AR11

DIAGNOSTIC TEST II

MD-11-DZARB-B

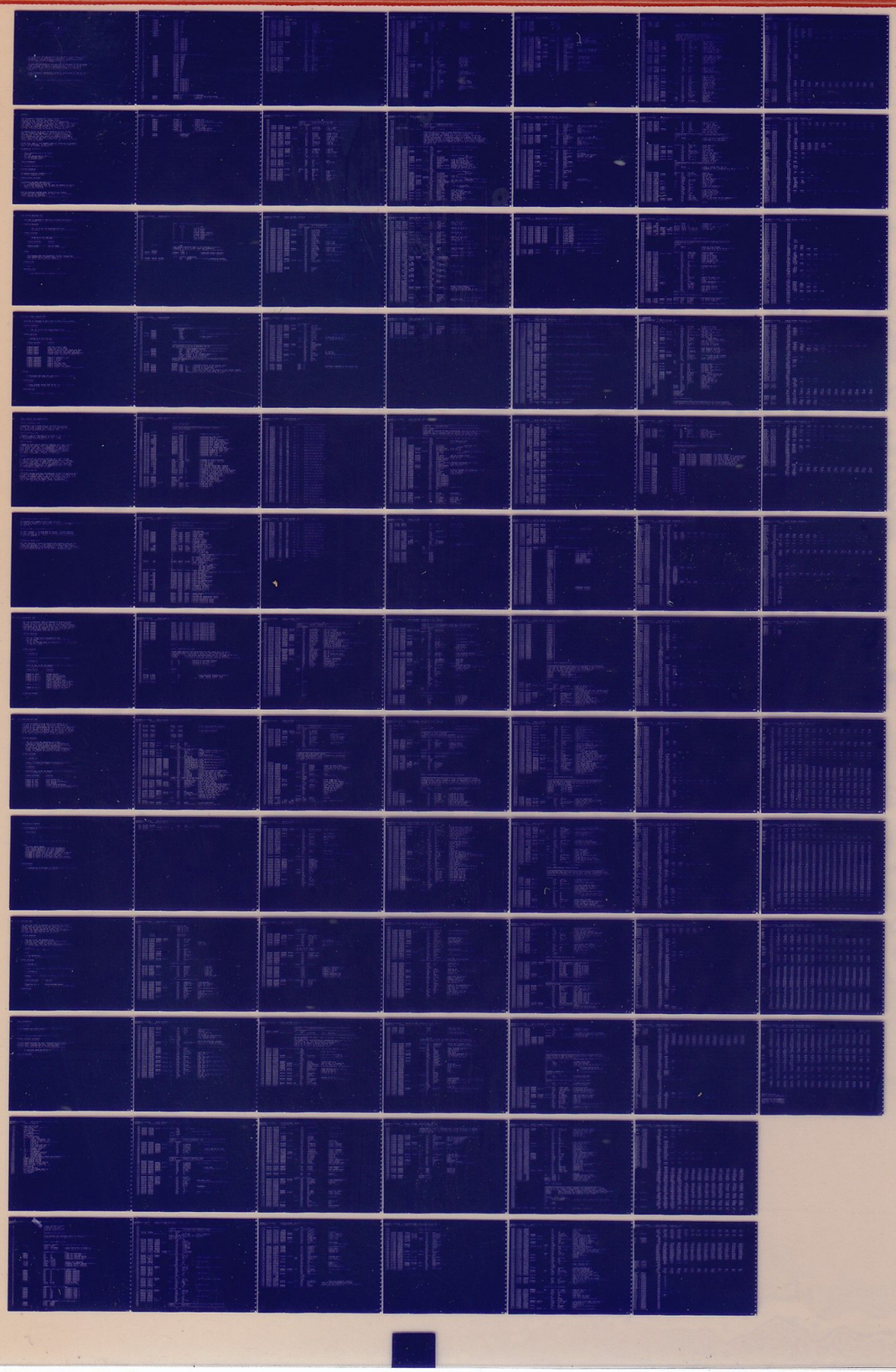
EP-DZARB-B-DL

FICHE 1 OF 1

MAY 1978

digital

MADE IN USA



IDENTIFICATION

PRODUCT CODE: MAINDEC-11-D7ARB-B-D

PRODUCT NAME: AR-11 DIAGNOSTIC TEST II

DATE: MAY 21, 1976

MAINTAINER: DIAGNOSTIC GROUP

FIRST PRINTING, 1974

1. ABSTRACT

THIS DIAGNOSTIC EXERCISES THE "AR-11" ANALOG CIRCUITRY. THE PROGRAM WHEN STARTED WILL TYPE OUT THE PROGRAM TITLE.
A MESSAGE IS THEN PRINTED GIVING THE LETTER DESIGNATORS TO BE TYPED TO RUN ANY ONE OF THE FOUR (4) SEPERATE TESTS OF WHICH THIS PROGRAM IS COMPRISED. THE PROGRAM THEN TYPES A 'CR .' AND THEN WAITS IN A KEYBOARD MONITOR MODE FOR A LETTER TO BE TYPED. ALTHOUGH THESE TESTS MAY BE RUN IN ANY ORDER IT IS IMPERATIVE THAT TEST I IS RUN FIRST AND PROVED 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 '^A' WHILE IN MONITOR MODE WILL ENABLE THE LETTER DESIGNATORS TO BE RETYPED.

UNLIKE AR-11 TEST I, THIS PROGRAM DOES NOT DETERMINE IF ADDITIONAL AR-11'S ARE CONNECTED. TO RUN ANOTHER AR-11, THE OPERATOR MUST SUPPLY THE BUS ADDRESS INDIVIDUALLY.

2. REQUIREMENTS (EQUIPMENT)

A. PDP-11 COMPUTER WITH 8K OF MEMORY
B. TELETYPE
C. AR11 HEX OPTION MODULE INSTALLED
D. VOLTAGE STANDARD (E.D.C.)
E. VR14 OR STORAGE SCOPE

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING BINARY TAPES.

4. STARTING PROCEDURE

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

5. CONSOLE SWITCH SETTINGS

THIS PROGRAM HAS BEEN MODIFIED TO RUN WITH OR WITHOUT A HARDWARE SWITCH REGISTER.
A. ALL SWITCHES SHOULD BE DOWN (0) WHEN THE PROGRAM IS STARTED.
B. REFER TO THE INDIVIDUAL TEST DESCRIPTIONS FOR APPLICABLE CONSOLE SWITCH SETTINGS

WHILE IN KEYBOARD MONITOR MODE, TYPING 'S WILL ENABLE THE SOFTWARE SWITCH REGISTER TO BE LOADED FROM THE TELETYPE WITHOUT HALTING THE PROCESSOR.
* TYPE 'CARRIAGE RETURN' (CR) TO TERMINATE ALL INPUT DATA.

6. QUICK STATIC REGISTER TEST

A. THIS TEST IS DESIGNED TO PROVIDE A REGISTER VERIFICATION TEST IN THIS PROGRAM. IF THIS SUB-TEST FAILS, AR-11 TEST I SHOULD BE LOADED.

B. STARTING SEQUENCE

1. TYPE 'A' TO RUN THE QUICK REGISTER TEST.
2. THE PROGRAM WILL THEN EXECUTE THE QUICK REGISTER TEST.

C. CONTROL SWITCHES

1. TYPING ^C AT ANY TIME WILL ENABLE THE PROGRAM TO EXIT AND RETURN TO THE MONITOR.

CONSOLE SWITCHES	FUNCTION
CONSOLE SW15=1	HALT ON ERROR
CONSOLE SW13=1	INHIBIT ERROR TYPEOUTS

D. ERRORS

THIS PROGRAM USES THE DIAGNOSTIC 'SYSMAC' PACKAGE FOR ERROR REPORTING AND TYPEOUT. REFER TO THE "ERROR POINTER TABLE" FOR TYPE OF LOGIC ERROR AND DESCRIPTION.

E. RESTRICTIONS

NONE

F. EXECUTION TIME

IT TAKES APPROXIMATELY 30 SECONDS TO THIS TEST.

7. POINT PLOT VISUAL DISPLAY TEST

A. THIS TEST IS DESIGNED TO AID IN THE ADJUSTING AND ALIGNMENT OF THE VR14 OR STORAGE SCOPE SCOPE ON THE AR-11 DISPLAY CONTROL.

B. STARTING SEQUENCE

1. TYPE 'B' TO RUN THE VISUAL DISPLAY TEST.
2. THE PROGRAM WILL THEN EXECUTE THE VISUAL DISPLAY TEST.

C. CONTROL SWITCHES

1. TYPING ^C AT ANY TIME WILL ENABLE THE PROGRAM TO EXIT AND RETURN TO THE MONITOR.

CONSOLE SWITCHES	FUNCTION
CONSOLE SW08=0	LOOP THRU DISPLAY TEST
CONSOLE SW08=1	SELECT TEST 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 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

NO PROVISIONS ARE MADE FOR LOGIC ERRORS. THE ONLY ERRORS IN THIS TEST ARE CHECKED VISUALLY.

E. RESTRICTIONS

IF VR14, CHANNEL SWITCH MUST BE SET TO "1 & 2" POSITION.
IF STORAGE SCOPE, POWER MUST BE APPLIED.

F. EXECUTION TIME

IT TAKES APPROXIMATELY 90 SECONDS TO THIS TEST.

8. VISUAL DISPLAY TEST 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 1000. THE POINTS ARE DISPLAYED USING THE DISPLAY INTERRUPT ENABLED.

DISPLAY VERTICAL LINE

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

DISPLAY SQUARE

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

DISPLAY X

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

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

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

DISPLAY CHANNEL 1 AND CHANNEL 2 <VR14>

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

PHOSPHOR AND ERASE TEST

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

9. A TO D CALIBRATION TEST

A. THE 'A/D CALIBRATION' TEST IS DESIGNED TO ACCEPT AN INPUT FROM THE TELETYPE TO INDICATE THE TYPE OF SYNC (EXTERNAL, INTERNAL OR AR-11 CLOCK TO BE USED AND THEN TAKES CONTINUOUS CONVERSIONS USING THE 'CH.' SELECTED VIA THE CONSOLE SWITCHES. THESE SETTINGS MAY BE CHANGED AT ANY TIME. THIS CAN ALSO BE USED FOR FINDING 50-50 POINT AND FOR FINDING MIDDLE OF A STATE FOR SUBSEQUENT REPEATABILITY TEST.

B. STARTING SEQUENCE

1. TYPE 'C' TO RUN THE A/D CALIBRATION TEST.
2. THE TEST HEADER PLUS A REQUEST FOR A SYNC TYPE WILL THEN BE TYPED.
3. TYPE IN THE DESIRED SYNC, 'I' FOR INTERNAL, 'E' FOR EXTERNAL 'C' FOR AR-11 CLOCK FOLLOWED BY 'CR'.
4. THE TEST WILL START.

C. CONTROL SWITCHES

1. ^A (CONTROL A)

TYPING ^A WILL ENABLE A NEW SYNC TYPE TO BE ENTERED.

2. ^C (CONTROL C)

TYPING ^C WILL CAUSE THE PROGRAM TO EXIT THE CALIBRATION TEST AND RETURN TO THE MONITOR.

3.

CONSOLE SWITCH	FUNCTION
CONSOLE SW '0-3'	CHANNEL SELECT
CONSOLE SW 05 = 0	SELECT BIPOLAR CHANNEL
CONSOLE SW 05 = 1	SELECT UNIPOLAR CHANNEL
CONSOLE SW 06 = 0	CONTINUOUS SAMPLES
CONSOLE SW 06 = 1	FREEZE ON CURRENT DATA
CONSOLE SW 07 = 0	DISPLAY DATA ON X AXIS TO CHECK FOR AD INTERACTION WITH X DA
CONSOLE SW 07 = 1	DISPLAY DATA ON Y AXIS TO CHECK FOR AD INTERACTION WITH Y DA
CONSOLE SW 10 = 0	DISPLAY VALUE ON THE SCREEN
CONSOLE SW 10 = 1	PRINT CONVERSION VALUE

D. CALIBRATION ERRORS

NONE

10. A TO D REPEATABILITY TEST

A. THIS TEST REQUESTS A CH.(S) AND A COUNT SPREAD OF '1-4' AND MODE OF OPERATION TO BE TYPED IN BY THE OPERATOR. A SERIES OF'512' CONVERSIONS ARE THEN TAKEN ON THE INPUT CH.(S) CONVERSIONS ARE THEN AVERAGED OUT AND IF THE COUNT SPREAD IS FOUND TO BE GREATER THAN REQUEST, THE RESULTS OF THE CONVERSIONS ARE TYPED OUT. A SINGLE CHANNEL OR A SERIES OF CHANNELS MAY BE TESTED VIA TYPING EITHER 'N(CR)' TO SELECT A SINGLE CHANNEL OR 'N,N(CR)' TO TEST A SERIES OF CHANNELS.

B. STARTING SEQUENCE

1. TYPE 'D' TO RUN THE 'REPEATABILITY' TEST.
2. A REQUEST IS THEN MADE FOR CH.(S) TO BE TESTED AND COUNT SPREAD (RANGE IN WHICH ALL 512 COUNTS MUST FALL FOR THE CH. TO BE CONSIDERED ACCEPTABLE).
3. IF THE CHANNEL IS FOUND TO BE WITHIN THE SELECTED COUNT SPREAD, THE PROGRAM WILL EITHER CONTINUE TO THE NEXT CHANNEL IF SELECTED OR RETEST THE CURRENT CHANNEL.

C. CONTROL SWITCHES

1. ^A (CONTROL A)

TYPING A ^A WHILE THE PROGRAM IS RUNNING WILL ENABLE A NEW CH.(S) AND COUNT SPREAD TO BE SELECTED.

2. ^C (CONTROL)

TYPING ^C WILL CAUSE THE PROGRAM TO EXIT THE 'REPEATABILITY' TEST AND RETURN TO THE MONITOR.

3.

CONSOLE SWITCHES	FUNCTION
CONSOLE SW 10=0	PRINT ERRORS ONLY
CONSOLE SW 10=1	PRINT OUT ALL CONVERSIONS
CONSOLE SW 13=0	PRINT ERRORS
CONSOLE SW 13=1	INHIBIT ERROR PRINTOUTS
CONSOLE SW 15=0	DO NOT HALT UPON REPEATABILITY REPORT
CONSOLE SW 15=1	HALT UPON REPEATABILITY REPORT

D. REPEATABILITY ERRORS

ON ENCOUNTERING AN ERROR (CONSOLE SWITCHES DOWN) THE ERROR DATA IS TYPED OUT.

1. ERROR FORMAT

CH.	HI	AV	LO
A	B	C	D

LO	-5	-4	-3	-2	-1	0	+1	+2	+3	+4	+5	HI
E	F	G	H	I	J	K	L	M	N	O	P	Q

WHERE:

A=CHANNEL BEING TESTED
B=THE HIGHEST READING OF THE '512' CONVERSIONS
C=THE AVERAGE READING OF THE '512' CONVERSIONS
D=THE LOWEST READING OF THE '512' CONVERSIONS
E=NUMBER OF COUNTS 'OUT OF RANGE' LOWER THAN 5 COUNTS
F-J=NUMBER OF COUNTS IN EACH PART LOWER THAN AVERAGE.
K=NUMBER OF COUNTS AT AVERAGE OF THE '512'
L-P=NUMBER OF COUNTS IN EACH PART HIGHER THAN AVERAGE.
Q=NUMBER OF COUNTS 'OUT OF RANGE' HIGHER THAN 5 COUNTS

E. RESTRICTIONS

1. IF A SECOND CH. IS ENTERED, IT MUST BE LARGER THAN THE FIRST CH. OR THE INPUT WILL NOT BE ACCEPTED.

11. A TO D RECOVERY TEST

A. THE "RECOVERY TEST" IS DESIGNED TO DETERMINE THE INTER-CHANNEL SETTLING CAPABILITY OF THE 'AR-11 A TO D'. THE TEST REQUESTS FOR TWO (2) CH. INPUTS TO BE TYPED IN.
THE TEST THEN TAKES A SERIES OF SIXTEEN (16) CONVERSIONS (8 ON EACH CH.) AND THEN TYPES OUT THE '8' AVERAGE VALUES IN THE ORDER THEY WERE TAKEN ON THE SECOND CH.

