;DATA FORMAT IS:        STX
1710                                    ;                       P OR N
1711                                    ;                       N VOLTS
1712                                    ;                       0 TENTHS VOLTS
1713                                    ;                       0
1714                                    ;                       0
1715                                    ;                       0
1716                                    ;                       0
1717                                    ;                       V       VOLTS
1718                                    ;                       ETX
1719
1720  013770  012500                    SNDVLT: MOV     (R5)+,R0                ;LOAD POINTER
1721  013772  112001                    2$:     MOVB    (R0)+,R1                ;GET A BYTE
1722  013774  001422                            BEQ     3$                      ;BR IF TERM.
1723  013776  005101                            COM     R1                      ;CONVERT IT
1724  014000  110177  177762                   MOVB    R1,@FILZ                ;LOAD FUNNY DATA BYTE
1725  014004  012701  001000                   MOV     #1000,R1                ;LOAD DELAY
1726  014010  005301                    5$:     DEC     R1                      ;DELAY
1727  014012  001376                            BNE     5$
1728  014014  042777  000200  177744           BIC     #BIT7,@FILZ             ;CLEAR BIT 7
1729  014022  012701  001000                   MOV     #1000,R1                ;LOAD DELAY COUNT
1730  014026  005301                    1$:     DEC     R1                      ;DELAY
1731  014030  001376                            BNE     1$
1732  014032  052777  000200  177726           BIS     #BIT7,@FILZ             ;SET BIT 7
1733  014040  000754                            BR      2$                      ;DO MORE DATA
1734
1735  014042  012701  000000            3$:     MOV     #0,R1                   ;LOAD DELAY
1736  014046  152777  000177  177712           BISB    #177,@FILZ              ;SET BITS 0-6
1737  014054  005301                    4$:     DEC     R1                      ;DELAY
1738  014056  001376                            BNE     4$
1739  014060  000205                            RTS     R5                      ;EXIT
1740
1741  014062  012537  014122            CROSS:  MOV     (R5)+,10$       ;READ REG.
1742  014066  052737  000177  014122            BIS     #177,10$                ;LOAD LOW 7 BITS
1743  014074  013777  014122  177664           MOV     10$,@FILZ               ;LOAD RELAYS
1744  014102  012700  000001                   MOV     #1,R0                   ;LOAD DELAY
1745  014106  005001                            CLR     R1
1746  014110  005301                    1$:     DEC     R1                      ;DELAY
1747  014112  001376                            BNE     1$
1748  014114  005300                            DEC     R0                      ;DELAY
1749  014116  001374                            BNE     1$                      ;DONE ?
1750  014120  000205                            RTS     R5                      ;EXIT
1751  014122  000000                    10$:    0
```

```
                                        .SBTTL  TIMER ROUTINE FOR VISUAL TEST PATTERNS
                                ; ENTER VIA JSR PC,TIMER
                                
014124  104407                  TIMER:  CKSWR
014126  004737  013246                  JSR     PC,ACTRLC               ;TEST FOR CTRL C
014132  017737  165002  020412          MOV     @SWR,TIMSV
                                
014140  032737  000400  020412  TIMERA: BIT     #BIT8,TIMSV
014146  001006                          BNE     TIMER2          ;BIT 8 SET ?
014150  005337  020414                  DEC     TICKS           ;NO, DECREMENT TICKS
014154  001002                          BNE     TIMER1
014156  062716  000002                  ADD     #2,(6)          ;ADD 2 TO STACK POINTER
014162  000207                  TIMER1: RTS     PC              ;RETURN
                                
                                ; SWR 8=1 SELECT TEST TO LOCK ON
                                ; SWR 2-0= TEST NUMBER
                                
014164  042737  177770  020412  TIMER2: BIC     #177770,TIMSV
014172  006337  020412                  ASL     TIMSV
014176  062737  014220  020412          ADD     #ROUTPT,TIMSV
014204  017737  004202  020412          MOV     @TIMSV,TIMSV
014212  022600                          CMP     (SP)+,R0
014214  000177  004172          TIMER4: JMP     @TIMSV
                                
014220  006664                  ROUTPT: PIC0                    ;DISPLAY A HORIZONTAL LINE
014222  006740                          PIC1                    ;DISPLAY A VERTICAL LINE
014224  007116                          PIC3                    ;DISPLAY A SQUARE
014226  007346                          PIC4                    ;DISPALY A "X"
014230  007554                          PIC5                    ;DISPLAY SETTLING TIME
014232  010012                          PIC6                    ;DISPLAY CHARACTER SET
014234  011030                          PIC7                    ;DISPLAY CHANNEL TEST
014236  011334                          PIC12                   ;DISPLAY ERASE AND PHOSPHOR TEST
                                
014240  013737  014264  020422  CHTIME: MOV     CPTYPE,BRLEV1
014246  005337  020422          1$:     DEC     BRLEV1
014252  001403                          BEQ     2$
014254  006337  020414                  ASL     TICKS
014260  000772                          BR      1$
014262  000207                  2$:     RTS     PC
                                
014264  000001                  CPTYPE: 1
```

```
1795
1796                                    ;DO 32 CONVERSIONS ON EACH PRESET VALUE OF THE DAC'S
1797                                    ;AND SUBTRACT THE FIRST RESULT FROM EACH OF THE RESULTS
1798
1799  014266  013537  014366  FUNDAC: MOV     @(R5)+,XDACUT                   ;GET BUS ADDRESS OF ARUT DAC
1800  014272  012537  014316          MOV     (R5)+,60$                       ;GET CHANNEL INPUT FOR ARKG
1801  014276  012777  007600  000062  MOV     #7600,@XDACUT                   ;PRESET THE DAC
1802  014304  012737  001700  001124  MOV     #1700,$GDDAT                    ;LOAD EXPECTED
1803
1804  014312  004537  014370  1$:     JSR     R5,CONVRT                       ;CONVERT USING ARKG
1805  014316  000051          60$:    CHS1                                    ;
1806
1807  014320  012737  000010  014534  MOV     #10,SPREAD                      ;LOAD LIMIT
1808  014326  004737  014536          JSR     PC,COMPAR                       ;TEST IF WITHIN LIMIT
1809  014332  000401                  BR      2$              ;;BR IF WITHIN
1810  014334  104016                  ERROR   16              ;SELECTED DAC HAS A LINEARITY ERROR
1811
1812  014336  162777  000400  000022  2$:     SUB     #400,@XDACUT            ;UPDATE DAC
1813  014344  162737  000200  001124          SUB     #200,$GDDAT             ;UPDATE GOOD VALUE
1814  014352  100357                  BPL     1$              ;;BR IF NOT FINISHED
1815  014354  000205                  RTS     R5                              ;EXIT
1816  014356  000240                  NOP
1917  014360  000240                  NOP
1918  014362  000240                  NOP
1819
1920  014364  000000          10$:    0
1621  014366  000000          XDACUT: 0
```

```
1823                                    ;SUBROUTINE TO GET AVERAGE OF 32 CONVERSIONS FROM THE KNOWN GOOD AR 11
1824
1825  014370  012537  014510            CONVRT: MOV     (R5)+,10$
1826  014374  013737  014510  014532            MOV     10$,CHANL
1827  014402  000337  014510                    SWAB    10$
1828  014406  012737  000020  014512            MOV     #16.,11$
1829  014414  005037  014526                    CLR     ADEND
1830  014420  013777  014510  000072            MOV     10$,@GADCS
1831  014426  005277  000066            2$:     INC     @GADCS
1832  014432  105777  000062            1$:     TSTB    @GADCS
1833  014436  100375                            BPL     1$
1834  014440  067737  000056  014526            ADD     @GADDBR,ADEND
1835  014446  005337  014512                    DEC     11$
1836  014452  001365                            BNE     2$
1837  014454  006237  014526                    ASR     ADEND
1838  014460  006237  014526                    ASR     ADEND
1839  014464  006237  014526                    ASR     ADEND
1840  014470  006237  014526                    ASR     ADEND
1841  014474  005537  014526                    ADC     ADEND                   ;ROUND UP
1842  014500  013737  014526  001126            MOV     ADEND,$BDDAT
1843  014506  000205                            RTS     R5
1844
1845  014510  000000            10$:    0
1846  014512  000000            11$:    0
1847  014514  001754            V1754:  1754
1848  014516  000024            V24:    24
1849  014520  170440            GADCS:  170440
1850  014522  170442            GADDBR: 170442
1851  014524  170450            GSTAT:  170450
1852  014526  000000            ADEND:  0
1853  014530  000000            FCHANL: 0
1854  014532  000000            CHANL:  0
1855  014534  000000            SPREAD: 0
1856
1857  014536  010046            COMPAR: MOV     R0,-(SP)
1858  014540  010146                    MOV     R1,-(SP)
1859  014542  013700  001124            MOV     $GDDAT,R0               ;LOAD R0
1860  014546  013701  001126            MOV     $BDDAT,R1               ;LOAD R1
1861  014552  160100                    SUB     R1,R0                   ;SUBTRACT
1862  014554  100001                    BPL     8$                      ;
1863  014556  005400                    NEG     R0
1864  014560  020037  014534    8$:     CMP     R0,SPREAD               ;MAGNITUDE OF DIFF. IN R0
1865  014564  003405                    BLE     10$
1866  014566  012601            9$:     MOV     (SP)+,R1
1867  014570  012600                    MOV     (SP)+,R0
1868  014572  062716  000002            ADD     #2,(SP)
1869  014576  000207                    RTS     PC
1870
1871  014600  012601            10$:    MOV     (SP)+,R1
1872  014602  012600                    MOV     (SP)+,R0
1873  014604  000207                    RTS     PC
1874
```

```
                                                .SBTTL  ASCII MESSAGES
                                                
014606 005015 040412 030501  TITLE:  .ASCII  <15><12><12>'AA11-K DIAGNOSTIC TEST, [MAINDEC-11-DZAAC-BB]'<15><12>
014614 026461 020113 044504
014622 043501 047516 052123
014630 041511 052040 051505
014636 026124 024040 040515
014644 047111 042504 026503
014652 030461 042055 040532
014660 041501 041055 024460
014666 005015
014670 042523 042514 052103          .ASCII  /SELECT TESTS BY TYPING A TWO LETTER I.D./<15><12>
014676 052040 051505 051524
014704 041040 020131 054524
014712 044520 043516 040440
014720 052040 047527 046040
014726 052105 042524 020122
014734 027111 027104 005015
014742 046101 035011 052501          .ASCII  /AL     :AUTO LOGIC TEST/<15><12>
014750 047524 046040 043517
014756 041511 052040 051505
014764 006524     012
014767     101 004504 040472          .ASCII  /AD     :AUTO DISPLAY TEST (DISPLAY SCOPE CONNECTED)/<15><12>
014774 052125 020117 044504
015002 050123 040514 020131
015010 042524 052123 024040
015016 044504 050123 040514
015024 020131 041523 050117
015032 020105 047503 047116
015040 041505 042524 024504
015046 005015
015050 041501 035011 052501          .ASCII  /AC     :AUTO CALIBRATION (AA11-K OPTION TEST AREA ONLY)/<15><12>
015056 047524 041440 046101
015064 041111 040522 044524
015072 047117 024040 040501
015100 030461 045055 047440
015106 052120 047511 020116
015114 042524 052123 040440
015122 042522 020101 047117
015130 054514 006451     012
015135     115 004514 046472          .ASCII  /ML     :MANUAL LOGIC TEST (SWR 15-13 = REGISTER  12-0 = DATA)/<15><12>
015142 047101 040525 020114
015150 047514 044507 020103
015156 042524 052123 024040
015164 053523 020122 032461
015172 030455 020063 020075
015200 042522 044507 052123
015206 051105 020040 031061
015214 030055 036440 042040
015222 052101 024501 005015
015230 042115 035011 040515          .ASCII  /MD     :MANUAL DISPLAY (NO DISPLAY SCOPE)/<15><12>
015236 052516 046101 042040
015244 051511 046120 054501
015252 024040 047516 042040
```