B. STARTING SEQUENCE

1. TYPE 'E' TO RUN THE RECOVERY TEST.
2. A REQUEST IS THEN MADE FOR THE CH.S TO BE TESTED.
3. TYPE 'N,N (CR)' WHERE 'N' IS ANY CH.
4. THE PROGRAM WILL THEN TAKE CONTINUOUS CONVERSIONS TYPING OUT THE CONVERSION VALUES FOR THE SECOND CH.

EXAMPLE:
CH. A XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

WHERE:

A = THE SECOND CH.
X = THE '8' CONVERSIONS TAKEN ON THAT CH.

C. CONTROL SWITCHES

1. ^A (CONTROL A)

TYPING A '^A' WILL ENABLE A NEW SET OF CH.S TO BE ENTERED.

2. ^C (CONTROL C)

TYPING A '^C' WILL ENABLE THE PROGRAM TO RETURN TO THE MONITOR.

3.

CONSOLE SWITCHES	FUNCTION
CONSOLE SW 10 = 0	PRINT RECOVERY REPORT
CONSOLE SW 10 = 1	INHIBIT PRINTING OF RECOVERY REPORT

D. RESTRICTIONS

NONE

12. MISC. INFORMATION

ACT-11 OR XXDP

IF THE PROGRAM WAS LOADED FROM ACT-11 OR CHAINED FROM XXDP, THE VISUAL DISPLAY TEST WILL BE RUN.

APT

THE APT HOOKS HAVE BEEN INSTALLED BUT NOT TESTED.

13. PROGRAM VARIABLE LOCATIONS

LOCATION $BASE CONTAINS THE AR-11 STARTING DEVICE ADDRESS <170400>
LOCATION $VECT1 CONTAINS THE AR-11 STARTING VECTOR <340>
LOCATION $NULL CONTAINS THE TTY FILLER CHARACTER
LOCATION $FILLS CONTAINS THE TTY FILLER CHARACTER COUNT

NOTE: IF LOCATIONS $BASE OR $VECT1 ARE CHANGED, THE TEST MUST BE RE-INITILIZED AT 200.

14. TABLE OF CONTENTS

ATTACHED

```
                        .TITLE  MAINDEC-11-DZARB-B
                        ;*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-DIQAC-B2),NOV 21, 1975.
                        ;*

                        .SBTTL  BASIC DEFINITIONS

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

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

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

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

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
```

```
                        010000                  SW12=   10000
                        004000                  SW11=   4000
                        002000                  SW10=   2000
                        001000                  SW09=   1000
                        000400                  SW08=   400
                        000200                  SW07=   200
                        000100                  SW06=   100
                        000040                  SW05=   40
                        000020                  SW04=   20
                        000010                  SW03=   10
                        000004                  SW02=   4
                        000002                  SW01=   2
                        000001                  SW00=   1
                                                .EQUIV  SW09,SW9
                                                .EQUIV  SW08,SW8
                                                .EQUIV  SW07,SW7
                                                .EQUIV  SW06,SW6
                                                .EQUIV  SW05,SW5
                                                .EQUIV  SW04,SW4
                                                .EQUIV  SW03,SW3
                                                .EQUIV  SW02,SW2
                                                .EQUIV  SW01,SW1
                                                .EQUIV  SW00,SW0

                                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                        100000                  BIT15=  100000
                        040000                  BIT14=  40000
                        020000                  BIT13=  20000
                        010000                  BIT12=  10000
                        004000                  BIT11=  4000
                        002000                  BIT10=  2000
                        001000                  BIT09=  1000
                        000400                  BIT08=  400
                        000200                  BIT07=  200
                        000100                  BIT06=  100
                        000040                  BIT05=  40
                        000020                  BIT04=  20
                        000010                  BIT03=  10
                        000004                  BIT02=  4
                        000002                  BIT01=  2
                        000001                  BIT00=  1
                                                .EQUIV  BIT09,BIT9
                                                .EQUIV  BIT08,BITA
                                                .EQUIV  BIT07,BIT7
                                                .EQUIV  BIT06,BIT6
                                                .EQUIV  BIT05,BIT5
                                                .EQUIV  BIT04,BIT4
                                                .EQUIV  BIT03,BIT3
                                                .EQUIV  BIT02,BIT2
                                                .EQUIV  BIT01,BIT1
                                                .EQUIV  BIT00,BIT0

                                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                        000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                        000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                        000014                  TBITVEC=14              ;;"T" BIT
```

```
                        TRTVEC= 14              ;;TRACE TRAP
                        BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                        IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                        PWRVEC= 24              ;;POWER FAIL
                        EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                        TRAPVEC=34              ;;"TRAP" TRAP
                        TKVEC=  60              ;;TTY KEYBOARD VECTOR
                        TPVEC=  64              ;;TTY PRINTER VECTOR
                        PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR

                                ABASE=170400
                                AVECT1=340
                                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            BELL ON ERROR
                                ;*         9            LOOP ON ERROR
                                ;*         8            LOOP ON TEST IN SWR<7:0>

                                .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  001356                  JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  001364                  JMP     BEGIN1                  ;JUMP TO RESTART ADDRESS
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
000210                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  004502                          SENDAD                  ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000210                                  .=$SVPC                 ;; RESTORE PC
001000                                  .=1000

                                .SBTTL  APT PARAMETER BLOCK

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

001000                          $APTHD:
001000  000000                  $MIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001200                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000030                  STSTM:  .WORD   30      ;;RUN TIM OF LONGEST TEST
001006  000060                  $PASTM: .WORD   60      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000120                  $UNITM: .WORD   120     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000052                          .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                          SCMTAG:                 ;;START OF COMMON TAGS
001100  000000                          .WORD   0
001102  000                     STSTNM: .BYTE   0       ;;CONTAINS THE TEST NUMBER
001103  000                     SERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
001104  000000                  SICNT:  .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  SLPADR: .WORD   0       ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  SLPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  SERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
001114  000                     SITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
001115  001                     SERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  SERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  SGDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  SBDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  SGDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
001126  000000                  SBDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0       ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134  000000                          .WORD   0
001136  177570                  SWR:    .WORD   DSWR    ;;ADDRESS OF SWITCH REGISTER
001140  177570                  DISPLAY: .WORD  DDISP    ;;ADDRESS OF DISPLAY REGISTER
001142  177560                  STKS:   177560          ;;TTY KBD STATUS
001144  177562                  STKB:   177562          ;;TTY KBD BUFFER
001146  177564                  STPS:   177564          ;;TTY PRINTER STATUS REG. ADDRESS
001150  177566                  STPB:   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
001152     000                  SNULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
001153     002                  SFILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001154     012                  SFILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001155     000                  STPFLG: .BYTE   0       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001156  000000                  SREGAD: .WORD   0       ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  (SREG0) WAS OBTAINED
001160  000000                  SREG0:  .WORD   0       ;;CONTAINS ((SREGAD)+0)
001162  000000                  SREG1:  .WORD   0       ;;CONTAINS ((SREGAD)+2)
001164  000000                  STIMES: 0               ;;MAX. NUMBER OF ITERATIONS
001166  000000                  SESCAPE:0               ;;ESCAPE ON ERROR ADDRESS
001170  177607  000377            SBELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001174     077                  SQUES:  .ASCII  /?/     ;;QUESTION MARK
001175     015                  SCRLF:  .ASCII  <15>    ;;CARRIAGE RETURN
001176  000012                  SLF:    .ASCIZ  <12>    ;;LINE FEED
```

```
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE
                                ;;*****************************************************************
                                .EVEN
001200                          $MAIL:                  ;;APT MAILBOX
001200  000000                  $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
001202  000000                  $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
001204  000000                  $TESTN: .WORD   ATESTN  ;;TEST NUMBER
001206  000000                  $PASS:  .WORD   APASS   ;;PASS COUNT
001210  000000                  $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
001212  000000                  $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
001214  000000                  $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
001216  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
001220                          $ETABLE:                ;;APT ENVIRONMENT TABLE
001220     000                  $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
001221     000                  $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001222  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001224  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001226  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                              11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
001230     000                  $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001231     000                  $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOLAR=002
                                ;*                              500 NSEC MOS=003
001232  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001234     000                  $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001235     000                  $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001236  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001240     000                  $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001241     000                  $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001242  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001244     000                  $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001245     000                  $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001246  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001250     340                  $VECT1: .BYTE   AVECT1  ;;INTERRUPT VECTOR#1
001251     000                  $VECT2: .BYTE   AVECT2  ;;INTERRUPT VECTOR#2
001252     200                  $PRIOR: .BYTE   APRIOR  ;;BUS PRIORITY #1,#2
001253     000                          .BYTE   0      ;;SPARE,NOT USED
                                        .EVEN
001254  170400                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001256  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001260  000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
001262  000000                  $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
001264  000000                  $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
001266  000000                  $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
001270  000000                  $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
```

```
                                SDDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
                                SDDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
                                SDDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
                                SDDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
                                SDDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
                                SDDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
                                SDDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
                                SDDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
                                SDDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
                                SDDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
                                SDDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
                                SDDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
                                SDDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15


001324                          SETEND:


                                .SBTTL  ERROR POINTER TABLE

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

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


001324                          SERRTB:

                                ;ITEM   1

001324  013032                          EM1                     ;DUAL REGISTER ADDRESSING DETECTED
001326  013075                          DH1                     ;ERRPC  BUFADR  EXPECT  READ
001330  013134                          DT1                     ;SERRPC BUFADR $GDDAT  $BDDAT
001332  000000                          0

001334  000000          NBEXT:  0
```

```
                                ADCS:   170400                  ;A TO D STATUS/CONTROL REGISTER
                                ADDBR:  170402                  ;A TO D CONVERTED VALUE <READ>

                                CSR:    170404                  ;CLOCK STATUS REGISTER
                                CSB:    170406                  ;CLOCK PRESET BUFFER

                                VCSTAT: 170410
                                VCXREG: 170412
                                VCYREG: 170414

                                CSC:    170416

001356  005037  013226          BEGIN:  CLR     TEMP
001362  000403                          BR      BEG
001364  012737  000001  013226  BEGIN1: MOV     #1,TEMP
001372  000005                  BEG:    RESET
                                ;;CLEAR THE COMMON TAGS (SCMTAG) AREA
001374  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001400  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001402  022706  001126                  CMP     #$RDDAT,R6      ;;DONE?
001406  001374                          BNE     .-6             ;;LOOP BACK IF NO
001410  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001414  012737  013344  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001422  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001430  012737  013624  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001436  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001444  012737  016134  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001452  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001460  012737  014142  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001466  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001474  005037  001164                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
001500  005037  001166                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
001504  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
001512  012737  001512  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
001520  012737  001520  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001526  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001532  012737  001570  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001540  012737  177570  001136          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001546  012737  177570  001140          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001554  022777  177777  177354          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001562  001013                          BNE     65$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
001564  005737  000001                  TST     @#1             ;;FORCE A TRAP THROUGH ERRVEC
001570  012737  000176  001136  64$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001576  012737  000174  001140          MOV     #DISPREG,DISPLAY ;;POINT TO SOFTWARE DISPLAY REG
001604  012716  001612                  MOV     #65$,(SP)       ;;REPLACE OLD PC WITH NEW
001610  000002                          RTI                     ;;RESTORE PC AND PSW
001612  012637  000004          65$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001616                          $ARG1:
001616  005037  001206                  CLR     $PASS           ;;CLEAR PASS COUNT
001622  132737  000200  001221          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
```

```
   391  001630  001403                          BEQ     64$                ;;YES,USE NON-APT SWITCH
   392  001632  012737  001222  001136          MOV     #$SWREG,SWR        ;;NO,USE APT SWITCH REGISTER
   393  001640                          64$:
   394  001640  005037  177776                  CLR     @#PS
```

```
                                        ;DETERMINE CPU TYPE
                                        ;       R0=3 IF 11/45
                                        ;       R0=2 IF 11/40
                                        ;       R0=1 IF 11/20
                                        ;       R0=0 IF 11/25

            177772                      PIRQ=177772
            177776                      PS=177776

001644  012737  000002  000006        MOV     #RTI,@#6
001652  012737  000006  000004        MOV     #6,@#4
001660  012700  000003                MOV     #3,R0           ;LOAD R0
001664  000261                        SEC                     ;SET C BIT
001666  005737  177772                TST     @#PIRQ          ;TEST PIRQ
001672  005600                        SBC     R0
001674  000261                        SEC
001676  105737  177777                TSTB    @#PS+1
001702  005600                        SBC     R0
001704  005037  177700                CLR     @#177700
001710  005037  000006                CLR     @#6
001714  010037  010750                MOV     R0,CPTYPE
001720  005237  010750                INC     CPTYPE
001724  006300                        ASL     R0
001726  016037  010736  010746        MOV     CPDLAY(R0),CPTIME       ;GET CP DELAY TIME
001734  000137  002042                JMP     INIT1

                                    ;LOAD TRAP CATCHER

001740  012702  000242        LDTRAP: MOV     #242,R2                 ;LOAD R2
001744  012701  000240                MOV     #240,R1                 ;LOAD R1
001750  010221                5$:     MOV     R2,(R1)+                ;LOAD .+2
001752  005021                        CLR     (R1)+                   ;LOAD HALT
001754  010102                        MOV     R1,R2                   ;LOAD R2
001756  005722                        TST     (R2)+                   ;BUMP R2
001760  020227  001002                CMP     R2,#1002                ;TEST FOR LAST
001764  001371                        BNE     5$                      ;BR UNTIL DONE

                                    ;LOAD DEVICE ADDRESSES LOCATIONS

001766  012700  001336                MOV     #ADCS,R0        ;LOAD POINTER
001772  013720  001254        10$:    MOV     $BASE,(R0)+     ;LOAD BASE ADDRESS
001776  022700  001356                CMP     #BEGIN,R0       ;TEST FOR DONE
002002  001373                        BNE     10$             ;BR
002004  013737  001254  010752        MOV     $BASE,ADCS1     ;LOAD HIGH BYTE POINTER
002012  005237  010752                INC     ADCS1           ;
002016  012700  001340                MOV     #ADDBR,R0       ;LOAD 2ND ADDRESS