```
                015260  051511  046120  054501
                015266  051440  047503  042520
                015274  006451      012
  1885          015277      115 004503  046472          .ASCII  /MC     :MANUAL CALIBRATION LOOP/<15><12>
                015304  047101  040525  020114
                015312  040503  044514  051102
                015320  052101  047511  020116
                015326  047514  050117  005015
  1886          015334  000056                          .ASCIZ  /./
  1887          015336  005015  051105  051501  MES3:   .ASCIZ  <15><12>"ERASE RETURN FAILED TO SET READY"<15><12>
                015344  020105  042522  052524
                015352  047122  043040  044501
                015360  042514  020104  047524
                015366  051440  052105  051040
                015374  040505  054504  005015
                015402      000
  1888          015403      015 051412  047503  MES6:   .ASCIZ  <15><12>'SCOPE ADJUSTMENT TEST'
                015410  042520  043440  045104
                015416  051525  046524  047105
                015424  020124  042524  052123
                015432      000
  1889          015433      015 051412  051127  MES15:  .ASCII  <15><12>/SWR8 AND C THRU 2 CONTROL PATTERN/<15><12>
                015440  020070  047101  020104
                015446  020060  044124  052522
                015454  031040  041440  047117
                015462  051124  046117  050040
                015470  052101  042524  047122
                015476  005015
  1890          015500  053523  020122  044502          .ASCII  'SWR BIT 3 SELECTS DAC 0+1 OR DAC 2+3'<15><12>
                015506  020124  020063  042523
                015514  042514  052103  020123
                015522  040504  020103  025460
                015530  020061  051117  042040
                015536  041501  031040  031453
                015544  005015
  1891          015546  053523  020122  044502          .ASCII  /SWR BIT 5 SELECTS STORAGE SCOPE MODE/<15><12>
                015554  020124  020065  042523
                015562  042514  052103  020123
                015570  052123  051117  043501
                015576  020105  041523  050117
                015604  020105  047515  042504
                015612  005015
  1892          015614  053523  020122  044502          .ASCII  /SWR BIT 7 ENABLES VERTICAL SETTLING TEST/<15><12>
                015622  020124  020067  047105
                015630  041101  042514  020123
                015636  042526  052122  041511
                015644  046101  051440  052105
                015652  046124  047111  020107
                015660  042524  052123  005015
  1893          015666  053523  020122  044502          .ASCIZ  /SWR BIT 9 SELECTS TWO'S COMPLEMENT DISPLAY MODE/<15><12>
                015674  020124  020071  042523
                015702  042514  052103  020123
                015710  053524  023517  020123
                015716  047503  050115  042514
```

```
                016724  042515  052116  042040
                016732  051511  046120  054501
                016740  046440  042117  006505
                016746  000012
 1894           016750  052123  052101  051525  EM1:    .ASCIZ  /STATUS REGISTER IN ERROR/
                016756  051040  043505  051511
                016764  042524  020122  047111
                016772  042440  051122  051117
                017000     000
 1895           017001     104  041501  020060  EM2:    .ASCIZ  /DAC0 REGISTER IN ERROR/
                017006  042522  044507  052123
                017014  051105  044440  020116
                017022  051105  047522  000122
 1896           017030  040504  030503  051040  EM3:    .ASCIZ  /DAC1 REGISTER IN ERROR/
                017036  043505  051511  042524
                017044  020122  047111  042440
                017052  051122  051117     000
 1897           017057     104  041501  020062  EM4:    .ASCIZ  /DAC2 REGISTER IN ERROR/
                017064  042522  044507  052123
                017072  051105  044440  020116
                017100  051105  047522  000122
 1898           017106  040504  031503  051040  EM5:    .ASCIZ  /DAC3 REGISTER IN ERROR/
                017114  043505  051511  042524
                017122  020122  047111  042440
                017130  051122  051117     000
 1899           017135     101  030501  026461  EM6:    .ASCIZ  /AA11-K FAILED TO INTERRUPT/
                017142  020113  040506  046111
                017150  042105  052040  020117
                017156  047111  042524  051122
                017164  050125  000124
 1900           017170  040501  030461  045455  EM7:    .ASCIZ  /AA11-k INTERRUPTED IN ERROR/
                017176  044440  052116  051105
                017204  052522  052120  042105
                017212  044440  020116  051105
                017220  047522  000122
 1901           017224  052502  020123  044524  EM10:   .ASCIZ  /BUS TIME-OUT WHEN REFERENCING THE AA11-K/
                017232  042515  047455  052125
                017240  053440  042510  020116
                017246  042522  042506  042522
                017254  041516  047111  020107
                017262  044124  020105  040501
                017270  030461  045455     000
 1902           017275     101  030501  026461  EM11:   .ASCIZ  /AA11-K READY BIT ERROR/
                017302  020113  042522  042101
                017310  020131  044502  020124
                017316  051105  047522  000122
 1903           017324  047520  042527  020122  EM12:   .ASCIZ  /POWER SUPPLY VOLTAGE WAS INCORRECT/
                017332  052523  050120  054514
                017340  053040  046117  040524
                017346  042507  053440  051501
                017354  044440  041516  051117
                017362  042522  052103     000
 1904           017367     117  042516  041040  EM13:   .ASCIZ  /ONE BIT THE DUST/
                017374  052111  052040  042510
```

```
                016402  042040  051525  000124
 1905           016410  040501  030461  045455  EM14:   .ASCIZ  /AA11-K LOGIC SIGNAL HIGH OUTPUT TO LOW/
                016416  046040  043517  041511
                016424  051440  043511  040516
                016432  020114  044510  044107
                016440  047440  052125  052520
                016446  020124  047524  046040
                016454  053517     000
 1906           016457     101  030501  026461  EM15:   .ASCIZ  /AA11-K LOGIC SIGNAL LOW OUTPUT TO HIGH/
                016464  020113  047514  044507
                016472  020103  044523  047107
                016500  046101  046040  053517
                016506  047440  052125  052520
                016514  020124  047524  044040
                016522  043511  000110
 1907           016526  042523  042514  052103  EM16:   .ASCIZ  /SELECTED DAC HAS A LINEARITY ERROR/
                016534  042105  042040  041501
                016542  044040  051501  040440
                016550  046040  047111  040505
                016556  044522  054524  042440
                016564  051122  051117     000
 1908           016571     105  051122  041520  DH1:    .ASCIZ  /ERRPC  IBASE   EXPECT    BAD/
                016576  044411  040502  042523
                016604  020040  042440  050130
                016612  041505  020124  020040
                016620  041040  042101     000
 1909           016625     105  051122  041520  DH6:    .ASCIZ  /ERRPC  IBASE/
                016632  044411  040502  042523
                016640     000
 1910           016641     105  051122  041520  DH13:   .ASCIZ  /ERRPC  IBASE   # FIRST # NOW/
                016646  044411  040502  042523
                016654  021411  043040  051111
                016662  052123  021411  047040
                016670  053517     000
 1911           016673     105  051122  041520  DH14:   .ASCIZ  /ERRPC  IBASE   GOOD    BAD/
                016700  044411  040502  042523
                016706  043411  047517  004504
                016714  040502  000104
 1912           016720  051105  050122  004503  DH16:   .ASCIZ  /ERRPC  DACADR  GOOD    BAD/
                016726  040504  040503  051104
                016734  043411  047517  004504
                016742  040502  000104
 1913
 1914           016746     015     012     077  QMARK:  .BYTE   15,12,77,15,12,0
                016751     015     012     000
 1915           016754     136     103     015  CONTC:  .BYTE   136,103,15,12,56,0
                016757     012     056     000
 1916           016762     015     012          FOUND1: .BYTE   15,12
 1917           016764  051120  043517  040522          .ASCIZ  /PROGRAM DETECTED /
                016772  020115  042504  042524
                017000  052103  042105  000040
 1918           017006  034050  020051  020040  FOUND2: .ASCIZ  /(8)   AA11-K(5)  /
                017014  040501  030461  045455
                017022  051450  020051  000040
```

```
                                                .BYTE   15,12,0
                                SELD01:         .BYTE   15,12
                                                .ASCIZ  /SET SWITCH TO SELECT DAC 0 AND 1/<15><12>
                                SELD23:         .BYTE   15,12
                                                .ASCIZ  /SET SWITCH TO SELECT DAC 2 AND 3/<15><12>
                                CALDON:         .BYTE   15,12
                                                .ASCIZ  /AUTO CALIBRATION COMPLETED/<15><12>
                                MANHED:         .BYTE   15,12
                                                .ASCII  /SW 15-13 SELECT THE REGISTER/<15><12>
                                                .ASCII  /SW 12-00 ARE LOADED INTO THE SELECTED REGISTER/<15><12>
                                                .BYTE   0
                                AUTOCL:         .BYTE   15,12
                                                .ASCII  /AUTO CALIBRATION (AA11-K OPTION TEST AREA ONLY)/
                                AUTOER:         .BYTE   15,12,0
                                                .ASCII  <15><12>/BUS TRAP WHEN LOADING VOLTAGE OR CHANGING A RELAY/<15><12>
```

1919	017030	015	012	000	
1920	017033	015	012		
1921	017035	123	052105	051440	
	017042	044527	041524	020110	
	017050	047524	051440	046105	
	017056	041505	020124	040504	
	017064	020103	020060	047101	
	017072	020104	006461	000012	
1922	017100	015	012		
1923	017102	042523	020124	053523	
	017110	052111	044103	052040	
	017116	020117	042523	042514	
	017124	052103	042040	041501	
	017132	031040	040440	042116	
	017140	031440	005015	000	
1924	017145	015	012		
1925	017147	101	052125	020117	
	017154	040503	044514	051102	
	017162	052101	047511	020116	
	017170	047503	050115	042514	
	017176	042524	006504	000012	
1926	017204	015	012		
1927	017206	053523	030440	026465	
	017214	031461	051440	046105	
	017222	041505	020124	044124	
	017230	020105	042522	044507	
	017236	052123	051105	005015	
1928	017244	053523	030440	026462	
	017252	030060	040440	042522	
	017260	046040	040517	042504	
	017266	020104	047111	047524	
	017274	052040	042510	051440	
	017302	046105	041505	042524	
	017310	020104	042522	044507	
	017316	052123	051105	005015	
1929	017324	000			
1930	017325	015	012		
1931	017327	101	052125	020117	
	017334	040503	044514	051102	
	017342	052101	047511	020116	
	017350	040450	030501	026461	
	017356	020113	050117	044524	
	017364	047117	052040	051505	
	017372	020124	051101	040505	
	017400	047440	046116	024531	
1932	017406	015	012	000	
1933	017411	015	041012	051525	
	017416	052040	040522	020120	
	017424	044127	047105	046040	
	017432	040517	044504	043516	
	017440	053040	046117	040524	
	017446	042507	047440	020122	
	017454	044103	047101	044507	
	017462	043516	040440	051040	