002022  012701  000002                MOV     #2,R1           ;LOAD R1
002026  060120                12$:    ADD     R1,(R0)+        ;UPDATE REAL DEVICE WORD
002030  005721                        TST     (R1)+           ;BUMP R1
002032  022701  000020                CMP     #20,R1          ;TEST FOR DONE
002036  001373                        BNE     12$             ;BR
002040  000207                        RTS     PC              ;EXIT
```

```
447
448  002042  004737  001740          INIT1:  JSR     PC,LDTRAP
449  002046  005737  013226                  TST     TEMP            ;TEST IF START OR RESTART
450  002052  001011                          BNE     INIT2           ;RESTART
451  002054  005737  000042                  TST     @42             ;TEST IF MONITOR
452  002060  001402                          BEQ     1$              ;BR IF NOT
453  002062  000137  002254                  JMP     VSUAL0          ;RUN SCOPE IF UNDER MONITOR
454  002066  104400                  1$:     TYPE                    ;CALL MESSAGE PRINTER VIA 'EMT'
455  002070  011706                          TITLE                   ;TYPE PROGRAM HEADER.
456  002072  104400                  INITA:  TYPE
457  002074  012070                          MES4                    ;PRINT THE TEST CALL LETTERS.
458  002076  012737  002072  013152  INIT2:  MOV     #INITA,AVECTR   ;SET UP '=A' VECTOR ADDRESS.
459  002104  004737  001740                  JSR     PC,LDTRAP       ;LOAD TRAP CATCHER AND ADDRESSES
460  002110  104400                          TYPE
461  002112  012343                          CNTRLC                  ;PRINT '.' TO INDICATE MONITOR READY
462  002114  005077  177216                  CLR     @ADCS
463  002120  005077  177216                  CLR     @CSR
464  002124  005077  177216                  CLR     @VCSTAT
465  002130  004737  010206                  JSR     PC,XTTYIN       ;WAIT FOR TTY ENTRY
466  002134  122737  000101  010436          CMPB    #'A,INBUF       ;TEST FOR 'A'
467  002142  001002                          BNE     1$              ;NOT 'A
468  002144  000137  007444                  JMP     RESTST          ;YES, RUN QUICK REGISTER TEST
469  002150  122737  000102  010436  1$:     CMPB    #'B,INBUF       ;TEST FOR 'B'
470  002156  001002                          BNE     2$              ;NOT 'B'
471  002160  000137  002236                  JMP     VISUAL          ;YES, RUN 'SCOPE ADJUSTMENT TEST'
472  002164  122737  000103  010436  2$:     CMPB    #'C,INBUF       ;TEST FOR 'C'
473  002172  001002                          BNE     3$              ;NOT 'C'
474  002174  000137  005104                  JMP     CALBRT          ;YES, RUN 'A TO D CALIBRATION' TEST
475  002200  122737  000104  010436  3$:     CMPB    #'D,INBUF       ;TEST FOR 'D'
476  002206  001002                          BNE     4$              ;NOT 'D'
477  002210  000137  006244                  JMP     REPTST          ;YES, RUN 'A TO D REPEATABILITY' TEST
478  002214  122737  000105  010436  4$:     CMPB    #'E,INBUF       ;TEST FOR 'E'
479  002222  001002                          BNE     5$              ;NOT 'E'
480  002224  000137  007024                  JMP     RECVRY          ;YES RUN 'A TO D RECOVERY TEST
481  002230  104400                  5$:     TYPE                    ;ILLEGAL ENTRY
482  002232  012365                          QMARK                   ;TYPE '?'
483  002234  000720                          BR      INIT2           ;WAIT AGAIN
```

```
                                ;;*****************************************************************
                                ;*TEST 1         DISPLAY HORIZONTAL LINE
                                ;;*****************************************************************
002256  000004          TST1:   SCOPE
002260  012737  000001  001164          MOV     #1,STIMES       ;;DO 1 ITERATION
002266  013700  001350  PIC0:   MOV     VCXREG,R0
002272  013701  001352          MOV     VCYREG,R1
002276  012737  000070  013210          MOV     #70,TICKS               ;LOAD TIMER FOR CP TYPE
002304  004737  005050          JSR     PC,CHTIME               ;CHANGE TIMER FOR CP TYPE
002310  004737  002370  1$:     JSR     PC,PBB
002314  004737  004726          JSR     PC,TIMER                ;DONE ?
002320  000773                  BR      1$

                                ;;*****************************************************************
                                ;*TEST 2         DISPLAY A VERTICAL LINE
                                ;;*****************************************************************
002322  000004          TST2:   SCOPE
002324  012737  000001  001164          MOV     #1,STIMES       ;;DO 1 ITERATION
002332  013700  001352  PIC1:   MOV     VCYREG,R0
002336  013701  001350          MOV     VCXREG,R1
002342  012737  000070  013210          MOV     #70,TICKS               ;LOAD TIME
002350  004737  005050          JSR     PC,CHTIME               ;CHANGE TIMER FOR CP TYPE
002354  004737  002370  1$:     JSR     PC,PBB
002360  004737  004726          JSR     PC,TIMER                ;DONE ?
002364  000773                  BR      1$
002366  000427                  BR      PIC3

002370  005077  176752  PBB:    CLR     @VCSTAT
002374  013704  001346          MOV     VCSTAT,R4
002400  012703  001777          MOV     #1777,R3        ;SET HIGH LIMIT
002404  012702  000001          MOV     #1,R2           ;INITIALIZE INCREMENTS BETWEEN POINTS
002410  012711  001000          MOV     #1000,(1)
002414  012710  000000          MOV     #0,(0)
002420  060210  1$:     ADD     R2,(0)          ;INCREMENT
002422  005214          INC     (4)             ;INTENSIFY
002424  105714          TSTB    (4)
002426  100376          BPL     .-2
002430  021003          CMP     (0),R3          ;DONE ALL POINTS?
002432  001372          BNE     1$              ;NO
002434  000207          RTS     PC
```

```
                                ;;*****************************************************************
                                ;*TEST 3         PINCUSHION TEST (DISPLAY SQUARE)
                                ;;*****************************************************************
002436  000004                  TST3:   SCOPE
002440  012737  000001  001164          MOV     #1,STIMES       ;;DO 1 ITERATION
                                ;PLOT A SQUARE FROM LOWER LEFT TO LOWER RIGHT TO
                                ;UPPER RIGHT TO UPPER LEFT TO LOWER LEFT.
                                ;NON STORE DISPLAY
002446  005037  013200          PIC3:   CLR     LOW
002452  005077  176670                  CLR     @VCSTAT
002456  012737  000100  013210          MOV     #100,TICKS
002464  004737  005050                  JSR     PC,CHTIME
002470  013701  001350                  MOV     VCXREG,R1
002474  013702  001352                  MOV     VCYREG,R2
002500  013703  001346                  MOV     VCSTAT,R3
002504  012704  000004                  MOV     #4,R4
002510  013777  013200  176632  P3:     MOV     LOW,@VCXREG
002516  013777  013200  176626          MOV     LOW,@VCYREG
                                ;DRAW BOTTOM LINE
002524  012700  000377                  MOV     #377,R0
002530  012713  000004                  MOV     #4,(3)          ;ENABLE INTENSIFY ON LOADING X
002534  060411                  P3A:    ADD     R4,(1)
002536  105713                          TSTB    (3)
002540  100376                          BPL     .-2             ;WAIT FOR READY
002542  005300                          DEC     R0
002544  001373                          BNE     P3A             ;NO
                                ;DRAW RIGHT LINE
002546  012713  000010                  MOV     #10,(3)         ;ENABLE INTENSIFY ON LOADING Y
002552  012700  000377                  MOV     #377,R0
002556  060412                  P3B:    ADD     R4,(2)
002560  105713                          TSTB    (3)
002562  100376                          BPL     .-2             ;WAIT FOR READY
002564  005300                          DEC     R0
002566  001373                          BNE     P3B             ;NO
                                ;DRAW TOP LINE
002570  012713  000004                  MOV     #4,(3)          ;ENABLE INTENSIFY ON LOADING X
002574  012700  000377                  MOV     #377,R0
002600  160411                  P3C:    SUB     R4,(1)
002602  105713                          TSTB    (3)
002604  100376                          BPL     .-2             ;WAIT FOR READY
002606  005300                          DEC     R0
002610  001373                          BNE     P3C             ;NO
                                ;DRAW LEFT LINE
002612  012713  000010                  MOV     #10,(3)         ;ENABLE INTENSIFY LOADING Y
002616  012700  000377                  MOV     #377,R0
002622  160412                  P3D:    SUB     R4,(2)
002624  105713                          TSTB    (3)
002626  100376                          BPL     .-2             ;WAIT FOR READY
002630  005300                          DEC     R0
002632  001373                          BNE     P3D             ;NO
002634  004737  004726                  JSR     PC,TIMER
002640  000723                          BR      P3
                                ;;*****************************************************************
                                ;*TEST 4         PLOT AN X
                                ;;*****************************************************************
```

```
   589  002642  000004                    TST4:   SCOPE
   590  002644  012737  000001  001164            MOV     #1,STIMES       ;DO 1 ITERATION
   591  002652  012737  000000  013200    PIC4:   MOV     #0,LOW
   592  002660  012737  001774  013202            MOV     #1774,HIGH
   593  002666  005077  176454                    CLR     @VCSTAT
   594  002672  012737  000200  013210            MOV     #200,TICKS
   595  002700  004737  005050                    JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
   596  002704  013701  001350            PIC4A:  MOV     VCXREG,R1
   597  002710  013702  001352                    MOV     VCYREG,R2
   598  002714  013703  001346                    MOV     VCSTAT,R3
   599  002720  012704  000004                    MOV     #4,R4
   600  002724  013712  013200            P4:     MOV     LOW,(2)
   601  002730  011211                            MOV     (2),(1)
   602
   603                                    ;PLOT LINE BEGINNING IN LOWER LEFT CORNER
   604  002732  012713  000004                    MOV     #4,(3)          ;ENABLE INTENSIFY ON LOADING X
   605  002736  012700  000377                    MOV     #377,R0
   606  002742  105713                    P4A:    TSTB    (3)
   607  002744  100376                            BPL     .-2
   608  002746  060412                            ADD     R4,(2)          ;+4 TO Y
   609  002750  060411                            ADD     R4,(1)          ;+4 TO X
   610  002752  005300                            DEC     R0
   611  002754  001372                            BNE     P4A             ;NO
   612  002756  105713                            TSTB    (3)
   613  002760  100376                            BPL     .-2
   614                                    ;PLOT LINE BEGINNING IN UPPER LEFT CORNER
   615  002762  013712  013202                    MOV     HIGH,(2)
   616  002766  013711  013200                    MOV     LOW,(1)
   617  002772  012700  000377                    MOV     #377,R0
   618  002776  105713                    P4B:    TSTB    (3)
   619  003000  100376                            BPL     .-2
   620  003002  160412                            SUB     R4,(2)          ;-4 TO Y
   621  003004  060411                            ADD     R4,(1)          ;+4 TO X
   622  003006  005300                            DEC     R0
   623  003010  001372                            BNE     P4B             ;NO
   624  003012  004737  004726                    JSR     PC,TIMER
   625  003016  000742                            BR      P4
```

```
                                ;;*****************************************************************
                                ;*TEST 5         SCOPE SETTLING TIME TEST
                                ;;*****************************************************************
003020  000004          TST5:   SCOPE
003022  012737  000001  001164          MOV     #1,$TIMES       ;;DO 1 ITERATION
003030  012737  000200  013210  PIC5:   MOV     #200,TICKS      ;LOAD TIMER
003036  004737  005050                  JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
                                
003042  005077  176300          1$:     CLR     @VCSTAT         ;CLEAR STATUS
003046  005077  176276                  CLR     @VCXREG         ;CLEAR X AXIS
003052  005077  176274                  CLR     @VCYREG         ;CLEAR Y AXIS
003056  013703  001346                  MOV     VCSTAT,R3       ;LOAD STATUS ADDRESS
003062  032777  000200  176046          BIT     #BIT7,@SWR      ;TEST @SWR
003070  001407                          BEQ     3$              ;BR IF CLEARED
003072  013701  001352                  MOV     VCYREG,R1       ;LOAD R1
003076  013700  001350                  MOV     VCXREG,R0       ;LOAD R0
003102  012713  000010                  MOV     #10,(R3)        ;LOAD INTENS ON Y
003106  000406                          BR      4$
                                
003110  013700  001352          3$:     MOV     VCYREG,R0       ;LOAD R0
003114  013701  001350                  MOV     VCXREG,R1       ;LOAD R1
003120  012713  000004                  MOV     #4,(R3)         ;LOAD INTENS ON X
003124  004537  003154          4$:     JSR     R5,LODPNT       ;LOAD A LINE
003130  000000                          0
003132  004537  003154                  JSR     R5,LODPNT       ;LOAD A LINE
003136  001777                          1777
                                
003140  005710                          TST     (R0)            ;END
003142  001370                          BNE     4$              ;BR IF NOT
003144  004737  004726                  JSR     PC,TIMER        ;TEST TIME
003150  000734                          BR      1$
003152  000413                          BR      TST6            ;;
                                
003154  012702  000200  LODPNT: MOV     #200,R2
003160  011511          1$:     MOV     (R5),(R1)       ;LOAD AXIS
003162  105713          2$:     TSTB    (R3)            ;DONE
003164  100376                  BPL     2$              ;WAIT
003166  062710  000002          ADD     #2,(R0)         ;UPDATE AXIS
003172  005302                  DEC     R2              ;DONE
003174  001371                  BNE     1$              ;BR IF NOT
003176  005725                  TST     (R5)+           ;UPDATE
003200  000205                  RTS     R5              ;EXIT
```

```
                                        ;;**********************************************************************
                                        ;*TEST 6          PLOT CHARACTER SET
                                        ;;**********************************************************************
003202  000004                  TST6:   SCOPE
003204  012737  000001  001164          MOV     #1,STIMES       ;;DO 1 ITERATION
003212  012737  000100  013210  PIC6:   MOV     #100,TICKS
003220  004737  005050                  JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
003224  012737  000340  003546  PIC6A:  MOV     #340,XPOS
003232  012737  001400  003544          MOV     #1400,YPOS
003240  012702  003554                  MOV     #A,R2           ;LOAD STARTING CHARACTER
003244  005077  176076                  CLR     @VCSTAT
003250  004737  003346                  JSR     PC,PIC6B
003254  012737  000340  003546          MOV     #340,XPOS       ;LOAD X POS
003262  012737  001000  003544          MOV     #1000,YPOS      ;LOAD Y POS
003270  012702  003655                  MOV     #N,R2           ;LOAD STARTING CHARACTER
003274  005077  176046                  CLR     @VCSTAT
003300  004737  003346                  JSR     PC,PIC6B        ;DISPLAY CHARACTERS
003304  012737  000340  003546          MOV     #340,XPOS       ;LOAD X POS
003312  012737  000400  003544          MOV     #400,YPOS       ;LOAD Y POS
003320  012702  003756                  MOV     #N0,R2          ;LOAD STARTING CHARACTER
003324  005077  176016                  CLR     @VCSTAT
003330  004737  003346                  JSR     PC,PIC6B        ;DISPLAY CHARACTERS
003334  004737  004726                  JSR     PC,TIMER
003340  000731                          BR      PIC6A
003342  000137  004070                  JMP     PIC7
003346  012737  177763  003550  PIC6B:  MOV     #-13.,CHRCOL    ;CHARACTERS PER ROW
003354  013705  001346                  MOV     VCSTAT,R5
003360  000240                          NOP
003362  000240                          NOP
003364  000240                          NOP
003366  004737  003402          GEN1:   JSR     PC,CHAR
003372  005237  003550                  INC     CHRCOL          !
003376  001373                          BNE     GEN1
003400  000207                          RTS     PC
```

```
                                        ;PLOT CHARACTER
003402  013737  003544  003552  CHAR:   MOV     YPOS,YPT
003410  042715  000016                  BIC     #16,(5)
003414  013777  003546  175726          MOV     XPOS,@VCXREG
003422  013777  003544  175722          MOV     YPOS,@VCYREG
003430  105777  175712          CHAR4:  TSTB    @VCSTAT
003434  100375                  BPL     CHAR4
003436  053715  003542                  BIS     MODE,(5)                ;ENABLE INTENSIFY ON LOADING Y
003442  012704  000010                  MOV     #10,R4
003446  012700  177773                  MOV     #-5,R0          ;INITIALIZE COLUMN COUNT
003452  012701  177771          CHAR1:  MOV     #-7,R1          ;INITIALIZE ROW COUNT
003456  112203                  MOVB    (2)+,R3         ;PUT CHARACTER POINTS IN R3
003460  106103          CHAR2:  ROLB    R3
003462  100006                  BPL     CHAR3           ;NO
003464  013777  003546  175656          MOV     XPOS,@VCXREG
003472  013777  003544  175652          MOV     YPOS,@VCYREG
003500  105777  175642          CHAR3:  TSTB    @VCSTAT
003504  100375                  BPL     CHAR3
003506  060437  003544                  ADD     R4,YPOS
003512  005201                  INC     R1              ;+1 TO ROW
003514  001361                  BNE     CHAR2           ;FINISH ROW
003516  013737  003552  003544          MOV     YPT,YPOS        ;REINITIALIZE ROW FOR NEXT COLUMN
003524  060437  003546                  ADD     R4,XPOS
003530  005200                  INC     R0              ;+1 TO COLUMN COUNT
003532  001347                  BNE     CHAR1
003534  060437  003546                  ADD     R4,XPOS
003540  000207                  RTS     PC              ;EXIT

003542  000010          MODE:   10
003544  000000          YPOS:   0                       ;CONTAINS Y POSITION AT ANY GIVEN TIME
003546  000000          XPOS:   0                       ;CONTAINS X POSITION AT ANY GIVEN TIME
003550  000000          CHRCOL: 0
003552  000000          YPT:    0
```

```
                                        ;DATA BYTES FOR VISUAL CHARACTERS
003554  176  021  021  A:   .BYTE 176,21,21,21,176
003557  021  176
003561  177  111  111  B:   .BYTE 177,111,111,111,66
003564  111  066
003566  076  101  101  C:   .BYTE 76,101,101,101,42
003571  101  042
003573  177  101  101  D:   .BYTE 177,101,101,101,76
003576  101  076
003600  177  111  111  E:   .BYTE 177,111,111,111,101
003603  111  101
003605  177  011  011  F:   .BYTE 177,11,11,11,1
003610  011  001
003612  076  101  121  G:   .BYTE 76,101,121,121,62
003615  121  062
003617  177  010  010  H:   .BYTE 177,10,10,10,177
003622  010  177
003624  000  101  177  I:   .BYTE 0,101,177,101,0
003627  101  000
003631  060  100  100  J:   .BYTE 60,100,100,100,77
003634  100  077
003636  177  010  024  K:   .BYTE 177,10,24,42,101
003641  042  101
003643  177  100  100  L:   .BYTE 177,100,100,100,100
003646  100  100
003650  177  004  010  M:   .BYTE 177,4,10,4,177
003653  004  177
003655  177  004  010  N:   .BYTE 177,4,10,20,177
003660  020  177
003662  076  101  101  O:   .BYTE 76,101,101,101,76
003665  101  076
003667  177  011  011  P:   .BYTE 177,11,11,11,6
003672  011  006
003674  076  101  121  Q:   .BYTE 76,101,121,141,176
003677  141  176
003701  177  011  031  R:   .BYTE 177,11,31,51,106
003704  051  106
003706  046  111  111  S:   .BYTE 46,111,111,111,62
003711  111  062
003713  001  001  177  T:   .BYTE 1,1,177,1,1
003716  001  001
003720  077  100  100  U:   .BYTE 77,100,100,100,77
003723  100  077
003725  037  040  100  V:   .BYTE 37,40,100,40,37
003730  040  037
003732  177  020  010  W:   .BYTE 177,20,10,20,177
003735  020  177
003737  143  024  010  X:   .BYTE 143,24,10,24,143
003742  024  143
003744  003  004  170  Y:   .BYTE 3,4,170,4,3
003747  004  003
003751  141  121  111  Z:   .BYTE 141,121,111,105,103
003754  105  103
003756  076  121  111  N0:  .BYTE   76,121,111,105,76
```

```
   797  003761      105      076
   798  003763      000      102      177  N1:     .BYTE 0,102,177,100,0
   799  003766      100      000
   800  003770      142      121      111  N2:     .BYTE 142,121,111,105,102
   801  003773      105      102
   802  003775      042      101      111  N3:     .BYTE 42,101,111,111,66
   803  004000      111      066
   804  004002      030      024      022  N4:     .BYTE 30,24,22,177,20
   805  004005      177      020
   806  004007      047      105      105  N5:     .BYTE 47,105,105,105,71
   807  004012      105      071
   808  004014      076      111      111  N6:     .BYTE 76,111,111,111,62
   809  004017      111      062
   810  004021      101      041      021  N7:     .BYTE 101,41,21,11,7
   811  004024      011      007
   812  004026      066      111      111  N8:     .BYTE 66,111,111,111,66
   813  004031      111      066
   814  004033      046      111      111  N9:     .BYTE 46,111,111,111,76
   815  004036      111      076
   816  004040      000      000      000  SPACEA: .BYTE 0,0,0,0,0
   817  004043      000      000
   818  004045      000      000      000          .BYTE   0,0,0,0,0
   819  004050      000      000
   820  004052      000      000      000          .BYTE   0,0,0,0,0
   821  004055      000      000
   822
   823           004060                            .EVEN
```

```
                                        ;;**********************************************************************
                                        ;*TEST 7         CHANNEL 1 CHANNEL 2
                                        ;;**********************************************************************
004060  000004                  TST7:   SCOPE
004062  012737  000001  001164          MOV     #1,$TIMES       ;;DO 1 ITERATION
004070  012737  000200  013210  PIC7:   MOV     #200,TICKS      ;SET UP A TIMER
004076  004737  005050                  JSR     PC,CHTIME       ;CHANGE TIMER FOR CP TYPE
004102  013705  001346                  MOV     VCSTAT,R5       ;SET UP R5 FOR STATUS REGISTER POINTER
004106  012777  000000  175232  PIC7AA: MOV     #0,@VCSTAT      ;SET UP SCOPE CONTROL
004114  012737  000440  003546          MOV     #440,XPOS       ;LOAD X POSITION
004122  012737  001200  003544          MOV     #1200,YPOS      ;LOAD Y POSITION
004130  012737  000011  004262          MOV     #9.,P7CNT       ;SAVE THE NUMBER OF CHARACTERS
004136  012737  004266  004264          MOV     #CH01,P7PNT     ;SAVE CHANNEL 1 POINTER
004144  017702  000114          PIC7A:  MOV     @P7PNT,R2       ;MOVE MESSAGE POINTER INTO R2 FOR DISPLAY ROUTINE
004150  004737  003402                  JSR     PC,CHAR         ;DISPLAY A CHARACTER
004154  062737  000002  004264          ADD     #2,P7PNT        ;ADD 2 TO THE MESSAGE POINTER
004162  005337  004262                  DEC     P7CNT           ;DECREMENT CHARACTER COUNT
004166  001366                          BNE     PIC7A           ;NOT FINISHED WITH ALL CHARACTERS
004170  012777  001000  175150          MOV     #1000,@VCSTAT
004176  012737  000440  003546          MOV     #440,XPOS       ;SET UP X POS FOR CHANNEL 2
004204  012737  000600  003544          MOV     #600,YPOS       ;SET UP Y
004212  012737  000011  004262          MOV     #9.,P7CNT       ;SET UP CHARACTER COUNT
004220  012737  004310  004264          MOV     #CH02,P7PNT     ;SET UP CHANNEL 2 POINTER
004226  017702  000032          PIC7B:  MOV     @P7PNT,R2       ;SET UP
004232  004737  003402                  JSR     PC,CHAR         ;DISPLAY A CHARACTER
004236  062737  000002  004264          ADD     #2,P7PNT        ;ADD 2 TO THE POINTER
004244  005337  004262                  DEC     P7CNT           ;DECREMENT COUNT
004250  001366                          BNE     PIC7B           ;NOT FINISHED
004252  004737  004726                  JSR     PC,TIMER        ;CHECK THE RUNTIME OF THIS ROUTINE
004256  000713                          BR      PIC7AA          ;NOT FINISHED
004260  000424                          BR      TST10           ;;BR TO NEXT TEST
004262  000000          P7CNT:  0
004264  000000          P7PNT:  0
004266  003566          CH01:   C
004270  003617                  H
004272  003554                  A
004274  003655                  N
004276  003655                  N
004300  003600                  E
004302  003643                  L
004304  004040                  SPACEA
004306  003763                  N1
004310  003566          CH02:   C
004312  003617                  H
004314  003554                  A
004316  003655                  N
004320  003655                  N
004322  003600                  E
004324  003643                  L
004326  004040                  SPACEA
004330  003770                  N2
```

```
                                        ;;**********************************************************************
                                        ;*TEST 10        PHOSPHOR TEST
                                        ;;**********************************************************************
004332  000004                TST10:  SCOPE
004334  012737  000001  001164          MOV     #1,STIMES       ;;DO 1 ITERATION
004342  012737  000002  013210  PIC12:  MOV     #2,TICKS
004350  032777  000040  174560  PIC12A: BIT     #BIT5,@SWR              ;TEST IF STORAGE SCOPE
004356  001402                        BEQ     1$                      ;BR IF NOT
004360  004737  004536                JSR     PC,CLRVC                ;ERASE THE SCREEN
004364  004737  004626          1$:   JSR     PC,LOADVC               ;LOAD THE SCREEN
004370  004737  004726                JSR     PC,TIMER                ;CHECK THE TIME
004374  000765                        BR      PIC12A
004376  032777  000040  174532        BIT     #BIT5,@SWR              ;TEST IF STORAGE SCOPE
004404  001402                        BEQ     2$                      ;BR IF NOT
004406  004737  004536                JSR     PC,CLRVC
004412  000005                2$:     RESET

                                        .SBTTL  END OF PASS ROUTINE

                                        ;;**********************************************************************
                                        ;*INCREMENT THE PASS NUMBER (SPASS)
                                        ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                        ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO VSUAL0

004414                          SEOP:
004414  000004                        SCOPE
004416  005037  001102                CLR     STSTNM          ;;ZERO THE TEST NUMBER
004422  005037  001164                CLR     STIMES          ;;ZERO THE NUMBER OF ITERATIONS
004426  005237  001206                INC     SPASS           ;;INCREMENT THE PASS NUMBER
004432  042737  100000  001206        BIC     #100000,SPASS   ;;DON'T ALLOW A NEG. NUMBER
004440  005327                        DEC     (PC)+           ;;LOOP?
004442  000001                SEOPCT: .WORD   1
004444  003022                        BGT     $DOAGN          ;;YES
004446  012737                        MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
004450  000001                SENDCT: .WORD   1
004452  004442                        SEOPCT
004454  104400  004521                TYPE    ,SENDMG         ;;TYPE "END PASS #"
004460  013746  001206                MOV     SPASS,-(SP)     ;;SAVE SPASS FOR TYPEOUT
004464  104404                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
004466  104400  004516                TYPE    ,SENULL         ;;TYPE A NULL CHARACTER
004472  013700  000042        SGET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
004476  001405                        BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
004500  000005                        RESET                   ;;CLEAR THE WORLD
004502  004710                SENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
004504  000240                        NOP                     ;;SAVE ROOM
004506  000240                        NOP                     ;;FOR
004510  000240                        NOP                     ;;ACT11
004512                          $DOAGN:
004512  000137                        JMP     @(PC)+          ;;RETURN
004514  002254                SRTNAD: .WORD   VSUAL0
004516     377     377     000  SENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
004521     015  042412  042116  SENDMG: .ASCIZ  <15><12>/END PASS #/
004526  050040  051501  020123
004534  000043
```

```
                                                                                
                                                                                
004536  012777  002000  174602  CLRVC:  MOV     #BIT10,@VCSTAT  ;ERASE THE SCREEN
004544  052777  010000  174574          BIS     #BIT12,@VCSTAT
004552  000240                          NOP
004554  012700  000020                  MOV     #20,R0          ;SET UP DELAY
004560  005001                          CLR     R1
004562  105777  174560          CLRVCA: TSTB    @VCSTAT         ;TEST FOR READY
004566  100416                          BMI     CLRVCB          ;BRANCH IF SET
004570  005301                          DEC     R1              ;DELAY
004572  001373                          BNE     CLRVCA
004574  005300                          DEC     R0              ;DELAY
004576  001371                          BNE     CLRVCA
004600  037727  174332  010000          BIT     @SWR,#SW12      ;TEST INHIBIT PRINTOUT
004606  001002                          BNE     1$
004610  104400                          TYPE
004612  012023                          MES3
004614  005777  174316          1$:     TST     @SWR            ;TEST @SWR
004620  100001                          BPL     CLRVCB
004622  000000                          HALT                    ;ERASE RETURN FAILED TO SET READY
                                                                                
004624  000207          CLRVCB: RTS     PC
                                                                                
004626  005077  174514          LOADVC: CLR     @VCSTAT         ;CLEAR STATUS
004632  012737  001777  013230          MOV     #1777,TEMP1
004640  013700  001346                  MOV     VCSTAT,R0
004644  013701  001350                  MOV     VCXREG,R1
004650  013702  001352                  MOV     VCYREG,R2
004654  012710  002000                  MOV     #BIT10,(0)      ;SET STORE MODE
004660  013712  013230                  MOV     TEMP1,(2)
004664  012711  001777          LODVCA: MOV     #1777,(1)
004670  000402                          BR      LODVCC
004672  162711  000004          LODVCB: SUB     #4,(1)
004676  005210          LODVCC: INC     (0)
004700  000240                          NOP
004702  105710                          TSTB    (0)
004704  100376                          BPL     .-2
004706  022711  000003                  CMP     #3,(1)
004712  001367                          BNE     LODVCB
004714  104405                          CKSWR                   ;TEST FOR "CTRL G"
004716  162712  000001                  SUB     #1,(2)
004722  001360                          BNE     LODVCA
004724  000207                          RTS     PC
                                                                                
```

```
                                        ;TIMER ROUTINE
                                        ; ENTER VIA JSR PC,TIMER
                                        
004726  104405                  TIMER:  CKSWR
004730  017737  174202  013206          MOV     @SWR,TIMSV
004736  004737  011672                  JSR     PC,TSTFLG
004742  032737  000400  013206  TIMERA: BIT     #BIT8,TIMSV
004750  001006                          BNE     TIMER2          ;BIT 8 SET ?
004752  005337  013210                  DEC     TICKS           ;NO, DECREMENT TICKS
004756  001002                          BNE     TIMER1
004760  062716  000002                  ADD     #2,(6)          ;ADD 2 TO STACK POINTER
004764  000207                  TIMER1: RTS     PC              ;RETURN
                                        
                                        ; SWR 8=1 SELECT TEST TO LOCK ON
                                        ; SWR 2-0= TEST NUMBER
                                        
004766  042737  177770  013206  TIMER2: BIC     #177770,TIMSV
004774  006337  013206                  ASL     TIMSV
005000  062737  005030  013206          ADD     #ROUTPT,TIMSV
005006  017737  006174  013206          MOV     @TIMSV,TIMSV
005014  022600                          CMP     (SP)+,R0
005016  000240                          NOP
005020  000240                          NOP
005022  000240                          NOP
005024  000177  006156          TIMER4: JMP     @TIMSV
                                        
005030  002266                  ROUTPT: PIC0                    ;DISPLAY A HORIZONTAL LINE
005032  002332                          PIC1                    ;DISPLAY A VERTICAL LINE
005034  002446                          PIC3                    ;DISPLAY A SQUARE
005036  002652                          PIC4                    ;DISPALY A "X"
005040  003030                          PIC5                    ;DISPLAY SETTLING TIME
005042  003212                          PIC6                    ;DISPLAY CHARACTER SET
005044  004070                          PIC7                    ;DISPLAY CHANNEL TEST
005046  004342                          PIC12                   ;DISPLAY ERASE AND PHOSPHOR TEST
                                        
005050  013737  010750  013236  CHTIME: MOV     CPTYPE,BRLEV1
005056  005337  013236          CHTMA:  DEC     BRLEV1
005062  001403                          BEQ     CHTMB
005064  006337  013210                  ASL     TICKS
005070  000772                          BR      CHTMA
005072  000207                  CHTMB:  RTS     PC
```

```
                                ;;*****************************************************************
                                ;*TEST 11        CALIBRATION ROUTINE
                                ;;*****************************************************************
005074  000004                  TST11:  SCOPE
005076  012737  000001  001164          MOV     #1,STIMES       ;;DO 1 ITERATION
                                ;*****************************************************************

                                ;ROUTINE REQUESTS THE TYPE OF 'SYNC' TO BE USED ('I' INTERNAL OR 'E' EXTERNAL
                                ;OR 'C' AR CLOCK)
                                ;THE PROGRAM THEN TAKES CONTINUOUS CONVERSIONS USING DATA SW'S 3-0
                                ;TO SELECT THE CH,, AND SW5 TO SELECT UNIPOLAR/BIPOLAR
                                ;USE SW '6' TO FREEZE ON SINGLE 512 POINT BURST.
                                ;USE SW '7' TO DISPLAY DATA ON Y AXIS TO CHECK FOR A/D INTERACTION
                                ;USE SW '10' TO PRINT THE CONVERSION VALUE.

005104  012737  005116  013152  CALBRT: MOV     #CALBT1,AVECTR  ;SET UP '^A' RESTART ADDRESS
005112  104400                          TYPE
005114  012420                          MES7                    ;TYPE TEST HEADER
005116  104400                  CALBT1: TYPE
005120  012513                          MES10                   ;TEXT 'SYNC I OR E OR C'
005122  004737  010206                  JSR     PC,XTYYIN                       ;WAIT FOR INPUT.
005126  013737  010436  013154          MOV     INBUF,PROC      ;SAVE IT IN TEMP STORAGE
005134  104400                          TYPE
005136  012356                          ACRLF
005140  005037  013222                  CLR     USECLK          ;CLEAR CLOCK FLAG
005144  012737  000001  013160          MOV     #1,COUNT        ;SET UP FOR '1' CONVERSION
005152  117737  173760  013147  CALBT2: MOVB    @SWR,ADWRD2+1   ;GET CH, FROM THE SW REG.
005160  042737  150377  013146          BIC     #150377,ADWRD2  ;CLR UNWANTED BITS, A/D WORD COMPLETE
005166  005037  006224                  CLR     ADWRD1
005172  017737  173740  013164          MOV     @SWR,KSTOR2     ;SAVE ORIGINAL SWITCH SETTING.
005200  022737  000105  013154          CMP     #105,PROC       ;TEST SYNC SELECT
005206  001004                          BNE     1$              ;BRANCH IF NOT 'E'
005210  052737  000120  006224          BIS     #120,ADWRD1     ;OTHERWISE ADD 'EXT' SYNC BIT TO A/D.
005216  000416                          BR      CAL82A
005220  122737  000103  013154  1$:     CMPB    #103,PROC
005226  001007                          BNE     2$              ;BRANCH IF NOT 'C'
005230  052737  000140  006224          BIS     #140,ADWRD1     ;OTHERWISE ADD 'CLOCK' SYNC TO A/D
005236  012737  177777  013222          MOV     #-1,USECLK
005244  000403                          BR      CAL82A
005246  052737  000101  006224  2$:     BIS     #101,ADWRD1
005254  012777  006154  173766  CAL82A: MOV     #CNSTS1,@SVECT1
005262  000240                          NOP
```

```
1061  005264  012705  016200            MOV     #ADBUFF,R5              ;RESET POINTER
1062  005270  000240              11$:  NOP
1063  005272  000240                    NOP
1064  005274  000240                    NOP
1065  005276  013777  013146  174032    MOV     ADWRD2,@ADCS            ;LOAD A/D STATUS
1066  005304  000240                    NOP
1067  005306  005077  174034            CLR     @VCSTAT                 ;CLEAR STAT
1068  005312  005077  174032            CLR     @VCXREG
1069  005316  005077  174030            CLR     @VCYREG
1070  005322  032777  000200  173606    BIT     #BIT7,@SWR              ;TEST BIT
1071  005330  001410                    BEQ     2$                      ;BR IF DOWN
1072  005332  013703  001350            MOV     VCXREG,R3               ;LOAD X ADDRESS
1073  005336  013704  001352            MOV     VCYREG,R4               ;LOAD Y ADDRESS
1074  005342  012777  000004  173776    MOV     #4,@VCSTAT              ;INTEN. ON X LOAD
1075  005350  000407                    BR      3$                      ;
1076  005352  013703  001352      2$:   MOV     VCYREG,R3               ;LOAD Y ADDRESS
1077  005356  013704  001350            MOV     VCXREG,R4               ;LOAD X ADDRESS
1078  005362  012777  000010  173756    MOV     #10,@VCSTAT             ;INTEN. ON Y LOAD
1079  005370  017737  173752  006226  3$:  MOV     @VCSTAT,AXIS1           ;SAVE .STATUS
1080  005376  010337  006230            MOV     R3,AXIS2                ;SAVE R3
1081  005402  010437  006232            MOV     R4,AXIS3                ;SAVE R4
1082  005406  012737  000004  006054    MOV     #4,10$                  ;LOAD COUNT
1083  005414  012700  000160      10$:  MOV     #160,R0                 ;LOAD A CONVERSION COUNT
1084  005420  012701  000000            MOV     #0,R1                   ;CLEAR OFFSET
1085  005424  005037  006222            CLR     MARKER                  ;NOT A MARKER SAMPLE
1086  005430  004737  006112            JSR     PC,CNVTSV               ;CONVERT AND SAVE
1087
1088  005434  012700  000020            MOV     #20,R0                  ;LOAD MARKER COUNT
1089  005440  012701  000011            MOV     #11,R1                  ;LOAD OFFSET
1090  005444  005237  006222            INC     MARKER                  ;SET MARKER SAMPLE
1091  005450  004737  006112            JSR     PC,CNVTSV               ;CONVERT AND SAVE
1092
1093  005454  005337  006054            DEC     10$                     ;DONE ?
1094  005460  001355                    BNE     10$                     ;BR IF NOT
1095
1096                          ;CONVERT INTO DIGITS
1097  005462  012737  000077  013230  15$:  MOV     #77,TEMP1               ;LOAD CONVERSION COUNT
1098  005470  012737  000006  011330        MOV     #6,CMPCNT               ;LOAD SHIFT COUNT
1099  005476  004737  011166                JSR     PC,CMPTEB               ;AVERAGE
1100  005502  013700  013266                MOV     AVRAGE,R0               ;GET A VALUE
1101  005506  006200                        ASR     R0
1102  005510  006200                        ASR     R0
1103  005512  005200                        INC     R0