```
                017470  046105  054501  005015
1934            017476  037477  040411  030501          .ASCIZ  /??     AA11-K OPTION TEST AREA ??      /<15><12>
                017504  026461  020113  050117
                017512  044524  047117  052040
                017520  051505  020124  051101
                017526  040505  037440  004477
                017534  005015     000
1935
1939            017537     015  040412  045104  R38:    .ASCIZ  <15><12>/ADJUST R38 FOR A NULL ON THE METER/
(1)             017544  051525  020124  031522
(1)             017552  020070  047506  020122
(1)             017560  020101  052516  046114
(1)             017566  047440  020116  044124
(1)             017574  020105  042515  042524
(1)             017602  000122
1940            017604  005015  042101  052512  R40:    .ASCIZ  <15><12>/ADJUST R40 FOR A NULL ON THE METER/
(1)             017612  052123  051040  030064
(1)             017620  043040  051117  040440
(1)             017626  047040  046125  020114
(1)             017634  047117  052040  042510
(1)             017642  046440  052105  051105
(1)             017650     000
1941            017651     015  040412  045104  R42:    .ASCIZ  <15><12>/ADJUST R42 FOR A NULL ON THE METER/
(1)             017656  051525  020124  032122
(1)             017664  020062  047506  020122
(1)             017672  020101  052516  046114
(1)             017700  047440  020116  044124
(1)             017706  020105  042515  042524
(1)             017714  000122
1942            017716  005015  042101  052512  R44:    .ASCIZ  <15><12>/ADJUST R44 FOR A NULL ON THE METER/
(1)             017724  052123  051040  032064
(1)             017732  043040  051117  040440
(1)             017740  047040  046125  020114
(1)             017746  047117  052040  042510
(1)             017754  046440  052105  051105
(1)             017762     000
1943            017763     015  040412  045104  R37:    .ASCIZ  <15><12>/ADJUST R37 FOR A NULL ON THE METER/
(1)             017770  051525  020124  031522
(1)             017776  020067  047506  020122
(1)             020004  020101  052516  046114
(1)             020012  047440  020116  044124
(1)             020020  020105  042515  042524
(1)             020026  000122
1944            020030  005015  042101  052512  R39:    .ASCIZ  <15><12>/ADJUST R39 FOR A NULL ON THE METER/
(1)             020036  052123  051040  034463
(1)             020044  043040  051117  040440
(1)             020052  047040  046125  020114
(1)             020060  047117  052040  042510
(1)             020066  046440  052105  051105
(1)             020074     000
1945            020075     015  040412  045104  R41:    .ASCIZ  <15><12>/ADJUST R41 FOR A NULL ON THE METER/
(1)             020102  051525  020124  032122
(1)             020110  020061  047506  020122
(1)             020116  020101  052516  046114
```

```
   (1)  020124  047440  020116  044124
   (1)  020132  020105  042515  042524
   (1)  020140  000122
  1946  020142  005015  042101  052512  R43:    .ASCIZ  <15><12>/ADJUST R43 FOR A NULL ON THE METER/
   (1)  020150  052123  051040  031464
   (1)  020156  043040  051117  040440
   (1)  020164  047040  046125  020114
   (1)  020172  047117  052040  042510
   (1)  020200  046440  052105  051105
   (1)  020206     000
  1947          000001                          STX=1
  1948          000003                          ETX=3
  1949  020207     001                N50000:  .BYTE   STX
  1950  020210  032516  030060  030060          .ASCII  /N500000V/
        020216  053060
  1951  020220     003     000                  .BYTE   ETX,0
  1952  020222     001                P49976:  .BYTE   STX
  1953  020223     120  034464  033471          .ASCII  /P499760V/
        020230  030066     126
  1954  020233     003     000                  .BYTE   ETX,0
  1955          020236                          .EVEN
  1956  020236  001116  001444  001124  DT1:    $ERRPC,STAT,$GDDAT,$BDDAT,0
        020244  001126  000000
  1957  020250  001116  001446  001124  DT2:    $ERRPC,DAC0,$GDDAT,$BDDAT,0
        020256  001126  000000
  1958  020262  001116  001450  001124  DT3:    $ERRPC,DAC1,$GDDAT,$BDDAT,0
        020270  001126  000000
  1959  020274  001116  001452  001124  DT4:    $ERRPC,DAC2,$GDDAT,$BDDAT,0
        020302  001126  000000
  1960  020306  001116  001454  001124  DT5:    $ERRPC,DAC3,$GDDAT,$BDDAT,0
        020314  001126  000000
  1961  020320  001116  001444  000000  DT6:    $ERRPC,STAT,0
  1962  020326  001116  001444  020426  DT13:   $ERRPC,STAT,EVER,$UNIT,0
        020334  001206  000000
  1963  020340  001116  001444  001124  DT14:   $ERRPC,STAT,$GDDAT,$BDDAT,0
        020346  001126  000000
  1964  020352  001116  014366  001124  DT16:   $ERRPC,XDACUT,$GDDAT,$BDDAT,0
        020360  001126  000000
  1965  020364  000000  000000  000000  DF0:    0,0,0,0,0,0,0,0
        020372  000000  000000  000000
        020400  000000  000000
  1966  020404  000000                  LOW:    0
  1967  020406  000000                  HIGH:   0
  1968  020410  000010                  INCR:   10
  1969  020412  000000                  TIMSV:  0
  1970  020414  000000                  TICKS:  0
  1971  020416  000000                  TEMP:   0
  1972  020420  000000                  TEMP1:  0                       ;TEMPORARY STORAGE
  1973  020422  000000                  BRLEV1: 0
  1974  020424  000000                  BRLEV2: 0
  1975  020426  000000                  EVER:   0
  1976  020430  000000                  EVER1:  0
  1977  020432  000000                  CPRIN:  0
  1978  020434  000000                  P7CNT:  0
```

```
 1979  020436  000000                 P7PNT:  0
 1980  020440  010536                 CH01:   C
 1981  020442  010567                         H
 1982  020444  010524                         A
 1983  020446  010625                         N
 1984  020450  010625                         N
 1985  020452  010550                         E
 1986  020454  010613                         L
 1987  020456  011010                         SPACEA
 1988  020460  010733                         N1
 1989  020462  010536                 CH02:   C
 1990  020464  010567                         H
 1991  020466  010524                         A
 1992  020470  010625                         N
 1993  020472  010625                         N
 1994  020474  010550                         E
 1995  020476  010613                         L
 1996  020500  011010                         SPACEA
 1997  020502  010740                         N2
 1998
 1999                                 .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)   020504                        $TYPDS:
  (3)   020504  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)   020506  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (3)   020510  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (3)   020512  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (3)   020514  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
  (1)   020516  012746  020200                MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
  (1)   020522  016605  000020                MOV     20(SP),R5       ;;GET THE INPUT NUMBER
  (1)   020526  100004                        BPL     1$              ;;BR IF INPUT IS POS.
  (1)   020530  005405                        NEG     R5              ;;MAKE THE BINARY NUMBER POS.
  (1)   020532  112766  000055  000001        MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
  (1)   020540  005000                1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
  (1)   020542  012703  020720                MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
  (1)   020546  112723  000040                MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
  (1)   020552  005002                2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
  (1)   020554  016001  020710                MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
  (1)   020560  160105                3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
  (1)   020562  002402                        BLT     4$              ;;BR IF DONE
  (1)   020564  005202                        INC     R2              ;;INCREASE THE BCD DIGIT BY 1
  (1)   020566  000774                        BR      3$
  (1)   020570  060105                4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
  (1)   020572  005702                        TST     R2              ;;CHECK IF BCD DIGIT=0
```

```
(1)  020574  001002                          BNE     5$              ;;FALL THROUGH IF 0
(1)  020576  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
(1)  020600  100407                          BMI     7$              ;;BR IF YES
(1)  020602  106316                  5$:     ASLB    (SP)            ;;MSD?
(1)  020604  103003                          BCC     6$              ;;BR IF NO
(1)  020606  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
(1)  020614  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
(1)  020620  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  020624  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  020626  005720                          TST     (R0)+           ;;JUST INCREMENTING
(1)  020630  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
(1)  020634  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
(1)  020636  003002                          BGT     8$              ;;GO TO EXIT
(1)  020640  010502                          MOV     R5,R2           ;;GET THE LSD
(1)  020642  000764                          BR      6$              ;;GO CHANGE TO ASCII
(1)  020644  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  020646  100003                          BPL     9$              ;;BR IF NO
(1)  020650  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
(1)  020656  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
(3)  020660  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  020662  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  020664  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  020666  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  020670  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  020672  104401  020720                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
(1)  020676  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
(1)  020704  012616                          MOV     (SP)+,(SP)
(1)  020706  000002                          RTI                     ;;RETURN TO USER
(1)  020710  023420                  $DTBL:  10000.
(1)  020712  001750                          1000.
(1)  020714  000144                          100.
(1)  020716  000012                          10.
(1)  020720  000004                  $DBLK:  .BLKW   4
2000                                 .SBTTL  SCOPE HANDLER ROUTINE
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
(1)                                  ;*AND LOAD THE TEST NUMBER($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)  020730                          $SCOPE:
(1)  020730  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
(2)  020732  004737  013246                  JSR     PC,ACTRLC
(1)  020736  032777  040000  160174  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(1)  020744  001070                          BNE     $OVER           ;;YES IF SW14=1
(1)                                  ;*****START OF CODE FOR THE XOR TESTER*****
(1)  020746  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
```

```
(1)  020750  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(1)  020754  012737  020774  000004        MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(1)  020762  005737  177060                TST     @#177060        ;;TIME OUT ON XOR?
(1)  020766  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  020772  000446                        BR      $SVLAD          ;;GO TO THE NEXT TEST
(1)  020774  022626                5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(1)  020776  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  021002  000451                        BR      $OVER           ;;LOOP ON THE PRESENT TEST
(1)  021004                        6$:;*****END OF CODE FOR THE XOR TESTER*****
(1)  021004  032777  000400  160126        BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
(1)  021012  001404                        BEQ     2$              ;;BR IF NO
(1)  021014  127737  160120  001102        CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
(1)  021022  001441                        BEQ     $OVER           ;;BR IF YES
(1)  021024  105737  001103        2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(1)  021030  001404                        BEQ     3$              ;;BR IF NO
(1)  021032  105037  001103        4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  021036  005037  001166                CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  021042  032777  004000  160070  3$:   BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
(1)  021050  001011                        BNE     1$              ;;BR IF YES
(1)  021052  005737  001202                TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(1)  021056  001406                        BEQ     1$              ;;      INHIBIT ITERATIONS
(1)  021060  005237  001104                INC     $ICNT           ;;INCREMENT ITERATION COUNT
(1)  021064  023737  001166  001104        CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(1)  021072  002015                        BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
(1)  021074  012737  000001  001104  1$:   MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(1)  021102  013737  021142  001166        MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(1)  021110  105237  001102        $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(1)  021114  113737  001102  001200        MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
(1)  021122  011637  001106                MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  021126  013777  001102  160006  $OVER: MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(1)  021134  013716  001106                MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  021140  000002                        RTI                     ;;FIXES PS
(1)  021142  003720                $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
2001                               .SBTTL  ERROR HANDLER ROUTINE
(1)
(2)                                ;;**********************************************************************
(1)                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.
(1)                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(1)                                ;*AND GO TO $ERRTYP ON ERROR
(1)                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                                ;*SW15=1          HALT ON ERROR
(1)                                ;*SW13=1          INHIBIT ERROR TYPEOUTS
(1)                                ;*CALL
(1)                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
(1)
(1)  021144                        $ERROR:
(1)  021144  104407                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
(2)  021146  004737  013246                JSR     PC,ACTRLC
(1)  021152  105237  001103        7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
(1)  021156  001775                        BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(1)  021160  013777  001102  157754        MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  021166  005237  001112                INC     $ERTTL          ;;INC THE ERROR COUNT
(1)  021172  011637  001116                MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  021176  162737  000002  001116        SUB     #2,$ERRPC
```

```
(1)  021204  117737  157706  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  021212  032777  020000  157720          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
(1)  021220  001004                          BNE     20$             ;;SKIP TYPEOUTS
(1)  021222  004737  021274                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
(1)  021226  104401  001171                  TYPE    ,$CRLF
(1)  021232                          20$:
(1)  021232  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  021240  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
(1)  021242  113737  001114  021254          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
(1)  021250  004737  023254                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
(1)  021254     000                  21$:    .BYTE   0
(1)  021255     000                          .BYTE   0
(1)  021256  000777                  22$:    BR      .22$            ;;APT ERROR LOOP
(1)  021260  005777  157654          2$:     TST     @SWR            ;;HALT ON ERROR
(1)  021264  100002                          BPL     3$              ;;SKIP IF CONTINUE
(1)  021266  000000                          HALT                    ;;HALT ON ERROR!
(1)  021270  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
(1)  021272                          3$:
(1)  021272  000002                          RTI                     ;;RETURN
2002                                 .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
(1)                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
(1)                                  ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
(1)
(1)  021274                          $ERRTYP:
(1)  021274  104401  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021300  010046                          MOV     R0,-(SP)        ;;SAVE R0
(1)  021302  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
(1)  021304  153700  001114                  BISB    @#$ITEMB,R0
(1)  021310  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                  ;;TYPE THE PC OF THE ERROR
(2)  021312  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                  ;;ERROR ADDRESS
(2)  021316  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1   021320  000426                          BR      6$              ;;GET OUT
(1)  021322  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(1)  021324  006300                          ASL     R0              ;;       WORK FOR THE ERROR TABLE
(1)  021326  006300                          ASL     R0
(1)  021330  006300                          ASL     R0
(1)  021332  062700  001256                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
(1)  021336  012037  021346                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
(1)  021342  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(1)  021344  104401                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
(1)  021346  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
(1)  021350  104401  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021354  012037  021364          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
(1)  021360  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
(1)  021362  104401                          TYPE                    ;;TYPE THE "DATA HEADER"
(1)  021364  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
(1)  021366  104401  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021372  011000                  5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
(1)  021374  001004                          BNE     7$              ;;GO TYPE THE DATA
```