1104  005514  006200                        ASR     R0
1105  005516  010037  006056                MOV     R0,10$                  ;SAVE FOR TYPEOUT
1106  005522  012737  000004  006052        MOV     #4,10$                  ;LOAD A COUNT
1107  005530  012702  006070                MOV     #DGT4,R2                ;LOAD A POINTER
1108  005534  000403                        BR      16$
1109  005536  006200              17$:  ASR     R0                      ;MOVE RIGHT
1110  005540  006200                    ASR     R0
1111  005542  006200                    ASR     R0
1112  005544  010001              16$:  MOV     R0,R1                   ;GET VALUE
1113  005546  042701  177770            BIC     #177770,R1              ;MASK
1114  005552  006301                    ASL     R1                      ;X2
1115  005554  016142  006072            MOV     AN0(R1),-(R2)           ;LOAD ADDRESS
1116  005560  005337  006052            DEC     10$                     ;FINISHED ?
```

```
1117  005564  001364                          BNE     17$                     ;BR IF NOT
1118                                  ;DISPLAY CONVERTED DIGITS
1119  005566  005077  173554          14$:    CLR     @VCSTAT
1120  005572  013705  001346                  MOV     VCSTAT,R5               ;LOAD R5
1121  005576  012737  000000  003546          MOV     #0,XPOS                 ;LOAD X POSITION
1122  005604  012737  000200  003544          MOV     #200,YPOS               ;LOAD Y POSITION
1123  005612  004737  006010                  JSR     PC,30$                  ;DISPLAY DIGITS
1124  005616  012737  001500  003546          MOV     #1500,XPOS              ;LOAD X POS
1125  005624  012737  001500  003544          MOV     #1500,YPOS              ;LOAD Y POS
1126  005632  004737  006010                  JSR     PC,30$
1127  005636  032777  000100  173272          BIT     #BIT6,@SWR              ;TEST @SWR
1128  005644  001432                          BEQ     22$                     ;BR IF CONTINOUS SAMPLES
1129
1130  005646  012705  016200                  MOV     #ADBUFF,R5              ;LOAD BUFFER POINTER
1131  005652  012700  001000                  MOV     #1000,R0                ;LOAD COUNT
1132  005656  005077  173464                  CLR     @VCSTAT                 ;CLEAR STATUS
1133  005662  013703  006230                  MOV     AXIS2,R3                ;RESTORE R3
1134  005666  013704  006232                  MOV     AXIS3,R4                ;RESTORE R4
1135  005672  005013                          CLR     (R3)
1136  005674  005014                          CLR     (R4)
1137  005676  013777  006226  173442          MOV     AXIS1,@VCSTAT           ;RESTORE STATUS
1138  005704  012513                  13$:    MOV     (R5)+,(R3)              ;LOAD POINT
1139  005706  105777  173434          12$:    TSTB    @VCSTAT                 ;WAIT FOR SCOPE
1140  005712  100375                          BPL     12$
1141  005714  062714  000002                  ADD     #2,(R4)                 ;UPDATE AXIS
1142  005720  005300                          DEC     R0                      ;DONE ?
1143  005722  001370                          BNE     13$                     ;BR IF NOT
1144  005724  004737  011672                  JSR     PC,TSTFLG               ;TEST KEYBOARD FLAG
1145  005730  000716                          BR      14$
1146  005732  012737  000001  013234  22$:    MOV     #1,TEMP3                ;SETUP TO PRINT '1' VALUE
1147  005740  032777  002000  173170          BIT     #SW10,@SWR
1148  005746  001406                          BEQ     21$
1149  005750  013746  006056                  MOV     102$,-(SP)
1150  005754  104402                          TYPOS
1151  005756     004     001                  .BYTE   4,1
1152  005760  104400                          TYPE
1153  005762  012356                          ACRLF
1154  005764  004737  011672          21$:    JSR     PC,TSTFLG               ;TEST FOR KEYBOARD INTERRUPT
1155  005770  023777  013164  173140          CMP     KSTOR2,@SWR             ;TEST IF SWITCH REGISTER HAS CHANGED
1156  005776  001402                          BEQ     23$                     ;BRANCH AND TAKE NEXT BURST OF 512 CONVERSIONS
1157  006000  000137  005152                  JMP     CALBT2                  ;YES, COMPUTE NEW INPUT
1158  006004  000137  005254          23$:    JMP     CALB2A
```

```
 1159  006010  012737  000004  006054  30$:    MOV     #4,10I$                 ;LOAD COUNT
 1160  006016  012737  006060  006052          MOV     #DGT0,100$              ;LOAD POINTER
 1161  006024  017702  000022          1A$:    MOV     #100$,R2                ;GET ADDRESS
 1162  006030  004737  003402                  JSR     PC,CHAR                 ;DISPLAY #
 1163  006034  062737  000002  006052          ADD     #2,100$                 ;UPDATE POINTER
 1164  006042  005337  006054                  DEC     10I$                    ;DEC COUNTER
 1165  006046  001366                          BNE     1A$
 1166  006050  000207                          RTS     PC                      ;EXIT
 1167  006052  000000                  100$:   0
 1168  006054  000000                  10I$:   0
 1169  006056  000000                  102$:   0
 1170  006060  000000                  DGT0:   0
 1171  006062  000000                  DGT1:   0
 1172  006064  000000                  DGT2:   0
 1173  006066  000000                  DGT3:   0
 1174  006070  000000                  DGT4:   0
 1175  006072  003756                  AN0:    N0
 1176  006074  003763                          N1
 1177  006076  003770                          N2
 1178  006100  003775                          N3
 1179  006102  004002                          N4
 1180  006104  004007                          N5
 1181  006106  004014                          N6
 1182  006110  004021                          N7
 1183
 1184  006112  005737  006222          CNVTSV: TST     MARKER                  ;TEST IF MARKER SAMPLE
 1185  006116  001401                          BEQ     4$                      ;BR IF NOT
 1186  006120  005301                          DEC     R1                      ;DEC OFFSET
 1187  006122  005737  013222          4$:     TST     @#USECLK                ;CLOCK ENABLE
 1188  006126  001406                          BEQ     5$                      ;BR IF NOT
 1189  006130  012777  177750  173206          MOV     #-30,@#CSR              ;LOAD PRESET
 1190  006136  012777  000003  173176          MOV     #3,@#CSR                ;LOAD RATE
 1191  006144  153777  006224  173164  5$:     BISB    ADWRD1,@#ADCS           ;ENABLE A/D
 1192  006152  000001                          WAIT
 1193  006154  105077  173156          CNSTS1: CLRB    @#ADCS
 1194  006160  022626                          CMP     (SP)+,(SP)+             ;CLEAN STACK
 1195  006162  017702  173152                  MOV     @#ADDBR,R2              ;LOAD CONVERTED VALUE
 1196  006166  060102                          ADD     R1,R2                   ;ADD OFFSET IF ANY
 1197  006170  006302                          ASL     R2                      ;X8
 1198  006172  006302                          ASL     R2
 1199  006174  006302                          ASL     R2
 1200  006176  010213                          MOV     R2,(R3)                 ;LOAD AN AXIS
 1201  006200  010225                          MOV     R2,(R5)+                ;SAVE IN BUFFER
 1202  006202  105777  173140          5$:     TSTB    @#VCSTAT                ;WAIT FOR SCOPE
 1203  006206  100375                          BPL     5$
 1204  006210  062714  000002                  ADD     #2,(R4)                 ;UPDATE D/A AXIS
 1205  006214  005300                          DEC     R0                      ;DONE ALL CONVERSIONS ?
 1206  006216  001335                          BNE     CNVTSV                  ;BR IF NOT
 1207  006220  000207                          RTS     PC                      ;EXIT
 1208  006222  000000                  MARKER: 0
 1209  006224  000101                  ADWRD1: 101
 1210  006226  000000                  AXIS1:  0
 1211  006230  000000                  AXIS2:  0
 1212  006232  000000                  AXIS3:  0
```

```
                                ;;**********************************************************************
                                ;*TEST 12        REPEATABILITY TEST
                                ;;**********************************************************************
006234  000004                  TST12:  SCOPE
006236  012737  000001  001164          MOV     #1,$TIMES       ;;DO 1 ITERATION

                                ;THIS ROUTINE TO DESIGNED TO SHOW REPEATABILITY BY TAKING A SERIES OF
                                ;'512' CONVERSIONS,  AVERAGING THEM AND THEN CATEGORIZING
                                ;THEM IN BINS FROM THE AVERAGE PLUS & MINUS 6 COUNTS.  THE ROUTINE
                                ;REQUESTS FOR A CHANNEL OR CHANNELS, AND A COUNT SPREAD TO BE TYPED
                                ;IN VIA THE OPERATOR.  A CONTINUOUS SERIES OF CONVERSIONS ARE THEN TAKEN
                                ;AND COMPARED AGAINST THE INPUT COUNT SPREAD.  IF ALL '512' CONVERSIONS
                                ;ARE FOUND TO BE WITHIN THE SPREAD THE NEXT CH. IS EXERCISED OTHERWISE
                                ;THE COUNTS ARE TYPED OUT.  SETTING SWITCH '10' TO A '1' WILL FORCE A PRINTOUT
                                ;OF THE CH (S).

006244  012737  006256  013152  REPTST: MOV     #REPT1,AVECTR   ;SET UP CNTR 'A' VECTOR ADDRESS
006252  104400                          TYPE
006254  012561                          MES13                   ;TEXT 'REPEATABILITY TEST'
006256  005037  013244          REPT1:  CLR     MESPRT
006262  005037  013224                  CLR     OPS1
006266  104400                          TYPE
006270  012617                          MES14                   ;REQUEST CHANNEL (S)
006272  004737  010206                  JSR     PC,XTTYIN                       ;WAIT FOR INPUT
006276  004737  010454                  JSR     PC,BCDBIN               ;CONVERT TO OCTAL
006302  013737  010610  013162          MOV     BCDTAB,KSTOR1   ;SAVE AS INITIAL CH.
006310  013737  013162  013164          MOV     KSTOR1,KSTOR2   ;ALSO SAVE AS 2ND CH. ENTRY
006316  005737  010612                  TST     BCDTAB+2        ;TEST FOR SECOND ENTRY
006322  001407                          BEQ     REPT2           ;BRANCH IF NO SECOND ENTRY
006324  023737  010612  013162          CMP     BCDTAB+2,KSTOR1 ;COMPARE ENTRY 1 TO ENTRY 2
006332  100751                          BMI     REPT1           ;BRANCH AND RESTART IF ILLEGAL
006334  013737  010612  013164          MOV     BCDTAB+2,KSTOR2 ;OTHERWISE SAVE AS SECOND CH.
006342  104400                  REPT2:  TYPE
006344  012701                          MES16                   ;TEXT 'COUNT SPREAD ?'
006346  004737  010206                  JSR     PC,XTTYIN                       ;WAIT FOR ENTRY
006352  004737  010454                  JSR     PC,BCDBIN               ;DECODE TO OCTAL
006356  013737  010610  013166          MOV     BCDTAB,KSTOR3   ;SAVE IT
006364  013737  013162  013170  REPT2A: MOV     KSTOR1,KSTOR4   ;SAVE STARTING CH.
006372  004737  011672          REPT3:  JSR     PC,TSTFLG                       ;TEST FOR KEYBOARD FLAG
006376  012737  001000  013160          MOV     #1000,COUNT     ;SET FOR '512' CONVERSIONS
006404  113737  013170  013147          MOVB    KSTOR4,ADWRD2+1 ;MOV SELECTED CH. TO HIGH BYTE OF ADWORD
006412  042737  100377  013146          BIC     #100377,ADWRD2  ;MASK
006420  052737  000001  013146          BIS     #1,ADWRD2
006426  004737  010620                  JSR     PC,ADCNVT       ;TAKE THE CONVERSIONS
006432  004737  011152                  JSR     PC,CMPTE                ;AVERAGE & COMPUTE DISTRIBUTION
006436  004737  011332                  JSR     PC,CATORZ
006442  032777  002000  172466          BIT     #SW10,@SWR      ;TEST DATA SW10
006450  001047                          BNE     REPT4           ;IF SET, FORCE TYPE OUT
006452  032777  020000  172456  TSTCT4: BIT     #SW13,@SWR      ;TEST FOR INHIBIT TYPEOUT
006460  001142                          BNE     REPT7           ;BRANCH IF SW SET
006462  022737  000004  013166          CMP     #4,KSTOR3       ;WAS 4 TYPED
006470  001005                          BNE     TSTCT3          ;NO, TEST FOR '3'
006472  022737  001000  013342          CMP     #1000,XSPRD4    ;TOTAL COUNTS WITHIN 4 COUNTS
006500  001033                          BNE     REPT4           ;BRANCH IF NO.
```

```
1269  006502  000531                              BR      REPT7           ;YES, TEST NEXT CH.
1270  006504  022737  000003  013166  TSTCT3: CMP     #3,KSTOR3       ;COUNT = TO 3
1271  006512  001005                              BNE     TSTCT2          ;NO TEST COUNT 2
1272  006514  022737  001000  013340          CMP     #1000,XSPRD3
1273  006522  001022                              BNE     REPT4           ;BRANCH IF COUNT NOT WITHIN 3
1274  006524  000520                              BR      REPT7           ;YES, TEST NEXT CH.
1275
1276  006526  022737  000002  013166  TSTCT2: CMP     #2,KSTOR3       ;COUNT =TO 2
1277  006534  001005                              BNE     TSTCT1          ;NO, TEST COUNT 1
1278  006536  022737  001000  013336          CMP     #1000,XSPRD2
1279  006544  001011                              BNE     REPT4           ;BRANCH IF NOT WITHIN 2
1280  006546  000507                              BR      REPT7           ;YES, TEST NEXT CH.
1281  006550  022737  000001  013166  TSTCT1: CMP     #1,KSTOR3       ;COUNT = TO 1
1282  006556  001004                              BNE     REPT4           ;NO, REPORT EVEN IF NOT '0'
1283  006560  022737  001000  013334          CMP     #1000,XSPRD1
1284  006566  001477                              BEQ     REPT7           ;BRANCH IF TOTAL WITHIN 1 COUNT
1285  006570  104400                  REPT4:  TYPE
1286  006572  012356                          ACRLF
1287  006574  005737  013244                  TST     MESPRT          ;TEST IF HEADER HAS BEEN TYPED
1288  006600  001002                          BNE     REPT5           ;BRANCH IF YES
1289  006602  104400                          TYPE
1290  006604  012720                          MES19                   ;TEXT 'CH, LOW AVG, HIGH'
1291  006606  104400                  REPT5:  TYPE
1292  006610  012356                          ACRLF                   ;CARRIAGE RETURN, LINE FEED
1293  006612  013737  013170  007020          MOV     KSTOR4,REPT8A   ;MOV. CH.
1294  006620  042737  177700  007020          BIC     #177700,REPT8A
1295  006626  013746  007020          REPT8:  MOV     REPT8A,-(SP)
1296  006632  104402                          TYPOS
1297  006634     002     001                  .BYTE   2,1
1298  006636  004737  007662                  JSR     PC,XSPACE
1299  006642  013746  013252                  MOV     ADLOW,-(SP)     ;SAVE  LOW VALUE
1300  006646  104402                          TYPOS
1301  006650     004     001                  .BYTE   4,1
1302  006652  004737  007662                  JSR     PC,XSPACE
1303  006656  013746  013266                  MOV     AVRAGE,-(SP)    ;SAVE AVERAGE
1304  006662  104402                          TYPOS
1305  006664     004     001                  .BYTE   4,1
1306  006666  004737  007662                  JSR     PC,XSPACE
1307  006672  013746  013250                  MOV     ADHIGH,-(SP)
1308  006676  104402                          TYPOS
1309  006700     004     001                  .BYTE   4,1
1310  006702  005737  013244                  TST     MESPRT
1311  006706  001002                          BNE     REPT6
1312  006710  104400                          TYPE
1313  006712  012743                          MES20                   ;PRINT 'COUNT SPREAD' HEADER
1314  006714  052737  000007  013244  REPT6:  BIS     #7,MESPRT       ;INHIBIT OTHER HEADERS
1315  006722  022737  001000  013316          CMP     #1000,AVGCNT    ;TEST IF ALL COUNTS WERE AT AVG.
1316  006730  000240                          NOP                     ;<BEQ REPT7> BRANCH TO NEXT CH. IF YES.
1317  006732  104400                          TYPE
1318  006734  012356                          ACRLF
1319  006736  012704  013302                  MOV     #ORLOW,R4
1320  006742  012402                  REPT6A: MOV     (R4)+,R2
1321  006744  004737  010754                  JSR     PC,DECPRT                       ;TYPE OUT COUNT SPREAD
1322  006750  022704  013334                  CMP     #XSPRD1,R4      ;TEST FOR DONE
1323  006754  001372                          BNE     REPT6A          ;BRANCH IF NO AND TYPE NEXT COUNT
1324  006756  005777  172154                  TST     @SWR
```

```
1325  006762  100001                          BPL     REPT7
1326  006764  000000                          HALT                    ;REPEATABILITY ERROR
1327  006766  013704  013170          REPT7:  MOV     KSTOR4,R4
1328  006772  042704  177700                  BIC     #177700,R4
1329  006776  023704  013164                  CMP     KSTOR2,R4       ;TESTED ALL CH.(S)?
1330  007002  001404                          BEQ     REPT7A
1331  007004  005237  013170                  INC     KSTOR4          ;TEST NEXT CHANNEL
1332  007010  000137  006372                  JMP     REPT3           ;
1333  007014  000137  006364          REPT7A: JMP     REPT2A
1334
1335  007020  000000          REPT8A: 0
```

```
                                ;;*****************************************************************
                                ;*TEST 13       RECOVERY TEST
                                ;;*****************************************************************
007022  000004                  TST13:  SCOPE
                                ;THIS TEST IS DESIGNED TO TEST INTER-CHANNEL SETTLING OF THE A/D CONVERTER BY
                                ;ACCEPTING TWO (2) CHANNELS FROM THE TELETYPE AND THEN TAKING A
                                ;SERIES OF EIGHT CONVERSIONS ON EACH CHANNEL AND TYPING OUT THE CON-
                                ;VERSION VALUES OF THE 2ND CHANNEL IN THE ORDER THEY WERE TAKEN.

007024  012737  007036  013152  RECVRY: MOV     #RECVY1,AVECTR  ;SET UP THE '"A' RETURN ADDRESS
007032  104400                          TYPE
007034  012454                          MESA                    ;TEXT 'RECOVERY TEST'
007036  104400                  RECVY1: TYPE                    ;REQUEST CHANNELS
007040  012617                          MES14
007042  004737  010206                  JSR     PC,XTTYIN                       ;WAIT FOR INPUT
007046  004737  010454                  JSR     PC,BCDBIN               ;CONVERT TO OCTAL
007052  013737  010610  013162          MOV     BCDTAB,KSTOR1   ;SAVE 1ST CH.
007060  013737  010612  013164          MOV     BCDTAB+2,KSTOR2 ;SAVE 2ND CH.
007066  004737  011672          RECVY2: JSR     PC,TSTFLG                       ;CHECK FOR KEYBOARD FLAG
007072  012737  000020  007406          MOV     #16.,10$                ;LOAD COUNT
007100  005037  007410                  CLR     11$
007104  012737  000010  013234          MOV     #10,TEMP3       ;SET UP TO PRINT EIGHT
007112  012737  000010  013160          MOV     #10,COUNT       ;SETUP TO TAKE '8' CONVERSIONS
007120  113737  013162  013147  1$:     MOVB    KSTOR1,ADWRD2+1 ;LOAD 1ST CH.
007126  042737  140377  013146          BIC     #140377,ADWRD2
007134  052737  000001  013146          BIS     #1,ADWRD2
007142  005237  013216                  INC     DELAY   ;NO DELAY BETWEEN MUX CHANGE AND CONVERT
007146  004737  010620                  JSR     PC,ADCNVT       ;TAKE THE CONVERSIONS
007152  113737  013164  013147          MOVB    KSTOR2,ADWRD2+1 ;SET UP 2ND CH.
007160  042737  140377  013146          BIC     #140377,ADWRD2
007166  052737  000001  013146          BIS     #1,ADWRD2
007174  005237  013216                  INC     DELAY           ;NO DELAY BETWEEN MUX CHANGE AND CONVERT
007200  004737  010620                  JSR     PC,ADCNVT       ;TAKE 2ND SERIES OF CONVERSIONS
007204  013701  007410                  MOV     11$,R1          ;LOAD A POINTER
007210  012700  016200                  MOV     #ADBUFF,R0      ;LOAD POINTER
007214  012061  016242                  MOV     (R0)+,ADTB0(R1)
007220  012061  016304                  MOV     (R0)+,ADTB1(R1)
007224  012061  016346                  MOV     (R0)+,ADTB2(R1)
007230  012061  016410                  MOV     (R0)+,ADTB3(R1)
007234  012061  016452                  MOV     (R0)+,ADTB4(R1)
007240  012061  016514                  MOV     (R0)+,ADTB5(R1)
007244  012061  016556                  MOV     (R0)+,ADTB6(R1)
007250  012061  016620                  MOV     (R0)+,ADTB7(R1)