```
(1)  021376  012600                    6$:     MOV     (SP)+,R0        ;;RESTORE R0
(1)  021400  104401  001171                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021404  000207                            RTS     PC              ;;RETURN
(2)  021406                            7$:
(2)  021406  013046                            MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
(2)  021410  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  021412  005710                            TST     (R0)            ;;IS THERE ANOTHER NUMBER?
(1)  021414  001770                            BEQ     6$              ;;BR IF NO
(1)  021416  104401  021424                    TYPE    ,8$             ;;TYPE TWO(2) SPACES
(1)  021422  000771                            BR      7$              ;;LOOP
(1)  021424  020040      000          8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
(1)          021430                            .EVEN
2003                                    .SBTTL  POWER DOWN AND UP ROUTINES
(1)
(2)                                     ;;**************************************************************
(1)                                     ;POWER DOWN ROUTINE
(1)  021430  012737  021574  000024    $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
(1)  021436  012737  000340  000026            MOV     #340,@#PWRVEC+2 ;;PRIO:7
(3)  021444  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  021446  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  021450  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  021452  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  021454  010446                            MOV     R4,-(SP)        ;;PUSH R4 ON STACK
(3)  021456  010546                            MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(3)  021460  017746  157454                    MOV     @$WR,-(SP)      ;;PUSH @$WR ON STACK
(1)  021464  010637  021600                    MOV     SP,$SAVR6       ;;SAVE SP
(1)  021470  012737  021502  000024            MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
(1)  021476  000000                            HALT
(1)  021500  000776                            BR      .-2             ;;HANG UP
(1)
(2)                                     ;;**************************************************************
(1)                                     ;POWER UP ROUTINE
(1)  021502  012737  021574  000024    $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(1)  021510  013706  021600                    MOV     $SAVR6,SP       ;;GET SP
(1)  021514  005037  021600                    CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
(1)  021520  005237  021600            1$:     INC     $SAVR6          ;;WAIT FOR THE INC
(1)  021524  001375                            BNE     1$              ;;OF  WORD
(3)  021526  012677  157406                    MOV     (SP)+,@$WR      ;;POP STACK INTO @$WR
(3)  021532  012605                            MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  021534  012604                            MOV     (SP)+,R4        ;;POP STACK INTO R4
(3)  021536  012603                            MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  021540  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  021542  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  021544  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  021546  012737  021430  000024            MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1)  021554  012737  000340  000026            MOV     #340,@#PWRVEC+2 ;;PRIO:7
(1)  021562  104401                            TYPE                    ;;REPORT THE POWER FAILURE
(1)  021564  021602            $PWRMG: .WORD   PWRMSG                  ;;POWER FAIL MESSAGE POINTER
(1)  021566  012716                            MOV     (PC)+,(SP)      ;;RESTART AT BEGIN
(1)  021570  001462            $PWRAD: .WORD   BEGIN                   ;;RESTART ADDRESS
(1)  021572  000002                            RTI
(1)  021574  000000            $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
(1)  021576  000776                            BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
(1)  021600  000000            $SAVR6: 0                               ;;PUT THE SP HERE
```

```
 2004  021602  005015  042522  052123  PWRMSG: .ASCIZ  <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12><12>
       021610  051101  044524  043516
       021616  040440  052106  051105
       021624  040440  050040  053517
       021632  051105  043040  044501
       021640  052514  042522  005015
       021646  000012
 2005                                  .EVEN
 2006
 2007                                  .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)  021650  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
  (1)  021654  116637  000001  022073          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
  (1)  021662  112637  022075          MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
  (1)  021666  062716  000002          ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
  (1)  021672  000406                  BR      $TYPON
  (1)  021674  112737  000001  022073  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
  (1)  021702  112737  000006  022075          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
  (1)  021710  112737  000005  022072  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
  (1)  021716  010346                  MOV     R3,-(SP)        ;;SAVE R3
  (1)  021720  010446                  MOV     R4,-(SP)        ;;SAVE R4
  (1)  021722  010546                  MOV     R5,-(SP)        ;;SAVE R5
  (1)  021724  113704  022075          MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
  (1)  021730  005404                  NEG     R4
  (1)  021732  062704  000006          ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
  (1)  021736  110437  022074          MOVB    R4,$OMODE       ;;SAVE IT FOR USE
  (1)  021742  113704  022073          MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
  (1)  021746  016605  000012          MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
  (1)  021752  005003                  CLR     R3              ;;CLEAR THE OUTPUT WORD
  (1)  021754  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
  (1)  021756  000404                  BR      3$              ;;GO DO MSB
```

```
(1)  021760  006105                2$:   ROL   R5              ;;FORM THIS DIGIT
(1)  021762  006105                      ROL   R5
(1)  021764  006105                      ROL   R5
(1)  021766  010503                      MOV   R5,R3
(1)  021770  006103                3$:   ROL   R3              ;;GET LSB OF THIS DIGIT
(1)  021772  105337  022074              DECB  $OMODE          ;;TYPE THIS DIGIT?
(1)  021776  100016                      BPL   7$              ;;BR IF NO
(1)  022000  042703  177770              BIC   #177770,R3      ;;GET RID OF JUNK
(1)  022004  001002                      BNE   4$              ;;TEST FOR 0
(1)  022006  005704                      TST   R4              ;;SUPPRESS THIS 0?
(1)  022010  001403                      BEQ   5$              ;;BR IF YES
(1)  022012  005204                4$:   INC   R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  022014  052703  000060              BIS   #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  022020  052703  000040        5$:   BIS   #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  022024  110337  022070              MOVB  R3,9$           ;;SAVE FOR TYPING
(1)  022030  104401  022070              TYPE  ,8$             ;;GO TYPE THIS DIGIT
(1)  022034  105337  022072        7$:   DECB  $OCNT           ;;COUNT BY 1
(1)  022040  003347                      BGT   2$              ;;BR IF MORE TO DO
(1)  022042  002402                      BLT   6$              ;;BR IF DONE
(1)  022044  005204                      INC   R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  022046  000744                      BR    2$              ;;GO DO THE LAST DIGIT
(1)  022050  012605                6$:   MOV   (SP)+,R5        ;;RESTORE R5
(1)  022052  012604                      MOV   (SP)+,R4        ;;RESTORE R4
(1)  022054  012603                      MOV   (SP)+,R3        ;;RESTORE R3
(1)  022056  016666  000002  000004      MOV   2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  022064  012616                      MOV   (SP)+,(SP)
(1)  022066  000002                      RTI                   ;;RETURN
(1)  022070     000              9$:     .BYTE 0               ;;STORAGE FOR ASCII DIGIT
(1)  022071     000                      .BYTE 0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  022072     000              $OCNT:  .BYTE 0               ;;OCTAL DIGIT COUNTER
(1)  022073     000              $OFILL: .BYTE 0               ;;ZERO FILL SWITCH
(1)  022074  000000              $OMODE: .WORD 0               ;;NUMBER OF DIGITS TO TYPE
2008                                    .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 CHAPACTER 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 IS FIRST ADDRESS OF AN ASCIZ STRING
(1)                                     ;*OR
(1)                                     ;*      TYPE
(1)                                     ;*      MESADR
(1)                                     ;*
(1)
(1)  022076  105737  001157        $TYPE:  TSTB  $TPFLG          ;;IS THERE A TERMINAL?
(1)  022102  100002                        BPL   1$              ;;BR IF YES
(1)  022104  000000                        HALT                  ;;HALT HERE IF NO TERMINAL
(1)  022106  000430                        BR    3$              ;;LEAVE
(1)  022110  010046                1$:     MOV   R0,-(SP)        ;;SAVE R0
```

```
(1)  022112  017600  000002                MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(1)  022116  122737  000001  001214        CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  022124  001011                        BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
(1)  022126  132737  000100  001215        BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
(1)  022134  001405                        BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(1)  022136  010037  022146                MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(1)  022142  004737  023244                JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(1)  022146  000000                61$:    .WORD   0               ;;MESSAGE ADDRESS
(1)  022150  132737  000040  001215  62$:  BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(1)  022156  001003                        BNE     60$             ;;YES,SKIP TYPE OUT
(1)  022160  112046                2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  022162  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  022164  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  022166  012600                60$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  022170  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  022174  000002                        RTI                     ;;RETURN
(1)  022176  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  022202  001430                        BEQ     8$
(1)  022204  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)  022210  001006                        BNE     5$
(1)  022212  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  022214  104401                        TYPE                    ;;TYPE A CR AND LF
(1)  022216  001171                        $CRLF
(1)  022220  105037  022354                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)  022224  000755                        BR      2$              ;;GET NEXT CHARACTER
(1)  022226  004737  022310        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  022232  123726  001156        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  022236  001350                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  022240  013746  001154                MOV     $NULL,-(SP      ;;GET # OF FILLER CHARS. NEEDED
(1)                                                                ;;AND THE NULL CHAR.
(1)  022244  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  022250  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  022252  004737  022310                JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  022256  105337  022354                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  022262  000770                        BR      7$              ;;LOOP
(1)
(1)                                ;HORIZONTAL TAB PROCESSOR
(1)
(1)  022264  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  022270  004737  022310        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  022274  132737  000007  022354        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  022302  001372                        BNE     9$              ;;TAB STOP
(1)  022304  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
(1)  022306  000724                        BR      2$              ;;GET NEXT CHARACTER
(1)  022310  105777  156634        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  022314  100375                        BPL     $TYPEC
(1)  022316  116677  000002  156626        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  022324  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  022332  001003                        BNE     1$              ;;BRANCH IF NO
(1)  022334  105037  022354                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  022340  000406                        BR      $TYPEX          ;;EXIT
(1)  022342  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  022350  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
(1)  022352  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
```

```
                                $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
                                $TYPEX: RTS     PC

                                .SBTTL  TTY INPUT ROUTINE

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

                                ;;**********************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                ;*WHEN OPERATING IN TTY FLAG MODE.
022360  022737  000176  001140  $CKSWR: CMP     #$WREG,SWR      ;;IS THE SOFT-SWR SELECTED?
022366  001074                          BNE     15$             ;;BRANCH IF NO
022370  105777  156550                  TSTB    @$TKS           ;;CHAR THERE?
022374  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
022376  117746  156544                  MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
022402  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
022406  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
022412  001062                          BNE     15$             ;;NO, RETURN TO USER
022414  123727  001134  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
022422  001456                          BEQ     15$             ;;BRANCH IF YES

022424  104401  023105                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
022430  104401  023112          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
022434  013746  000176                  MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
022440  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
022442  104401  023123                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
022446  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
022450  005046                          CLR     -(SP)           ;;THE NEW SWR
022452  105777  156466          7$:     TSTB    @$TKS           ;;CHAR THERE?
022456  100375                          BPL     7$              ;;IF NOT TRY AGAIN

022460  117746  156462                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
022464  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII


022470  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
022474  001005                          BNE     10$             ;;BRANCH IF NOT
022476  104401  023100                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
022502  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
022506  000757                          BR      19$             ;;LET'S TRY IT AGAIN


022510  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
022514  001022                          BNE     16$             ;;BRANCH IF NO
022516  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
022522  001403                          BEQ     11$             ;;BRANCH IF YES
022524  016677  000002  156406          MOV     2(SP),@SWR      ;;SAVE NEW SWR
022532  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
022536  104401  001171          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
022542  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS
```

```
(1)  022550  001003                          BNE     15$             ;;BRANCH IF NOT
(1)  022552  012777  000100  156364          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1)  022560  000002                  15$:    RTI                     ;;RETURN
(1)  022562  004737  022310          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
(1)  022566  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
(1)  022572  002420                          BLT     18$             ;;BRANCH IF YES
(1)  022574  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
(1)  022600  003015                          BGT     18$             ;;BRANCH IF YES
(1)  022602  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
(1)  022606  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
(1)  022612  001403                          BEQ     17$             ;;BRANCH IF YES
(1)  022614  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
(1)  022616  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
(1)  022620  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
(1)  022622  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
(1)  022626  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
(1)  022632  000707                          BR      7$              ;;GET THE NEXT ONE
(1)  022634  104401  001170          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
(1)  022640  000720                          BR      20$             ;;SIMULATE CONTROL-U
(1)                                  .DSABL  LSB
(1)
(1)
(2)                                  ;;**************************************************************
(1)                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                  ;*CALL:
(1)                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
(1)                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
(1)                                  ;
(1)
(1)  022642  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
(1)  022644  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
(1)  022652  105777  156256          1$:     TSTB    @$TKS           ;;WAIT FOR
(1)  022656  100375                          BPL     1$              ;;A CHARACTER
(1)  022660  117766  156252  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
(1)  022666  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
(1)  022674  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
(1)  022702  001013                          BNE     3$              ;;BRANCH IF NO
(1)  022704  105777  156234          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
(1)  022710  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
(1)  022712  117746  156230                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
(1)  022716  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)  022722  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
(1)  022726  001366                          BNE     2$              ;;IF NOT DISCARD IT
(1)  022730  000750                          BR      1$              ;;YES, RESUME
(1)  022732  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
(1)  022740  002407                          BLT     4$              ;;BRANCH IF YES
(1)  022742  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
(1)  022750  003003                          BGT     4$              ;;BRANCH IF YES
(1)  022752  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
(1)  022760  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)  022762  010346                     $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
(1)  022764  012703  023070             1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
(1)  022770  022703  023100             2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
(1)  022774  101405                             BLOS    4$              ;;BR IF YES
(1)  022776  104410                             RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
(1)  023000  112613                             MOVB    (SP)+,(R3)      ;;GET CHARACTER
(1)  023002  122713  000177             10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
(1)  023006  001003                             BNE     3$              ;;SKIP IF NOT
(1)  023010  104401  001170             4$:     TYPE    ,$QUES          ;;TYPE A '?'
(1)  023014  000763                             BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1)  023016  111337  023066             3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
(1)  023022  104401  023066                     TYPE    ,9$
(1)  023026  122723  000015                     CMPB    #15,(R3)+       ;;CHECK FOR RETURN
(1)  023032  001356                             BNE     2$              ;;LOOP IF NOT RETURN
(1)  023034  105063  177777                     CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
(1)  023040  104401  001172                     TYPE    ,$LF            ;;TYPE A LINE FEED
(1)  023044  012603                             MOV     (SP)+,R3        ;;RESTORE R3
(1)  023046  011646                             MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  023050  016666  000004  000002             MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
(1)  023056  012766  023070  000004             MOV     #$TTYIN,4(SP)
(1)  023064  000002                             RTI                     ;;RETURN
(1)  023066     000             9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  023067     000                     .BYTE   0                       ;;TERMINATOR
(1)  023070  000010             $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
(1)  023100  052536  005015  000  $CNTLU: .ASCIZ  /^U/<15><12>           ;;CONTROL "U"
(1)  023105     136  006507  000012 $CNTLG: .ASCIZ  /^G/<15><12>        ;;CONTROL "G"
(1)  023112  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)  023120  020075     000
(1)  023123     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
(1)  023130  036440  000040

2010                                    .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 $HIOCT
(1)
(1)  023134  011646             $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
(1)  023136  016666  000004  000002     MOV     4(SP),2(SP)     ;;INPUT NUMBER
(3)  023144  010046                     MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  023146  010146                     MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  023150  010246                     MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(1)  023152  104411             1$:     RDLIN                   ;;READ AN ASCIZ LINE
(1)  023154  012600                     MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
(1)  023156  005001                     CLR     R1              ;;CLEAR DATA WORD
(1)  023160  005002                     CLR     R2
(1)  023162  112046             2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
```

```
(1)  023164  001412                          BEQ     3$              ;;IF ZERO GET OUT
(1)  023166  006301                          ASL     R1              ;;*2
(1)  023170  006102                          ROL     R2
(1)  023172  006301                          ASL     R1              ;;*4
(1)  023174  006102                          ROL     R2
(1)  023176  006301                          ASL     R1              ;;*8
(1)  023200  006102                          ROL     R2
(1)  023202  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
(1)  023206  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
(1)  023210  000764                          BR      2$              ;;LOOP
(1)  023212  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
(1)  023214  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
(1)  023220  010237  023234                  MOV     R2,$HIOCT
(3)  023224  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  023226  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  023230  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  023232  000002                          RTI                     ;;RETURN
(1)  023234  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
2011                                 .SBTTL  APT COMMUNICATIONS ROUTINE
(1)
(2)                                  ;;**************************************************************
(1)  023236  112737  000001  023502  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
(1)  023244  112737  000001  023500  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
(1)  023252  000403                          BR      $ATYC
(1)  023254  112737  000001  023502  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
(2)  023262                          $ATYC:
(3)  023262  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  023264  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(1)  023266  105737  023500                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
(1)  023272  001450                          BEQ     5$              ;;IF NOT:  BR
(1)  023274  122737  000001  001214          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
(1)  023302  001031                          BNE     3$              ;;IF NOT:  BR
(1)  023304  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(1)  023312  001425                          BEQ     3$              ;;IF NOT:  BR
(1)  023314  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
(1)  023320  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(1)  023326  005737  001174          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
(1)  023332  001375                          BNE     1$              ;;IF NOT:  WAIT
(1)  023334  010037  001210                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
(1)  023340  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
(1)  023342  001376                          BNE     2$
(1)  023344  163700  001210                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
(1)  023350  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
(1)  023352  010037  001212                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
(1)  023356  012737  000004  001174          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
(1)  023364  000413                          BR      5$
(1)  023366  017637  000004  023412  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
(2)  023374  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
(3)  023402  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
(1)  023406  004737  022076                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
(1)  023412  000000                  4$:     .WORD   0
(1)  023414                          5$:
(1)  023414  105737  023502          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
(1)  023420  001416                          BEQ     12$             ;;IF NO?:  BR
```

```
(1)  023422  005737  001214                    TST     $ENV            ;;RUNNING UNDER APT?
(1)  023426  001413                            BEQ     12$             ;;IF NOT:  BR
(1)  023430  005737  001174          11$:      TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
(1)  023434  001375                            BNE     11$             ;;IF NOT:  WAIT
(1)  023436  017637  000004  001176            MOV     @4(SP),$FATAL   ;;GET ERROR #
(1)  023444  062766  000002  000004            ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(1)  023452  005237  001174                    INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
(1)  023456  105037  023502          12$:      CLRB    $FFLG           ;;CLEAR FATAL FLAG
(1)  023462  105037  023501                    CLRB    $LFLG           ;;CLEAR LOG FLAG
(1)  023466  105037  023500                    CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
(3)  023472  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  023474  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  023476  000207                            RTS     PC              ;;RETURN
(1)  023500     000          $MFLG:  .BYTE   0       ;;MESSG. FLAG
(1)  023501     000          $LFLG:  .BYTE   0       ;;LOG FLAG
(1)  023502     000          $FFLG:  .BYTE   0       ;;FATAL FLAG
(1)          023504                  .EVEN
(1)          000200          APTSIZE=200
(1)          000001          APTENV=001
(1)          000100          APTSPOOL=100
(1)          000040          APTCSUP=040
2012
2013                         .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)  023504  010046          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
(1)  023506  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
(1)  023512  005740                  TST     -(R0)           ;;BACKUP BY 2
(1)  023514  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
(1)  023516  006300                  ASL     R0              ;;POSITION FOR INDEXING
(1)  023520  016000  023540          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
(1)  023524  000200                  RTS     R0              ;;GO TO ROUTINE
(1)
(1)
(1)                          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
(1)
(1)  023526  011646          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
(1)  023530  016666  000004  000002  MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
(1)  023536  000002                  RTI                     ;;RESTORE THE PSW
(1)
(3)                          .SBTTL  TRAP TABLE
(3)
(1)                          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(1)                          ;*BY THE "TRAP" INSTRUCTION.
(1)
(1)                          ;       ROUTINE
(1)                          ;       -------
(1)  023540  023526          $TRPAD: .WORD   $TRAP2
(1)  023542  022076                  $TYPE   ;;CALL=TYPE     TRAP+1(1044C!)  TTY TYPEOUT ROUTINE
```

```
   (3)  023544  021674                    $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
   (3)  023546  021650                    $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
   (3)  023550  021710                    $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL
   (3)  023552  020504                    $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
   (1)
   (3)  023554  022430                    $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
   (1)
   (3)  023556  022360                    $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
   (3)  023560  022642                    $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
   (3)  023562  022762                    $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
   (3)  023564  023134                    $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
 2014
 2015  023566  000074            BUF2:   .BLKW   60.
 2016  023756  000074            BUF3:   .BLKW   60.
 2017  024146  000074            BUFFER: 60.
 2018
 2019  024150  000240                    NOP
 2020          000001                    .END
```

```
A       010524          1051    1125*   1982    1991
ABASE = 170416            16*     30
AC      002116           180*    202
ACDW1 = 000000            30
ACDW2 = 000000            30
ACPUOP= 000000            30
ACTRLC  013246          1242    1546    1581*   1615    1628    1665    1700    1757    2000    2001
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   014526          1310    1320    1331    1342    1353    1362    1371    1396    1407    1829*   1834*   1837*   1838*
                        1839*   1840*   1841*   1842    1852#
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          2008    2011*
APTENV= 000001          2001    2008    2011*
APTSIZ= 000200           136    2011*
APTSPC= 000100          2008    2011*
ASWREG= 000000            30
```

```
ATESTN= 000000            30
AUNIT = 000000            30
AUSWR = 000000            30
AUTCAL  011506           216    1249*   1532
AUTOCL  017325          1253    1930*
AUTOER  017411          1537    1933*
AUTRAP  013064          1259    1534*
AVECT1= 100360            17*     30
AVECT2= 000000            30
B       010531          1126*
BEG     001506           132     134*
BEGIN   001462            23     129*   2003
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*    2000
BIT09 = 001000            14*
BIT1  = 000002            14*     431     432     676    1094
BIT10 = 002000            14*     506     507     623     684    1338    1349    1367    1383    1384    1403    1655    1689
BIT11 = 004000            14*     302     328     355     383     514     515     523     524     533     684    1349    1367
                         1392    2000
BIT12 = 010000            14*     619     624    1338    1367    1392    1418    1656
BIT13 = 020000            14*     809    1304    1438    1444    1454    1460    1485    1493    1503    1511    1554    1562
                         2001
BIT14 = 040000            14*    1438    1454    1485    1503    1558    1562    2000
BIT15 = 100000            14*     252     809    1251
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     849     892     950     987    1091
                         1227    1376    1377    1384
BIT5  = 000040            14*    1653    1659
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    1394    1426
                         1728    1732
BIT8  = 000400            14*    1760
BIT9  = 001000            14*     498     499     523     524     533     684    1367    1403    1424    1425
BPTVEC= 000014            14*
BRLEV1  020422           164*    165*    166     658    1786*   1787*   1973*
BRLEV2  020424           166*    167*    643    1974*
BUFFER  024146          2017*
BUF2    023566          2015*
BUF3    023756          2016*
C       010536          1127*   1980    1989
CALDAC  013370           219    1612*
CALDON  017145          1531    1924*
CHANL   014532          1826*   1854*
```