007254  062737  000002  007410          ADD     #2,11$          ;UPDATE POINTER
007262  005337  007406                  DEC     10$
007266  001314                          BNE     1$

007270  012700  000010                  MOV     #8.,R0          ;LOAD COUNT
007274  012702  016200                  MOV     #ADBUFF,R2      ;LOAD POINTER
007300  012701  000000                  MOV     #0,R1
007304  012737  000017  013230  2$:     MOV     #15.,TEMP1      ;LOAD TEMP
007312  012737  000004  011330          MOV     #4,CMPCNT       ;LOAD AVRG. COUNT
007320  016104  007414                  MOV     LADTB(R1),R4    ;GET POINTER
007324  004737  011172                  JSR     PC,CMPTEA       ;AVERAGE
007330  013722  013266                  MOV     AVRAGE,(R2)+    ;SAVE AVERAGE
```

```
1392  007334  005721                          TST     (R1)+           ;UPDATE POINTER
1393  007336  005300                          DEC     R0
1394  007340  001361                          BNE     2$              ;BR IF NOT DONE
1395
1396
1397  007342  032777  002000  171566          BIT     #SW10,@SWR      ;TEST BIT 10
1398  007350  001246                          BNE     RECVY2
1399  007352  104400                          TYPE
1400  007354  012505                          MES9                    ;TEXT 'CH.'
1401  007356  013746  013164                  MOV     KSTOR2,-(SP)
1402  007362  104402                          TYPOS
1403  007364     002     001                  .BYTE   2,1
1404  007366  012737  000002  007712          MOV     #2,SPACEX
1405  007374  004737  007662                  JSR     PC,XSPACE
1406  007400  004737  007714                  JSR     PC,XPRTAV               ;PRINT VALUES OF 2ND CH.
1407  007404  000630                          BR RECVY2               ;DO IT AGAIN
1408  007406  000000                  10$:    0
1409  007410  000000                  11$:    0
1410  007412  000000                  12$:    0
1411
1412  007414  016242                  LADTR:  ADTB0
1413  007416  016304                          ADTB1
1414  007420  016346                          ADTB2
1415  007422  016410                          ADTB3
1416  007424  016452                          ADTB4
1417  007426  016514                          ADTB5
1418  007430  016556                          ADTB6
1419  007432  016620                          ADTB7
```

```
                                ;;*****************************************************************
                                ;*TEST 14       LOAD DIFFERENT NUMBERS INTO DIFFERENT REG.
                                ;;*****************************************************************
007434  000004                  TST14:  SCOPE
007436  012737  000040  001164          MOV     #40,STIMES      ;;DO 40 ITERATIONS
007444  012737  007452  013152  RESTST: MOV     #20$,AVECTR
007452  104400                  20$:    TYPE
007454  011772                          MES2                            ;TYPE HEADER ABOUT TEST
007456  005037  007636                  CLR     71$
007462  000005                          RESET
007464  012737  000000  007634  11$:    MOV     #0,70$                  ;LOAD COUNTER
007472  000240                  10$:    NOP
007474  013700  001354          2$:     MOV     CSC,R0                  ;LOAD STARTING ADDRESS
007500  062700  000002                  ADD     #2,R0
007504  012701  007660                  MOV     #BUFNUM+20,R1           ;LOAD STARTING TABLE ADDRESS
007510  012702  000010                  MOV     #8.,R2                  ;LOAD COUNT
007514  014140                  1$:     MOV     -(R1),-(R0)             ;LOAD THE REG
007516  005302                          DEC     R2                      ;DONE ALL ?
007520  001375                          BNE     1$                      ;BR IF NOT
                                
007522  013700  001336                  MOV     ADCS,R0                 ;LOAD STARTING POINTER
007526  012701  007640                  MOV     #BUFNUM,R1              ;LOAD STARTING POINTER <EXPECTED>
007532  012702  000010                  MOV     #8.,R2                  ;LOAD # OF REG
                                
007536  011137  001124          3$:     MOV     (R1),$GODAT             ;READ REG
007542  011037  001126                  MOV     (R0),$BODAT             ;READ REG
007546  023737  001124  001126          CMP     $GODAT,$BODAT           ;COMPARE
007554  001403                          BEQ     4$              ;;BR IF EQUAL
007556  010037  007660                  MOV     R0,BUFADR               ;SAVE BUS ADDRESS
007562  104001                          ERROR   1               ;INCORRECT DATA, REG. WAS CHANGED IN ERROR
007564  022021                  4$:     CMP     (R0)+,(R1)+             ;UPDATE POINTERS
007566  005302                          DEC     R2                      ;DONE ALL REG ?
007570  001362                          BNE     3$                      ;BR IF NOT
007572  005077  171550                  CLR     @VCSTAT
007576  004737  011672                  JSR     PC,TSTFLG
007602  005337  007634                  DEC     70$                     ;DONE
007606  001331                          BNE     10$                     ;NO
007610  104400                          TYPE
007612  004521                          SENDMG
007614  005237  007636                  INC     71$
007620  013746  007636                  MOV     71$,-(SP)
007624  104404                          TYPDS
007626  104400  004516                  TYPE    ,$ENULL
007632  000714                          BR      11$                     ;LOOP BACK
007634  000000                  70$:    0
007636  000000                  71$:    0
                                
                                ;# TO BE LOADED INTO DIFFERENT REG
                                
007640  027560                  BUFNUM: 27560                           ;A TO D STATUS
007642  000000                          0                               ;A TO D BUFFER
007644  140736                          140736                          ;CLOCK STATUS
007646  000377                          377                             ;CLOCK PRESET
007650  007214                          7214                            ;VC STATUS
007652  001252                          1252                            ;VC X POS
007654  000525                          525                             ;VC Y POS
```

```
                                                377                             ;CLOCK COUNTER

007660  170400          BUFADR: 170400                                        ;BUS ADDRESS OF REG IN ERROR

                        ;*****************************************************************
                        ;SUBROUTINE TO ISSUE N SPACES
                        ;N IS ONE PLUS VALUE CONTAINED IN SPACEX
                        ;SPACEX IS CLEARED WITHIN THE SUBROUTINE, SO THAT A CALL ON
                        ;SPACE WITHOUT LOADING SPACEX ISSUES ONLY ONE SPACE

007662  105777  171260  XSPACE: TSTB    @STPS           ;WAIT FOR TTY READY
007666  100375                BPL     .-4
007670  012777  000240  171252        MOV     #240,@STPB      ;OUTPUT A SPACE
007676  005337  007712              DEC     SPACEX          ;DECREMENT COUNT
007702  003367                      BGT     XSPACE          ;LOOP IF NOT DONE
007704  005037  007712              CLR     SPACEX          ;RESET COUNT TO ZERO
007710  000207                      RTS     PC              ;RETURN
007712  000000          SPACEX: 0

                        ;SUBROUTINE TO TYPE OUT AVERAGES FOR THE RECOVERY TEST

007714  012737  016200  007724  XPRTAV: MOV     #ADBUFF,AVGTAB
007722  013746          XPTA1:  MOV     @(PC)+,-(SP)
007724  016200          AVGTAB: ADBUFF
007726  104402                  TYPOS
007730     004     001          .BYTE   4,1
007732  062737  000002  007724        ADD     #2,AVGTAB
007740  012737  000002  007712        MOV     #2,SPACEX
007746  004737  007662              JSR     PC,XSPACE
007752  005337  013234              DEC     TEMP3
007756  001361                      BNE     XPTA1
007760  000207                      RTS     PC


                        .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

007762          STYPDS:
007762  010046              MOV     R0,-(SP)        ;;PUSH R0 ON STACK
007764  010146              MOV     R1,-(SP)        ;;PUSH R1 ON STACK
007766  010246              MOV     R2,-(SP)        ;;PUSH R2 ON STACK
007770  010346              MOV     R3,-(SP)        ;;PUSH R3 ON STACK
007772  010546              MOV     R5,-(SP)        ;;PUSH R5 ON STACK
007774  012746  020200        MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
010000  016605  000020        MOV     20(SP),R5       ;;GET THE INPUT NUMBER
010004  100004                BPL     1$              ;;BR IF INPUT IS POS.
010006  005405                NEG     R5              ;;MAKE THE BINARY NUMBER POS.
```

```
1532  010010  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
1533  010016  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
1534  010020  012703  010176                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
1535  010024  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
1536  010030  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
1537  010032  016001  010166                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
1538  010036  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
1539  010040  002402                          BLT     4$              ;;BR IF DONE
1540  010042  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
1541  010044  000774                          BR      3$
1542  010046  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
1543  010050  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
1544  010052  001002                          BNE     5$              ;;FALL THROUGH IF 0
1545  010054  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
1546  010056  100407                          BMI     7$              ;;BR IF YES
1547  010060  106316                  5$:     ASLB    (SP)            ;;MSD?
1548  010062  103003                          BCC     6$              ;;BR IF NO
1549  010064  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
1550  010072  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
1551  010076  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
1552  010102  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
1553  010104  005720                          TST     (R0)+           ;;JUST INCREMENTING
1554  010106  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
1555  010112  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
1556  010114  003002                          BGT     8$              ;;GO TO EXIT
1557  010116  010502                          MOV     R5,R2           ;;GET THE LSD
1558  010120  000764                          BR      6$              ;;GO CHANGE TO ASCII
1559  010122  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
1560  010124  100003                          BPL     9$              ;;BR IF NO
1561  010126  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
1562  010134  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
1563  010136  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
1564  010140  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
1565  010142  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
1566  010144  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
1567  010146  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
1568  010150  104400  010176                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
1569  010154  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
1570  010162  012616                          MOV     (SP)+,(SP)
1571  010164  000002                          RTI                     ;;RETURN TO USER
1572  010166  023420                  $DTBL:  10000.
1573  010170  001750                          1000.
1574  010172  000144                          100.
1575  010174  000012                          10.
1576  010176  000004                  $DBLK:  .BLKW   4
```

```
1577                                    ;KEYBOARD SERVICE ROUTINE
1578
1579  010206  012704  010436          XTTYIN: MOV     #INBUF,R4       ;SETUP CHARACTER BUFFER
1580  010212  042777  000100  170722          BIC     #BIT6,@STKS
1581  010220  005037  013156                  CLR     CHRCNT          ;CLEAR CHARACTER COUNTER
1582  010224  005037  010436                  CLR     INBUF
1583  010230  005037  010440                  CLR     INBUF+2
1584  010234  105777  170702          11$:    TSTB    @STKS           ;CHARACTER READY?
1585  010240  100375                          BPL     11$             ;NO, WAIT IT OUT
1586  010242  017701  170676                  MOV     @STKB,R1        ;SAVE CHARACTER
1587  010246  042701  177640                  BIC     #177640,R1      ;STRIP PARITY BIT
1588  010252  120127  000060                  CMPB    R1,#60          ;IS IT A SPECIAL CHARACTER ?
1589  010256  100420                          BMI     14$             ;YES, TEST IT
1590  010260  122701  000137                  CMPB    #137,R1
1591  010264  100415                          BMI     14$
1592  010266  010124                  12$:    MOV     R1,(R4)+        ;SAVE CHARACTER
1593  010270  005237  013156                  INC     CHRCNT          ;INCREMENT THE CHARACTER COUNT.
1594  010274  022737  000006  013156          CMP     #6,CHRCNT
1595  010302  100451                          BMI     4$
1596  010304  105777  170636          13$:    TSTB    @STPS           ;ECHO CHARACTER
1597  010310  100375                          BPL     13$
1598  010312  110177  170632                  MOVB    R1,@STPB
1599  010316  000746                          BR      11$             ;WAIT FOR NEXT CHARACTER
1600                                    ;SUBROUTINE TO TEST FOR SPECIAL CHARACTERS :'^A','^C','^G','CR',',' OR 'RUBOUT'
1601
1602  010320  122701  000001          14$:    CMPB    #1,R1           ;CHAR,=:'^A' ?
1603  010324  001005                          BNE     1$              ;NO, NOT '^A'
1604  010326  104400                          TYPE                    ;ECHO '^A'
1605  010330  012351                          CNTRLA
1606  010332  005726                          TST     (SP)+           ;RESTORE SP
1607  010334  000177  002612                  JMP     @AVECTR         ;YES, EXIT VIA '^A' VECTOR ADDRESS.
1608  010340  122701  000003          1$:     CMPB    #3,R1           ;CHAR,=:'^C' ?
1609  010344  001003                          BNE     2$              ;NO,NOT '^C'
1610  010346  005726                          TST     (SP)+
1611  010350  000137  002076                  JMP     INIT2           ;YES, EXIT TO MONITOR
1612  010354  122701  000137          2$:     CMPB    #137,R1         ;CHAR, = 'RUBOUT' ?
1613  010360  001011                          BNE     3$              ;IGNORE CHAR, & EXIT
1614  010362  005737  013156                  TST     CHRCNT          ;IS RUBOUT LEGAL?
1615  010366  001722                          BEQ     11$             ;NO, IGNORE IT
1616  010370  005337  013156                  DEC     CHRCNT
1617  010374  012701  000134                  MOV     #134,R1         ;TYPE '\' TO INDICATE RUBOUUT
1618  010400  005744                          TST     -(R4)           ;POP OFF LAST CHARACTER
1619  010402  000740                          BR      13$             ;WAIT FOR NEXT CHARACTER
1620  010404  122701  000054          3$:     CMPB    #54,R1          ;TEST FOR ','
1621  010410  001726                          BEQ     12$             ;LEGAL CHAR,, SAVE IT
1622  010412  122701  000015                  CMPB    #15,R1          ;=TO 'CARRIAGE RETURN' TO TERMINATE?
1623  010416  001003                          BNE     4$              ;NO, CONTINUE
1624  010420  104400                          TYPE                    ;YES, TYPE 'CR-LF'
1625  010422  012356                          ACRLF
1626  010424  000207                          RTS     PC              ;EXIT
```

```
1627  010426  104405                  4$:     CKSWR                   ;TEST FOR CTRL G
1628  010430  104400                          TYPE                    ;OTHERWISE TYPE '?'
1629  010432  012365                          QMARK
1630  010434  000664                          BR      XTTYIN          ; WAIT FOR NEW ENTRY
1631  010436  000000                  INBUF:  0                       ;CHARACTER STORAGE BUFFER
1632
1633
1634          010454                          .=.+14
1635                                  ;SUBROUTINE WILL CONVERT 'N' WORDS (SEPARATED VIA COMMA'S)
1636                                  ;WHICH WERE STORED IN A TABLE VIA 'TTYIN' TO OCTAL AND STORE THEM.
1637
1638  010454  012704  010436          BCDBIN: MOV     #INBUF,R4       ;SETUP ASCII STORAGE TABLE
1639  010460  012703  010610                  MOV     #BCDTAB,R3      ;TABLE FOR STORAGE OF CONVERTED WORDS
1640  010464  005037  010612                  CLR     BCDTAB+2
1641  010470  005001                  BCDBN1: CLR     R1              ;REG. TO STORE RUNNING TOTAL
1642  010472  005002                          CLR     R2              ;TEMP. STORAGE FOR 'R1'
1643  010474  005737  013156          BCDBN2: TST     CHRCNT          ;END OF DATA?
1644  010500  003424                          BLE     BCDEND          ;YES, EXIT
1645  010502  005337  013156                  DEC     CHRCNT          ;DECREMENT CHARACTER COUNTER
1646  010506  122714  000054                  CMPB    #54,(R4)        ;IS CHARACTER = TO ','?
1647  010512  001417                          BEQ     BCDEND          ;YES, DECODE NEW WORD
1648  010514  121427  000060                  CMPB    (R4),#60
1649  010520  002425                          BLT     BCDERR          ;TEST FOR LEGAL NO.
1650  010522  021427  000067                  CMP     (R4),#67
1651  010526  003022                          BGT     BCDERR
1652  010530  042714  177770                  BIC     #177770,(R4)    ;STRIP NO.
1653  010534  012400                          MOV     (R4)+,R0        ;SAVE NO. IN R0.
1654  010536  010102                          MOV     R1,R2           ;SAVE CURRENT TOTAL
1655  010540  006301                          ASL     R1              ;*2
1656  010542  006301                          ASL     R1              ;*4
1657  010544  006301                          ASL     R1              ;*8
1658  010546  060001                          ADD     R0,R1           ;N=NEW NO.
1659  010550  000751                          BR      BCDBN2
1660  010552  020127  000077          BCDEND: CMP     R1,#77
1661  010556  003006                          BGT     BCDERR
1662  010560  005724                          TST     (R4)+           ;UPDATE BUFFER
1663  010562  010123                          MOV     R1,(R3)+        ;SAVE CONVERTED VALUE & SETUP TO SAVE NEXT
1664  010564  005737  013156                  TST     CHRCNT          ;FINISHED?
1665  010570  001337                          BNE     BCDBN1          ;NO, CONVERT NEXT WORD
1666  010572  000207                          RTS     PC              ;YES, EXIT
1667  010574  104400                  BCDERR: TYPE
1668  010576  012365                          QMARK                   ;TYPE '?'.
1669  010600  004737  010206                  JSR     PC,X"TYIN       ;TO BE TYPED ON QUESTIONABLE ENTRIES.
1670  010604  000137  010454                  JMP     BCDBIN
1671  010610  000000                  BCDTAB: 0                       ;OCTAL STORAGE TABLE
1672  010612  000000                          0
1673  010614  000000                          0
1674  010616  000000                          0
```

```
1675                                    ;SUBROUTINE TO TAKE 'N' CONVERSIONS AND STORE THEM IN AN A/D BUFFER. ROUTINE
1676                                    ;IS ENTERED WITH 'N' IN COUNT AND THE CH TO BE CONVERTED IN 'ADWORD'.
1677                                    ;ENTERING WITH DELAY = 0 CAUSES DELAY BETWEEN MUX LOAD AND START OF
1678                                    ;FIRST CONVERSION
1679
1680  010620  005077  167152           ADCNVT: CLR     @PSW
1681  010624  013737  013160  013230           MOV     COUNT,TEMP1     ;SET UP NO. OF CONVERSIONS TO BE TAKEN
1682  010632  012704  016200                   MOV     #ADBUFF,R4      ;SET UP BUFFER ADDRESS.
1683  010636  005737  013216                   TST     DELAY           ;CHECK IF DELAY SET
1684  010642  001017                           BNE     3$              ;BR IF INHIBIT DELAY
1685  010644  113777  013147  000100           MOVB    ADWRD2+1,@ADCS1         ;LOAD MUX