```
CHAR    010332          1077    1085*   1182    1196
CHAR1   010420          1097*   1112
CHAR2   010426          1099*   1108
CHAR3   010450          1100    1104*   1105
CHAR4   010376          1090    1092*   1093
CHRCOL  010520           535*    536*    537*   1075*   1078*   1120*
CHTIME  014240           827     841     872     929     978    1043    1169    1786*
CH0   = 000000          1266*
CH01    020440          1180    1980*
CH02    020462          1194    1989*
CH1   = 000001          1267*
CH10  = 000010          1274*
CH11  = 000011          1275*
CH12  = 000012          1276*
CH13  = 000013          1277*
CH14  = 000014          1278*
CH15  = 000015          1279*
CH16  = 000016          1280*
CH17  = 000017          1281*
CH2   = 000002          1268*
CH3   = 000003          1269*   1309
CH4   = 000004          1270*   1319
CH40  = 000040          1285*
CH41  = 000041          1286*
CH42  = 000042          1287*
CH43  = 000043          1288*
CH44  = 000044          1289*
CH45  = 000045          1290*
CH46  = 000046          1291*
CH47  = 000047          1292*
CH5   = 000005          1271*
CH50  = 000050          1293*
CH51  = 000051          1294*   1469    1522    1805
CH52  = 000052          1295*   1474    1528
CH53  = 000053          1296*   1330    1341    1352
CH54  = 000054          1297*
CH55  = 000055          1298*   1361    1370
CH56  = 000056          1299*   1395    1406
CH57  = 000057          1300*
CH6   = 000006          1272*
CH7   = 000007          1273*
CKSWR = 104407           193    1547    1614    1627    1652    1672    1699    1756    2000    2001    2013*
CLRVC   013524          1207    1211    1652*
COMPAR  014536          1312    1322    1808    1857*
CONTC   016754           222    1915*
CONVRT  014370          1308    1318    1329    1340    1351    1360    1369    1394    1405    1804    1925*
CPTYPE  014264          1786    1793*
CR    = 000015            14*   2008
CRLF  = 000200            14*   2008
CROSS   014062           808    1303    1438    1444    1454    1460    1485    1493    1503    1511    1741*
CSPACE  013310          1441    1448    1457    1464    1489    1497    1507    1515    1593*   1594    1605
CTRLC   002312           187     221*   1538    1587    1599
D       010543          1128*
DACO    001446           122*    154     276     287*    288     295*    296     303*    304     393*    399     574*    693*
```

```
                        696     738*    781*    820     835     873     930     981*    994     1044    1170    1228    1437*
                       1453*   1468    1552    1617*   1635*   1687    1957
DAC1    001450          123*    277     313*    314     321*    322     329*    330     394*    405     604*    702*    705
                        739*    782*    821     834     874     931     982*    993     1044    1170    1230    1445*   1461*
                       1473    1556    1618*   1636*   1688    1958
DAC2    001452          124*    278     340*    341     348*    349     356*    357     395*    411     712*    715     740*
                        783*    824     839     877     934     983*    991     1044    1170    1232    1484*   1502*   1521
                       1560    1619*   1637*   1684    1959
DAC3    001454          125*    279     368*    369     376*    377     384*    385     396*    417     720*    723     741*
                        784*    825     838     878     935     984*    990     1044    1170    1234    1492*   1494*   1510*
                       1527    1564    1620*   1638*   1685    1960
DDISP = 177570          14*     30      136
DELAY   001442          120*    1569    1573
DFB     020364          38      44      50      56      62      68      75      81      87      93      99      105     111
                        117     1965*
DH1     016571          36      42      48      54      60      85      1908*
DH13    016641          97      1910*
DH14    016673          91      103     109     1911*
DH16    016720          115     1912*
DH6     016625          66      73      79      1909*
DISPLA  001142          30*     136*    2000*   2001*
DISPRE  000174          23*     136
DSWR  = 177570          14*     30      136
DT1     020236          37      86      1956*
DT13    020326          98      1962*
DT14    020340          92      104     110     1963*
DT16    020352          116     1964*
DT2     020250          43      1957*
DT3     020262          49      1958*
DT4     020274          55      1959*
DT5     020306          61      1960*
DT6     020320          67      74      80      1961*
DYNCAL  013434          25      1625*
E       010550         1129*    1985    1994
EMTVEC= 000030          14*     136*
EM1     015750          35      1894*
EM10    016224          78      1901*
EM11    016275          84      1902*
EM12    016324          90      1903*
EM13    016367          96      1904*
EM14    016410          102     1905*
EM15    016457          108     1906*
EM16    016526          114     1907*
EM2     016001          41      1895*
EM3     016030          47      1896*
EM4     016057          53      1897*
EM5     016106          59      1898*
EM6     016135          65      1899*
EM7     016170          72      1900*
ERRVEC= 000004          14*     136*    231*    274*    283*    1259*   1535*   2000*
ETX   = 000003         1948*    1951    1954
EVEP    020426          130*    241     251*    252*    254     778     1962    1975*
EVEP1   020430          131*    806     810     815*    1251*   1976*
F       010555         1130*
```

```
FCHANL  014530          1853#
FILZ    013766          1706#   1724#   1728#   1732#   1736#   1743#
FOUND1  016762           244    1916#
FOUND2  017006           250    1918#
FLLRMP  011372           218    1223#
FUNDAC  014266          1467    1472    1520    1526    1799#
G       010562          1131#
GADCS   014520          1830#   1831#   1832    1849#
GADDBR  014522          1834    1850#
GEN1    010316          1077#   1079
GNS   = ****** U          23    2013
GSTAT   014524          1424#   1425#   1851#
GTSWR = 104406          2013#
H       010567          1132#   1981    1990
HIGH    020406          1967#
HT    = 000011            14#   2008
I       010574          1133#
INCR    020410          1968#
INIT1   002126           138     185#
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       010601          1134#
K       010606          1135#
L       010613          1136#   1986    1995
LDTRAP  001742           141#    185     259    1224    1250    1543    1613    1628
LF    = 000012            14#   2008
LOADVC  013642          1208    1679#
LODPNT  007762          1003    1005    1009    1010    1018#
LOW     020404          1966#
LTEST   002466           260#    788
M       010620          1137#
MANHED  017204          1545    1926#
MANUL   013104           217    1542#
MC      002124           183#    208
MD      002122           182#    206
MESIS   015433           815    1889#
MES3    015336          1671    1887#
MES6    015403           813    1888#
ML      002120           181#    204
MTEST   002324           189     214     229#    791
N       010625          1060    1138#   1983    1984    1992    1993
NSUPLY  011636          1313    1317#
N0      010726          1069    1151#
N1      010733          1152#   1988
N2      010740          1153#   1997
N3      010745          1154#
N4      010752          1155#
N5      010757          1156#
N5000C  020207          1436    1483    1949#
N6      010764          1157#
N7      010771          1158#
N8      010776          1159#
N9      011003          1160#
```

```
.          010632  1139#
.OFFIN     020432   195*  196*  198   200   202   204   206   208   1977#
.B         010637  1140#
.PBB       007016   828   842   846#
.PC    =%000007    14#   168*  185*  258*  791*  827*  828*  829*  841*  842*  843*  863*  872*
                   923*  929*  971*  978*  1014* 1043* 1053* 1062* 1071* 1072* 1077* 1080* 1114*
                   1169* 1182* 1196* 1200* 1207* 1208* 1209* 1211* 1224* 1229* 1231* 1233* 1235*
                   1242* 1243* 1250* 1312* 1322* 1441* 1448* 1457* 1464* 1489* 1497* 1507* 1515*
                   1543* 1546* 1588* 1602* 1613* 1615* 1626* 1628* 1630* 1632* 1642* 1665* 1676*
                   1700* 1703* 1757* 1765* 1791* 1808* 1869* 1873* 2000* 2001* 2002* 2003  2008*
                   2009* 2011*
PIC0       006664   811   820#  1213  1777
PIC1       006740   834#  1778
PIC12      011334  1202   1206#  1784
PIC3       007116   845   870#  1779
PIC4       007346   927#  1780
PIC4B      007364   930#
PIC5       007554   977#  1781
PIC6       010012  1016   1042#  1782
PIC6A      010066  1045#  1073
PIC6B      010304  1053   1062   1071   1075#
PIC7       011030  1074   1168#  1783
PIC7A      011172  1181#  1185
PIC7AA     011110  1172#  1201
PIC7B      011300  1195#  1199
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     011574  1261   1262  1307#
PSW    = 177776     14#   640*  643*  655*  658*  667*
PWRMSG     021602  2003   2004#
PWRVEC= 000024      14#   136*  2003#
P3         007174   881#  924
P3A        007244   891   893#  898
P3B        007264   901#  906
P3C        007304   909#  914
P3D        007324   917#  922
P4         007424   938#  972
P4A        007474   949   951#  957
P4B        007530   964#  970
P49976     020222  1452   1501  1952#
P7CNT      020434  1179#  1184#  1193#  1198#  1978#
P7PNT      020436  1180#  1181   1183#  1194#  1195   1197#  1979#
.          010644  1141#
.SMARK     016746   211   1604  1914#
.          010651  1142#
```

```
RDCHR = 104410          2009    2013#
RDLIN = 104411           194    2010    2013#
RDOCT = 104412          2013#
RESVEC= 000010            14#
RCVPT  014220           1772    1777#
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*    820*    824*    834*
                         838*    856*    857*    861     865*    889*    897*    900*    905*    908*    913*    916*    921*
                         947*    956*    963*    969*    990*    993*    997     998*    1012    1023*   1096*   1111*   1228*
                        1230*   1232*   1234*   1238*   1239*   1240    1548*   1549    1550*   1554    1558    1562    1616*
                        1617    1618    1619    1620    1629*   1631*   1635    1636    1637    1638    1639*   1640*   1657*
                        1666*   1681*   1720*   1721    1744*   1748*   1771    1857    1859*   1861*   1863*   1864    1867*
                        1872*   1999*   2002*   2003*   2008*   2010*   2011*   2013*
R1    =%000001            14#    142*    143*    144*    145     155*    156     157     158     821*    825*    835*    839*
                         855*    864*    873*    877*    883*    886*    930*    934*    941*    944*    991*    994*    998
                         999*   1019*   1097*   1107*   1549*   1568    1558*   1663*   1684*   1687*   1721*   1723*   1724
                        1725*   1726*   1729*   1730*   1735*   1737*   1745*   1746*   1858    1860*   1861    1866*   1871*
                        1999*   2003*   2010*   2011*
R2    =%000002            14#    141*    143     145*    146     147     852*    957     874*    878*    884*    887*    931*
                         935*    940*    943*   1018*   1024*   1051*   1060*   1069*   1181*   1195*   1552*   1556*   1560*
                        1564*   1566*   1568*   1572*   1685*   1688*   1999*   2003*   2010*
R3    =%000003            14#    851*    861     879*    894*    902*    910*    918*    936*    951*    964*    997*    999
                        1000*   1020*   1021    1098*   1099*   1569*   1570*   1573*   1574*   1999*   2003*   2007*   2009*
R37     017763          1456    1943#
R38     017537          1440    1939#
R39     020030          1463    1944#
R4    =%000004            14#    850*    858*    859     880*    893     901     909     917     937*    954     955     967
                         968    1095*   1106    1110    1113    2003*   2007*
R40     017604          1447    1940#
R41     020075          1506    1945#
R42     017651          1488    1941#
R43     020142          1514    1946#
R44     017716          1496    1942#
R5    =%000005            14#    808*   1003*   1005*   1008*   1010*   1019    1026    1027*   1076*   1103*   1104    1171*
                        1303*   1308*   1318*   1329*   1340*   1351*   1360*   1369*   1394*   1405*   1435*   1438*   1444*
                        1451*   1454*   1460*   1467*   1472*   1482*   1485*   1493*   1500*   1503*   1511*   1520*   1526*
                        1720    1739*   1741    1750*   1799    1800    1804*   1815*   1825    1843*   1999*   2003*   2007*
R6    =%000006            14#    136*
R7    =%000007            14#
S       010656          1143#
SELD01  017033          1255    1920#
SELD23  017100          1480    1922#
SNDV.   013770          1435    1451    1482    1500    1720#
SP    =%000006            14#    136*    192*    195     236     245*    281     650     671     791*    805*   1223*   1249*
                        1534    1542*   1612*   1625*   1774    1857*   1858*   1866    1867    1868*   1871    1872    1999*
                        2000*   2001    2002*   2003*   2007*   2008*   2009*   2010*   2011*   2013*
SPACEA  011010          1161#   1987    1996
SPREAD  014534          1311#   1321*   1807*   1855#   1864
STACK = 001100            14#    136     192     805    1223    1249    1542    1612    1625
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*    549     553     559
                         565*    566     575     582     587     596*    597     605     613     615     623*    624*    625
                         629     644*    647*    659*    663*    668*    676*    679     694*    697     745*    746*    748
```

```
                    754    762*   764*   765    770    780*   846*   849*   850    870*   879    892*   927*
                    936    950*   980*   987*  1000   1052*  1061*  1070*  1076   1091*  1092   1171   1172*
                   1186*  1227*  1327*  1337*  1338*  1349*  1358*  1367*  1376*  1378*  1379   1383*  1385
                   1389*  1392*  1403*  1412*  1416*  1418*  1419   1427   1566   1655*  1656*  1661   1679*
                   1681   1956   1961   1962   1963
STKLMT= 177774       14*
STX   = 000001     1947*  1949   1952
SWR     001140       30*   136*   619    692    701    711    719    738    739    740    741    743    760
                     822    836    847    853    875    881    890    932    938    948    985    988    995
                    1001   1044   1047   1056   1065   1089   1170   1175   1189   1225   1548   1618   1629
                    1653   1659   1668   1673   1682   1758   2000   2001   2003*  2009*
SWREG   000176       23*   136   2009
SW0   = 000001       14*
SW00  = 000001       14*
SW01  = 000002       14*
SW02  = 000004       14*
SW03  = 000010       14*    822    836    875    932    988   1044   1170   1682
SW04  = 000020       14*
SW05  = 000040       14*
SW06  = 000100       14*
SW07  = 000200       14*    995
SW08  = 000400       14*
SW09  = 001000       14*    692    701    711    719    738    739    740    741    853    981    939   1031
                    1047   1056   1065   1175   1189
SW1   = 000002       14*
SW10  = 002000       14*    743    760    847    890    948    985   1089   1225
SW11  = 004000       14*
SW12  = 010000       14*   1668
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       010663     1144*
TBITVE= 000014       14*
TEMP    020416      129*   133*   186    622*   631*  1971*
TEMP1   020420     1680*  1690   1972*
TICKS   020414      826*   840*   871*   928*   977*  1042*  1168*  1206*  1262*  1789*  1970*
TIMER   014124      829    843    923    971   1014   1072   1200   1259   1756*
TIMERA  014140     1760*
TIMER1  014162     1763   1765*
TIMER2  014164     1761   1770*
TIMER4  014214     1775*
TIMSV   020412     1758*  1760   1770*  1771*  1772*  1773*  1775   1969*
TTYIE   014606      191   1878*
TV.EC   = 000080     14*
TV.ES   = 000004     14*
TVEC.E= 000004       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#
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   011572          1306#
TST56   011634          1316#
TST57   011676          1323    1326#
TST6    002746           316     319#
TST60   011742          1333    1336#
TST61   012012          1344    1348#
TST62   012056          1355    1357#
TST63   012120          1364    1366#
TST64   012164          1373    1375#
TST65   012266          1391#
TST66   012332          1398    1402#
```

```
TST67   012402          1415#
TST7    003004           324    327#
TST70   012504          1429    1433#
TST71   012664          1477#
TYPDS = 104405           791    2013#
TYPE  = 104401           190     210     221     243     249     791     812     814    1252    1254    1439    1446    1455
                        1462    1479    1487    1495    1505    1513    1530    1536    1544    1603    1670    1999    2031
                        2002    2003    2007    2008    2009    2013#
TYPOC = 104402          2002    2009    2013#
TYPON = 104404          2013#
TYPOS = 104403           246    2013#
U       010670          1145#
V       010675          1146#
VADDR   001436           118#    233     780     781     782     783     794
VBUTMP  010510          1044*   1087*   1101*   1116*   1170*
VECTMP  010512          1044*   1088*   1102*   1117*   1170*
VEKJAL  006622           215    805#
VVECT   001440           119#    785     786
V1754   014514          1307    1847#
V24     014516          1311    1317    1321    1848#
W       010702          1147#
WAITIN  002152           192#    212     224
X       010707          1148#
XDACUT  014366          1799*   1801*   1812*   1821#   1964
XPOS    010516          1045*   1049*   1054*   1058*   1063*   1067*   1087    1101    1110*   1113*   1119#   1173*   1177*
                        1187*   1191*
Y       010714          1149#
YPOS    010514          1046*   1050*   1055*   1059*   1064*   1068*   1085    1089    1102    1106*   1109*   1118#   1174*
                        1178*   1188*   1192*
YPT     010522          1085*   1109    1121#
Z       010721          1150#
$APTHD  001000            29#
$ASTAT= ****** L        2011
$ATYC   023262          2011#
$ATY1   023236          2011#
$ATY3   023244          2008    2011#
$ATY4   023254          2001    2011#
$AJTOB  001134            30#   2009
$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*   1310*   1320*   1331*   1332    1342*   1343    1353*   1354    1362*
                        1363    1371*   1372    1379*   1380    1385*   1386    1396*   1397    1407*   1408    1419*   1420
                        1427*   1428    1842*   1860    1956    1957    1958    1959    1960    1963    1964
$CCW1   001254            30#
$CHARC  022354          2008#*
$CKSWR  022360          2009#   2013
$CMTAG  001100            30#    136
```

```
$CM1   = 000002          30#
$CM2   = 000004          30#
$CM3   = 000002          30#
$CNTLG  023105         2009#
$CNTLU  023100         2009#
$CPUOP  001222           30#
$CRLF   001171           30#  2001    2002    2008    2009
$DBLK   020720         1999#
$DEVCT  001204           30#
$DEVM   001252           30#
$DOAGN  006576          791#
$DTBL   020710         1999#
$ENDAD  006566           27    791#
$ENDCT  006534          791#
$ENDMG  006605          791#
$ENULL  006602          791#
$ENV    001214           30#  2001    2008    2011
$ENVM   001215           30#   136    2008    2011
$EOP    006500          779    791#
$EOPCT  006526          791#
$ERFLG  001103           30#  2000*   2001*
$ERMAX  001115           30#  2000
$ERROR  021144          136   2001#
$ERRPC  001116           30#  1956    1957    1958    1959    1960    1961    1962    1963    1964    2001#   2002
$ERRTB  001256           30#  2002
$ERRTY  021274         2001   2002#
$ERTTL  001112           30#  2001#
$ETABL  001214           30#
$ETEND  001256           29     30#
$FATAL  001176           30#  2011#
$FFLG   023502         2011#*
$FILLC  001156           30#  2008
$FILLS  001155           30#  2008
$GDADR  001120           30#
$GDDAT  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     393#    384
                        386    389#    398#    400     404#    406     410#    412     416#    418     423#    426     432#
                        434    438#    440     446#    448     452#    454     462#    464     464#    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     695#    689
                        694#   697     703#    706     713#    721#    738#    739#    740#    741#    753#    763#   1307#
                       1317#  1328#   1332    1339#   1343    1350#   1354    1359#   1363    1369#   1372    1377#   1380
                       1384#  1386    1393#   1397    1404#   1408    1417#   1420    1426#   1428    1802#   1813#   1859
                       1956   1957    1958    1959    1960    1963    1964
$GET42  006556          791#
$GTSWR  022430         2009#   2013
$HD    = 000000           13
$HIBTS  001000           29#
$HIOCT  023234         2010#*
$ICNT   001104           30#  2000*
$ILLUP  021574         2003#
$INTAG  001135           30#  2009
$ITEMB  001114           30#  2001#   2002
```

```
$LF     001172            30*   2001    2008    2009
$LFLG   023501          2011**
$LPADR  001106            30*    136*    273*   1262*   2000*
$LPERR  001110            30*   1251*
$MADR1  001226            30*
$MADR2  001232            30*
$MADR3  001236            30*
$MADR4  001242            30*
$MAIL   001174            29      30*     136    2000    2001    2008
$MAMS1  001224            30*
$MAMS2  001230            30*
$MAMS3  001234            30*
$MAMS4  001240            30*
$MBADR  001002            29*
$MFLG   023500          2011**
$MNEW   023123          2009*
$MSGAD  001210            30*   2011*
$MSGLG  001212            30*   2011*
$MSGTY  001174            30*   2011*
$MSWR   023112          2009*
$MTYP1  001225            30*
$MTYP2  001231            30*
$MTYP3  001235            30*
$MTYP4  001241            30*
$MXCNT  021142          2000*
$NULL   001154            30*   2008
$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*   739*   739*
                         740*   741*   742*   759*   775*  1306*  1316*  1326*  1336*  1348*  1357*  1366*  1375*
                        1391*  1402*  1415*  1433*  1477*
$OCNT   022072          2007**
$OMODE  022074          2007**
$OVER   021126          2000*
$PASS   001202            30*    136*    223*    791*   2000
$PASTM  001006            29*
$PWRAD  021570          2003*
$PWRDN  021430           136   2003*
$PWRMG  021564          2003*
$PWRUP  021502          2003*
$QUES   001170            30*   2001    2008    2009
$RDCHR  022642          2009*   2013
$RDDEC= ****** U        2013
$RDLIN  022762          2009*   2013
$RDOCT  023134          2010*   2013
$RDSZ = 000010          2009*
$REGAD  001160            30*
$REGO   001162            30*
$REG1   001164            30*
$RTNAD  006600            791*
$R2A  = ****** U        2013
$SAVRE= ****** U        2013
$SAVR6  021600          2003**
$SCOPE  020730           136   2000*
```

```
$SETUP= 000117          32#   135#   136    791   2000   2001   2009
$STUP  = 177777          32#   135#
$SVLAD  021110         2000#
$SVPC  = 000214          27#
$SWR   = 164400           5#    13     21     30    136    272    285    293    301    311    319    327    328
                        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   1306   1316
                       1326   1336   1348   1357   1366   1375   1391   1402   1415   1433   1477   2000   2001
                       2003
$SWREG  001216          30#   136
$SWRMK= 000000          21   2000
$TESTN  001200          30#  2000*
$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#  1433#  1477#
                      2000#
$TKB    001146          30#  1583   1595   2009
$TKS    001144          30#  1581   1593   2009#
$TN    = 000072           6#    13    240    272#   285#   290    293#   298    301#   311#   316    319#   324
                       327#   338#   343    346#   351    354#   366#   371    374#   379    382#   392#   419
                       422#   427    430#   441    444#   455    460#   472    475#   486    489#   494    497#
                       502    505#   510    513#   518    522#   528    532#   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    753#   761
                       775#   1260   1306#  1316#  1323   1326#  1333   1336#  1344   1348#  1355   1357#  1364
                      1366#   1373   1375#  1391#  1398   1402#  1415#  1429   1433#  1477#
$TPB    001152          30#  2008*
$TPFLG  001157          30#  2008
$TPS    001150          30#  2008
$TRAP   023504         136   2013#
$TRAP2  023526        2013#
$TRP   = 000013       2013#
$TRPAD  023540        2013#
$TSTM   001004          29#
$TSTNM  001102          30#   787#   791#  1260#  2000#  2001
$TTYIN  023070        2009#
$TYPBN= ****** U      2013
$TYPDS  020504        1999#  2013
$TYPE   022076        2008#  2011   2013
$TYPEC  022310        2008#  2009
$TYPEX  022356        2008#
$TYPOC  021674        2007#  2013
$TYPON  021710        2007#  2013
$TYPOS  021650        2007#  2013
$UNIT   001206          30#   230#   234#   237    245    251    254    257#   777#   778   1962
$UNITM  001010          29#
$USWR   001220          30#
$VECT1  001244          30#   160    164
$VECT2  001246          30#
$XTSTR  020746        2000#
$$CET4= 000000         791#
$ZFILL  022073        2007#*
$ZCCAT= ****** U      2000   2001
```

```
.      = 024152            23*    27*    28*    29*    30*   136   646   661   791   860   896   904   912
                           920    953    959    966   1165*  1696  1955* 1999* 2000  2001  2002* 2003  2008
                          2009*  2011*  2015*  2016*
.$ASTA= ****** J          2011
.$X    = 001000             29*
```

```
COMMEN    14#
CROSS    264#  1438  1444  1454  1460  1485  1493  1503  1511
CSPACE   261#  1441  1448  1457  1464  1489  1497  1507  1515
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
          1314  1324  1334  1345  1356  1365  1374  1382  1388  1399  1410  1422  1430  1810
ESCAPE    14#
FIXAD1  1030#  1044  1170
GETPRI    14#
GETSWR    14#
LRXX    1936#  1939  1940  1941  1942  1943  1944  1945  1946
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   739   739   740   741   742   759   775
          1306  1316  1326  1336  1348  1357  1366  1375  1391  1402  1415  1433  1477
POP       14#  1999  2003  2010  2011
PUSH      14#  1999  2003  2010  2011
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   619
           639   654   675   683   691   700   710   718   738   739   740   741   742   759   775
           791  1306  1316  1326  1336  1348  1357  1366  1375  1391  1402  1415  1433  1477
SETPRI    14#
SETTRA  2013#
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  1313  1323  1333  1344  1355  1364  1373  1381  1387  1398
          1409  1421  1429  1809  1814
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  1306  1316  1326  1336  1348  1357  1366  1375  1391  1402  1415
          1433  1477  1999  2000  2001  2002  2003  2007  2008  2009  2010  2011  2013
SWRSU     14#   136#
TRMTRP  2013#
TSTSW9   268#   692   701   711   719
TYPBIN    14#
TYPDEC    14#   791
TYPNAM    14#
TYPNUM    14#
TYPOCS    14#
TYPOCT    14#  2002  2009
TYPTXT    14#
```

```
$$CMRE      30*
$$CMTM      30*
$$ESCA      14*
$$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
          1306  1316  1326  1336  1348  1357  1366  1375  1391  1402  1415  1433  1477
$$SET     2013*
$$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  1323  1333  1344  1355  1364  1373  1338  1429
.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#  2011
.$CATC       7#    23
.$CMTA       7#    30
.$ECP        7#   791
.$ERRO       7#  2001
.$ERRT       9#  2002
.$PARM       8*
.$POWE       8*  2003
.$RDOC      10*  2010
.$READ       8*  2009
.$SAVE       9*
.$SCOP       8*  2000
.$SPAC       9*
.$SWDO       8*
.$TRAP       8*  2013
.$TYPD       9*  1999
.$TYPE       7#     8*  2008
.$TYPO       7#  2007
```

	1999	2000	2009												
CMPB	254	778	2000	2001	2008	2009	2011								
COM	1723														
DEC	555	584	612	631	633	645	660	750	767	791	897	905	913	921	956
	969	1024	1184	1198	1570	1574	1640	1663	1666	1726	1730	1737	1745	1748	1762
	1787	1835	2002												
DECB	2007	2008													
EMT	14														
HALT	23	1675	2001	2003	2008										
INC	234	536	777	791	858	894	902	910	918	951	964	1020	1078	1103	1107
	1111	1378	1694	1831	1999	2000	2001	2003	2007	2009	2011				
INCB	746	764	816	2000	2001	2008									
JCT	14														
JMP	23	24	25	138	214	215	216	217	218	219	788	791	1074	1213	1532
	1538	1587	1599	1775											
JSR	185	258	791	808	827	828	829	841	842	843	872	923	929	971	978
	1003	1005	1008	1010	1014	1043	1053	1062	1071	1072	1077	1169	1182	1196	1200
	1207	1208	1209	1211	1224	1229	1231	1233	1235	1242	1250	1303	1308	1312	1318
	1322	1329	1340	1351	1360	1369	1394	1405	1435	1438	1441	1444	1448	1451	1454
	1457	1460	1464	1467	1472	1482	1485	1489	1493	1497	1500	1503	1507	1511	1515
	1520	1526	1543	1546	1613	1615	1626	1628	1630	1632	1665	1700	1757	1804	1808
	2000	2001	2008	2009	2011										
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	296	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	619	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	805	820	821	824	825	826	834	835
	838	839	840	850	851	852	855	871	873	874	877	878	879	880	883
	884	889	900	908	916	928	930	931	934	935	936	937	940	941	947
	961	963	977	990	991	993	994	997	998	999	1000	1018	1019	1042	1044
	1045	1046	1049	1050	1051	1054	1055	1058	1059	1060	1063	1064	1067	1068	1069
	1075	1076	1085	1087	1088	1095	1096	1097	1101	1102	1109	1168	1170	1171	1172
	1173	1174	1177	1178	1179	1180	1181	1186	1187	1188	1191	1192	1193	1194	1195
	1206	1223	1228	1230	1232	1234	1249	1259	1260	1261	1262	1307	1310	1311	1317
	1320	1321	1327	1328	1331	1338	1339	1342	1349	1350	1353	1359	1362	1367	1368
	1371	1376	1377	1379	1384	1385	1392	1393	1396	1403	1404	1407	1418	1419	1426
	1427	1433	1453	1461	1477	1502	1510	1535	1542	1548	1549	1552	1556	1560	1564
	1566	1568	1569	1573	1583	1595	1612	1616	1617	1618	1619	1620	1625	1629	1635
	1636	1637	1638	1639	1655	1657	1680	1681	1684	1685	1687	1688	1689	1690	1691
	1720	1725	1729	1735	1741	1743	1744	1758	1773	1786	1799	1800	1801	1802	1907
	1825	1826	1828	1830	1842	1857	1858	1859	1860	1866	1967	1971	1972	1999	2000
	2001	2002	2003	2007	2008	2009	2010	2011	2013						
MOVB	154	1098	1721	1724	1999	2000	2001	2007	2008	2009	2010	2011	2013		
NEG	1863	1999	2007												
NOP	260	527	791	1044	1170	1244	1245	1816	1817	1818	2019				
RESET	134	424	678	686	695	704	714	722	738	739	740	741	773	775	791

```
            2009    2010    2011    2013
.IFF          14      21      27      29      30     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    1306    1313    1316    1323    1326    1333    1336    1344    1348
            1355    1357    1364    1366    1373    1375    1381    1387    1391    1398    1402    1409    1415    1421    1429
            1433    1477    1809    1814    1999    2000    2001    2002    2003    2007    2008    2009    2010    2011    2013
.IFT        2000    2001    2009    2010
.IFTF       2000    2001    2009    2010
.IIF          13      21      23      30     136     791    2000    2001    2002    2008    2009    2013
.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    1306    1316    1326    1336    1348    1357    1366    1375    1391    1402    1415    1433    1477    1999
            2000    2001    2003    2010    2011
.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     819     933    1039    1220
            1301    1306    1316    1326    1336    1348    1357    1366    1375    1391    1402    1415    1433    1477    1645
            2000    2001    2009    2013
.MACRO        21      30     136     261     264     268     726    1030    1936    2013
.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     817     831    1029    1315
            1263    1306    1316    1326    1336    1348    1357    1366    1375    1391    1402    1415    1433    1477    1644
            2000    2001    2009    2013
.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     818     832     868     925     975
            1040    1166    1204    1216    1217    1218    1219    1221    1248    1306    1316    1326    1336    1348    1357
            1366    1375    1391    1402    1415    1433    1477    1540    1610    1623    1645    1646    1647    1651    1679
            1753    1876    1999    2000    2001    2002    2003    2007    2008    2009    2010    2011    2013
.TITLE        13
.WORD         23      27      29      30     791    2002    2003    2007    2008    2010    2011    2013
```

ERRORS DETECTED: 0

#DSKZ:DZAACB.DSKZ:DZAACB/CRF=DSKZ:DZAACB

RUN-TIME: 88 45 7 SECONDS
CORE USED: 25K

```
                FOR ANY   ...B1    1171  011104  013705   ...B5                                ...B9
      OVER THE PROGRAM    ...C1    1213  011366  000137   ...C5  ESCAPE     14*   1324      ...C9
                2.  THE   ...D1    1257                    ...D5  $$SKIP     14*    240      ...D9
                    ENAB  ...E1    1332  011730  023737   ...E5                351    359      ...E9
                SWITCH    ...F1    1354  012044  023737   ...F5  INC        234    536      ...F9
            ------------  ...G1    1397  012320  023737   ...G5  TST        146    157      ...G9
                2.  SW10    .H1    1421  012436  001401   ...H5                752    759      ...H9
      DISPLAY SQUARE W    ...I1    1438  012526  004537   ...I5                               ...I9
      THIS TEST PROVID    ...J1    1482  012700  004537   ...J5                               ...J9
      WHICH TIME THE P    ...K1    1527  013050  001454   ...K5
  136         INITIALI    ...L1    1549  013132  010001   ...L5
  794         END OF P    ...M1    1587  013300  000137   ...M5
 2009         TTY INPU    ...N1    1618  013416  010077   ...N5

  (1)                     ...B2    1660  013566  001424   ...B6
  (1)           000020    ...C2    1715                    ...C6
  (1)           000060    ...D2    1762  014150  005337   ...D6
  (1)                     ...E2    1804  014312  004537   ...E6
  (1)           000214    ...F2    1832  014432  105777   ...F6
  (1)  001102      000    ...G2          014660  041501   ...G6
  (2)  001220   000000    ...H2    1887  015336  005015   ...H6
  (1)                     ...I2    1895  016001     104   ...I6
   80  001352   020320    ...J2          016464  020113   ...J6
  134  001506   000005    ...K2    1923  017102  042523   ...K6
  149                     ...L2     (1)  017544  051525   ...L6
  186  002132   005737    ...M2     (1)  020206     000   ...M6
  236  002364   022626    ...N2    1988  020460  010733   ...N6

  291  002600   104002    ...B3     (1)  020626  005720   ...B7
  317  002744   104003    ...C3     (1)  021004  032777   ...C7
  343  003106   001401    ...D3     (1)  021250  004737   ...D7
  370  003244   023737    ...E3     (1)  021422  000771   ...E7
  398  003424   012737    ...F3    2007                    ...F7
  427  003610   001401    ...G3     (1)  022006  005704   ...G7
  465  004026   001401    ...H3     (1)  022156  001003   ...H7
  495  004210   104001    ...I3     (1)                    ...I7
  528  004402   001401    ...J3     (1)  022606  005766   ...J7
  549  004510   105737    ...K3     (1)  023000  112613   ...K7
  598  004730   105737    ...L3     (1)  023210  000764   ...L7
  644  005162   012777    ...M3     (1)  023466  105037   ...M7
  680  005374   023737    ...N3     (3)  023562  022762   ...N7

  714  005636   000005    ...B4  $ADDW1 = 000000           ...B8
  (1)  006110   000005    ...C4  B       010531           ...C8
  764  006332   105277    ...D4  CHO2    020462           ...D8
  (1)  006500             ...E4                           ...E8
  814  006654   104401    ...F4  GEN1    010316           ...F8
                          ...G4                           ...G8
  877  007154   013701    ...H4                           ...H8
  931  007370   013702    ...I4                           ...I8
  983  007602   005077    ...J4  TST17   003554           ...J8
  (1)  010056   013737    ...K4  TYPON = 10-404           ...K8
 1092  010376   105777    ...L4  $DEVM   001252           ...L8
 1129  010550      177    ...M4  $MAMS1  001224           ...M8
 1156  010757      047    ...N4                           ...N8
```