1686  010652  013737  010746  013220           MOV     CPTIME,DELAY1   ;SET UP TIMER
1687  010660  005337  013220           4$:     DEC     DELAY1          ;LOOP
1688  010664  001375                           BNE     4$
1689  010666  113777  013146  170442           MOVB    ADWRD2,@ADCS    ;LOAD LOW BYTE
1690  010674  000405                           BR      1$
1691  010676  000240                   2$:     NOP
1692  010700  000240                           NOP
1693  010702  013777  013146  170426   3$:     MOV     ADWRD2,@ADCS    ;LOAD CONTROL
1694                                                                   ;LOAD CH. & START CONVERT
1695  010710  105777  170422           1$:     TSTB    @ADCS           ;WAIT FOR DONE
1696  010714  100375                           BPL     1$              ;DELAY
1697  010716  017724  170416                   MOV     @ADDBR,(R4)+    ;SAVE DATA
1698  010722  005337  013230                   DEC     TEMP1           ;DECREMENT COUNTER
1699  010726  003363                           BGT     2$              ;IF NOT '0' TAKE NEXT CONVERSION
1700  010730  005037  013216                   CLR     DELAY
1701  010734  000207                           RTS     PC
1702
1703  010736  002000                   CPDLAY: 2000                    ;PDP-11/05
1704  010740  002400                           2400                    ;PDP-11/20
1705  010742  003500                           3500                    ;PDP-11/40
1706  010744  006000                           6000                    ;PDP-11/45
1707
1708  010746  002000                   CPTIME: 2000
1709  010750  000001                   CPTYPE: 1
1710  010752  170401                   ADCS1:  170401
1711
```

```
1712
1713                                    ;PRINT DECIMAL VALUE IN R2
1714
1715  010754  005077  167016          DECPRT: CLR     @PSW
1716  010760  012737  177774  011134          MOV     #-4,DIGCNT
1717  010766  012737  011142  011140          MOV     #DECPNT+2,DECPNT
1718  010774  012737  000240  011136          MOV     #240,ZERO
1719  011002  012737  177777  011132  TYPT1:  MOV     #-1,DIGIT
1720  011010  005237  011132          TYPT2:  INC     DIGIT
1721  011014  167702  000120                  SUB     @DECPNT,R2
1722  011020  100373                          BPL     TYPT2
1723  011022  067702  000112                  ADD     @DECPNT,R2
1724  011026  004737  011052                  JSR     PC,DECOUT
1725  011032  005237  011134                  INC     DIGCNT
1726  011036  001001                          BNE     TYPT3
1727  011040  000207                          RTS     PC
1728  011042  062737  000002  011140  TYPT3:  ADD     #2,DECPNT
1729  011050  000754                          BR      TYPT1
1730  011052  005737  011132          DECOUT: TST     DIGIT
1731  011056  001010                          BNE     DEC1
1732  011060  022737  177777  011134          CMP     #-1,DIGCNT
1733  011066  001404                          BEQ     DEC1
1734  011070  013737  011136  011132          MOV     ZERO,DIGIT
1735  011076  000406                          BR      DEC2
1736  011100  012737  000260  011136  DEC1:   MOV     #260,ZERO
1737  011106  052737  000260  011132          BIS     #260,DIGIT
1738  011114  105777  170026          DEC2:   TSTB    @STPS
1739  011120  100375                          BPL     .-4
1740  011122  013777  011132  170020          MOV     DIGIT,@STPB
1741  011130  000207                          RTS     7
```

```
1742  011132  000000                    DIGIT:  0
1743  011134  000000                    DIGCNT: 0
1744  011136  000240                    ZERO:   240
1745  011140  011142                    DECPNT: .+2
1746  011142  001750                            1000.
1747  011144  000144                            100.
1748  011146  000012                            10.
1749  011150  000001                            1.
1750
1751                                    ;COMPUTE THE RESULTS OF 512 CONVERSIONS AS HIGH, LOW AND AVERAGE
1752
1753  011152  012737  000777  013230  CMPTE:  MOV     #777,TEMP1      ;SET UP TO COMPARE '511' NUMBERS
1754  011160  012737  000011  011330          MOV     #11,CMPCNT      ;LOAD COUNTER
1755  011166  012704  016200          CMPTER: MOV     #ADBUFF,R4      ;LOAD STARTING ADDRESS
1756  011172  005037  013246          CMPTEA: CLR     HIORDV          ;CLR HI ORDER DIVIDEND
1757  011176  012437  013266                  MOV     (R4)+,AVRAGE    ;STORE 1ST VALUE AS AVERAGE
1758  011202  013737  013266  013250          MOV     AVRAGE,ADHIGH   ;HIGH
1759  011210  013737  013266  013252          MOV     AVRAGE,ADLOW    ;& LOW
1760  011216  012437  013232          GETDAT: MOV     (R4)+,TEMP2
1761  011222  023737  013232  013250          CMP     TEMP2,ADHIGH    ;IS NEW NO. GREATER THAN OLD NO. ?
1762  011230  003403                          BLE     TSLO            ;BRANCH IF NOT GREATER
1763  011232  013737  013232  013250          MOV     TEMP2,ADHIGH    ;OTHERWISE SAVE AS NEW HIGH
1764  011240  023737  013232  013252  TSLO:   CMP     TEMP2,ADLOW
1765  011246  003003                          BGT     TAGA
1766  011250  013737  013232  013252          MOV     TEMP2,ADLOW     ;OTHERWISE SAVE AS NEW LOW
1767  011256  063737  013232  013266  TAGA:   ADD     TEMP2,AVRAGE    ;ADD LOW ORDER
1768  011264  005537  013246                  ADC     HIORDV          ;ADD CARRY TO HI ORDER
1769  011270  005337  013230                  DEC     TEMP1
1770  011274  001350                          BNE     GETDAT          ;512 ADDITIONS?
1771  011276  013737  011330  013230          MOV     CMPCNT,TEMP1    ;YES, DIVIDE/512
1772  011304  006237  013246          AVGDAT: ASR     HIORDV
1773  011310  006037  013266                  ROR     AVRAGE          ;SHIFT CARRY BIT INTO LO ORDER
1774  011314  005337  013230                  DEC     TEMP1
1775  011320  001371                          BNE     AVGDAT          ;DONE?
1776  011322  005537  013266                  ADC     AVRAGE          ;YES, ADD REMAINDER TO LO ORDER
1777  011326  000207                          RTS     PC
1778
1779  011330  000011          CMPCNT: 11
```

```
1780                                        ;SUBROUTINE TO CALCULATE THE PLUS & MINUS 5 COUNT LIMITS FROM AN AVERAGE
1781
1782  011332  012737  000005  013230  CATORZ: MOV     #5,TEMP1
1783  011340  013737  013266  013232          MOV     AVRAGE,TEMP2    ;MOV AVER. TO WORK AREA
1784  011346  012703  013270                  MOV     #AVERP1,R3      ;SETUP DISTRIBUTION TABLE (POS.)
1785  011352  005237  013232          FILE1:  INC     TEMP2           ;A=A+1
1786  011356  013723  013232                  MOV     TEMP2,(R3)+     ;SAVE A+1
1787  011362  005337  013230                  DEC     TEMP1           ;SAVED '5' COUNTS?
1788  011366  001371                          BNE     FILE1           ;BRANCH IF NO
1789                                        ;SET UP TABLE OF AVG. -1 TO -5
1790  011370  012737  000005  013230          MOV     #5,TEMP1
1791  011376  013737  013266  013232          MOV     AVRAGE,TEMP2    ;MOV AVG. TO WORK AREA.
1792  011404  012703  013266                  MOV     #AVRAGE,R3      ;SET UP DISTRIBUTION TABLE NEG.
1793  011410  005337  013232          FILE2:  DEC     TEMP2           ;A=1-1
1794  011414  013743  013232                  MOV     TEMP2,-(R3)     ;SAVE 'A-1'
1795  011420  005337  013230                  DEC     TEMP1           ;SAVED '5' COUNTS?
1796  011424  001371                          BNE     FILE2           ;BRANCH IF NO
1797
1798                                        ;CATEGORIZE THE COUNT SPREAD AS '+6 & -6' COUNTS FROM THE AVERAGE
1799
1800  011426  012703  013302                  MOV     #ORLOW,R3       ;CLEAR COUNTS
1801  011432  005023          CATR1:  CLR     (R3)+
1802  011434  022703  013334                  CMP     #ORHIGH+2,R3    ;FINISHED?
1803  011440  001374                          BNE     CATR1           ;NO, CLEAR NEXT COUNTER
1804  011442  012737  001001  013230          MOV     #1001,TEMP1     ;COMPARE '512' COUNTS
1805  011450  012700  016200                  MOV     #ADBUFF,R0      ;SET UP A/D BUFFER
1806  011454  005337  013230          CATR2:  DEC     TEMP1
1807  011460  001437                          BEQ     CATR5           ;EXIT IF '0'
1808  011462  012037  013232                  MOV     (R0)+,TEMP2
1809  011466  023737  013300  013232          CMP     AVERP5,TEMP2
1810  011474  100423                          BMI     OVRHI
1811  011476  023737  013232  013254          CMP     TEMP2,AVERM5
1812  011504  100422                          BMI     OVRLO
1813  011506  005001                          CLR     R1
1814  011510  012702  013254                  MOV     #AVERM5,R2
1815  011514  022237  013232          CATR3:  CMP     (R2)+,TEMP2
1816  011520  001405                          BEQ     CATR4
1817  011522  005201                          INC     R1
1818  011524  022701  000013                  CMP     #13,R1
1819  011530  001371                          BNE     CATR3
1820  011532  000000                          HALT                    ;FATAL ERROR
1821  011534  006301          CATR4:  ASL     R1              ;MULTIPLY 'OFFSET' X2
1822  011536  005261  013304                  INC     MINUS5(R1)
1823  011542  000744                          BR      CATR2
1824  011544  005237  013332          OVRHI:  INC     ORHIGH
1825  011550  000741                          BR      CATR2
1826  011552  005237  013302          OVRLO:  INC     ORLOW
1827  011556  000736                          BR      CATR2
```

```
1828
1829                                    ;ADD THE COUNTS AND SAVE TOTAL IN SPREADS OF '1-4'
1830
1831  011560  013737  013316  013334  CATR5:  MOV     AVGCNT,XSPRD1
1832  011566  063737  013320  013334          ADD     PLUS1,XSPRD1
1833  011574  063737  013314  013334          ADD     MINUS1,XSPRD1   ;=TO NO. COUNTS AT SPREAD OF '1'
1834  011602  013737  013334  013336          MOV     XSPRD1,XSPRD2
1835  011610  063737  013322  013336          ADD     PLUS2,XSPRD2
1836  011616  063737  013312  013336          ADD     MINUS2,XSPRD2   ;=TO NO. COUNTS AT SPREAD OF '2'
1837  011624  013737  013336  013340          MOV     XSPRD2,XSPRD3
1838  011632  063737  013324  013340          ADD     PLUS3,XSPRD3
1839  011640  063737  013310  013340          ADD     MINUS3,XSPRD3   ;=TO NO. COUNTS AT SPREAD OF '3'
1840  011646  013737  013340  013342          MOV     XSPRD3,XSPRD4
1841  011654  063737  013326  013342          ADD     PLUS4,XSPRD4
1842  011662  063737  013306  013342          ADD     MINUS4,XSPRD4   ;=TO NO. COUNTS AT SPREAD OF '4'
1843  011670  000207                          RTS     PC              ;EXIT
1844
1845                                    ;SUBROUTINE TO TEST FOR THE KEYBOARD FLAG BEING SET
1846
1847  011672  105777  167244  TSTFLG: TSTB    @$TKS           ;FLAG SET?
1848  011676  100002                  BPL     1$              ;NO, EXIT
1849  011700  004737  010206                  JSR     PC,XTTYIN                       ;YES,INQUIRE
1850  011704  000207  1$:     RTS     PC
1851
```

```
 1852  011706  005015  040412  026522  TITLE:  .ASCIZ  <15><12><12>'AR-11 DIAGNOSTIC TEST II, (MAINDEC-11-DZARB-B)'<15><12>
 1853  011714  030461  042040  040511
 1854  011722  047107  051517  044524
 1855  011730  020103  042524  052123
 1856  011736  044440  026111  024040
 1857  011744  040515  047111  042504
 1858  011752  026503  030461  042055
 1859  011760  040532  041122  041055
 1860  011766  006451  000012
 1861  011772  005015  052123  052101  MES2:   .ASCIZ  <15><12>"STATIC REGISTER TEST"<15><12>
 1862  012000  041511  051040  043505
 1863  012006  051511  042524  020122
 1864  012014  042524  052123  005015
 1865  012022     000
 1866  012023     015  042412  040522  MES3:   .ASCIZ  <15><12>"ERASE RETURN FAILED TO SET READY"<15><12>
 1867  012030  042523  051040  052105
 1868  012036  051125  020116  040506
 1869  012044  046111  042105  052040
 1870  012052  020117  042523  020124
 1871  012060  042522  042101  006531
 1872  012066  000012
 1873  012070  005015  054524  042520  MES4:   .ASCII  <15><12>"TYPE LETTER ' ' TO RUN DESIRED TEST:"<15><12>
 1874  012076  046040  052105  042524
 1875  012104  020122  020047  020047
 1876  012112  047524  051040  047125
 1877  012120  042040  051505  051111
 1878  012126  042105  052040  051505
 1879  012134  035124  005015
 1880  012140  040447  036447  052123          .ASCII  "'A'=STATIC REGISTER TEST"<15><12>
 1881  012146  052101  041511  051040
 1882  012154  043505  051511  042524
 1883  012162  020122  042524  052123
 1884  012170  005015
 1885  012172  041047  036447  041523          .ASCII  "'B'=SCOPE ADJUSTMENT TESTS"<15><12>
 1886  012200  050117  020105  042101
 1887  012206  052512  052123  042515
 1888  012214  052116  052040  051505
 1889  012222  051524  005015
 1890  012226  041447  036447  020101          .ASCII  "'C'=A TO D CALIBRATION TEST"<15><12>
 1891  012234  047524  042040  041440
 1892  012242  046101  041111  040522
 1893  012250  044524  047117  052040
 1894  012256  051505  006524     012
 1895  012263     047  023504  040475          .ASCII  "'D'=A TO D REPEATABILITY"<15><12>
 1896  012270  052040  020117  020104
 1897  012276  042522  042520  052101
 1898  012304  041101  046111  052111
 1899  012312  006531     012
 1900  012315     047  023505  040475          .ASCIZ  "'E'=A TO D RECOVERY"<15><12>
 1901  012322  052040  020117  020104
 1902  012330  042522  047503  042526
 1903  012336  054522  005015     000
 1904  012343     136  006503  027012  CNTRLC: .ASCIZ  ^C<15><12><56>
 1905  012350     000
 1906  012351     136  006501  000012  CNTRLA: .ASCIZ  ^A<15><12>
 1907  012356     015     012     000  ACRLF:  .BYTE   15,12,0
```

```
1908  012361  015     012     056     DOT:    .BYTE   15,12,56,0
1909  012364  000
1910  012365  077     000040          QMARK:  .ASCIZ  '? '
1911  012370  005015  041523  050117  MES6:   .ASCIZ  <15><12>'SCOPE ADJUSTMENT TEST'
1912  012376  020105  042101  052512
1913  012404  052123  042515  052116
1914  012412  052040  051505  000124
1915  012420  005015  020101  047524  MES7:   .ASCIZ  <15><12>'A TO D CALIBRATION TEST'<15><12>
1916  012426  042040  041440  046101
1917  012434  041111  040522  044524
1918  012442  047117  052040  051505
1919  012450  006524  000012
1920  012454  005015  020101  047524  MES8:   .ASCIZ  <15><12>'A TO D RECOVERY TEST'<15><12>
1921  012462  042040  051040  041505
1922  012470  053117  051105  020131
1923  012476  042524  052123  005015
1924  012504  000
1925  012505  015     041412  020110  MES9:   .ASCIZ  <15><12>'CH '
1926  012512  000
1927  012513  015     023412  023505  MES10:  .ASCIZ  <15><12>"'E'XT. OR 'I'NT. OR 'C'LOCK. SYNC? "
1928  012520  052130  020056  051117
1929  012526  023440  023511  052116
1930  012534  020056  051117  023440
1931  012542  023503  047514  045503
1932  012550  020056  054523  041516
1933  012556  020077    000
1934  012561  015     040412  052040  MES13:  .ASCIZ  <15><12>'A TO D REPEATABILITY TEST'<15><12>
1935  012566  020117  020104  042522
1936  012574  042520  052101  041101
1937  012602  046111  052111  020131
1938  012610  042524  052123  005015
1939  012616  000
1940  012617  015     005012  044103  MES14:  .ASCIZ  <15><12><12>'CH.(S)? '
1941  012624  024056  024523  020077
1942  012632  000
1943  012633  015     051412  051127  MES15:  .ASCIZ  <15><12>'SWR8 AND 0 THRU 2 CONTROL PATTERN'<15><12>
1944  012640  020070  047101  020104
1945  012646  020060  044124  052522
1946  012654  031040  041440  047117
1947  012662  051124  046117  050040
1948  012670  052101  042524  047122
1949  012676  005015    000
1950  012701  103     052517  052116  MES16:  .ASCIZ  'COUNT SPREAD? '
1951  012706  051440  051120  040505
1952  012714  037504  000040
1953  012720  005015  044103  020056  MES19:  .ASCIZ  <15><12>'CH. LO   AV   HI'
1954  012726  047514  020040  040440
1955  012734  020126  020040  044510
1956  012742  000
1957  012743  015     020012  046040  MES20:  .ASCIZ  <15><12>'  LO  -5  -4  -3  -2  -1  AV  +1  +2  +3  +4  +5  HI'
1958  012750  020117  026440  020065
1959  012756  026440  020064  026440
1960  012764  020063  026440  020062
1961  012772  026440  020061  040440
1962  013000  020126  025440  020061
1963  013006  025440  020062  025440
```

```
 1964  013014  020063  025440  020064
 1965  013022  025440  020065  044040
 1966  013030  000111
 1967  013032  042522  044507  052123  EM1:    .ASCIZ  /REGISTER CONTENTS CHANGED IN ERROR/
 1968  013040  051105  041440  047117
 1969  013046  042524  052116  020123
 1970  013054  044103  047101  042507
 1971  013062  020104  047111  042440
 1972  013070  051122  051117     000
 1973  013075     105  051122  041520  DM1:    .ASCIZ  /ERRPC   BUFADR   EXPECT    BAD/
 1974  013102  020040  041040  043125
 1975  013110  042101  020122  020040
 1976  013116  054105  042520  052103
 1977  013124  020040  020040  040502
 1978  013132  000104
 1979                                  .EVEN
 1980  013134  001116  007660  001124  DT1:    $ERRPC,BUFADR,$GDDAT,$BDDAT,0
 1981  013142  001126  000000
 1982  013146  000000                  ADWRD2: 0                       ;LOW BYTE OF 'ADWORD'
 1983  013150  000000                  PRINT1: 0
 1984  013152  002072                  AVECTR: INITA                   ;'^A' VECTOR ADDRESS
 1985  013154  000000                  PROC:   0                       ;TEMP STORAGE FOR 'PSW'
 1986  013156  000000                  CHRCNT: 0                       ;TEMP STORAGE
 1987  013160  000000                  COUNT:  0                       ;TEMP STORAGE
 1988  013162  000000                  KSTOR1: 0                       ;PERMANENT STORAGE
 1989  013164  000000                  KSTOR2: 0                       ;PERMANENT STORAGE
 1990  013166  000000                  KSTOR3: 0                       ;PERMANENT STORAGE
 1991  013170  000000                  KSTOR4: 0                       ;PERMANENT STORAGE
 1992  013172  000000                  KSTOR5: 0
 1993  013174  000000                  KSTR11: 0
 1994  013176  000000                  KSTR12: 0
 1995  013200  000000                  LOW:    0
 1996  013202  000000                  HIGH:   0
 1997  013204  000010                  INCR:   10
 1998  013206  000000                  TIMSV:  0
 1999  013210  000000                  TICKS:  0
 2000  013212  000000                  STCHAN: 0
 2001  013214  000000                  FIRST:  0
 2002  013216  000000                  DELAY:  0
 2003  013220  000000                  DELAY1: 0
 2004  013222  000000                  USECLK: 0
 2005  013224  000000                  OPS1:   0
 2006  013226  000000                  TEMP:   0
 2007  013230  000000                  TEMP1:  0                       ;TEMPORARY STORAGE
 2008  013232  000000                  TEMP2:  0                       ;TEMPORARY STORAGE
 2009  013234  000000                  TEMP3:  0                       ;TEMPORARY STORAGE
 2010  013236  000000                  BRLEV1: 0
 2011  013240  000000                  BRLEV2: 0
 2012  013242  000000                  BRLEV3: 0
 2013  013244  000000                  MESPRT: 0
 2014  013246  000000                  MIORDV: 0
 2015  013250  000000                  ADHIGH: 0
 2016  013252  000000                  ADLOW:  0
 2017  013254  000000                  AVERM5: 0
 2018  013256  000000                  AVERM4: 0
 2019  013260  000000                  AVERM3: 0
```

```
2023  013262  000000                  AVERM2: 0
2021  013264  000000                  AVERM1: 0
2022  013266  000000                  AVRAGE: 0
2023  013270  000000                  AVERP1: 0
2024  013272  000000                  AVERP2: 0
2025  013274  000000                  AVERP3: 0
2026  013276  000000                  AVERP4: 0
2027  013300  000000                  AVERP5: 0
2028  013302  000000                  ORLOW:  0
2029  013304  000000                  MINUS5: 0
2030  013306  000000                  MINUS4: 0
2031  013310  000000                  MINUS3: 0
2032  013312  000000                  MINUS2: 0
2033  013314  000000                  MINUS1: 0
2034  013316  000000                  AVGCNT: 0
2035  013320  000000                  PLUS1:  0
2036  013322  000000                  PLUS2:  0
2037  013324  000000                  PLUS3:  0
2038  013326  000000                  PLUS4:  0
2039  013330  000000                  PLUS5:  0
2040  013332  000000                  ORHIGH: 0
2041  013334  000000                  XSPRD1: 0
2042  013336  000000                  XSPRD2: 0
2043  013340  000000                  XSPRD3: 0
2044  013342  000000                  XSPRD4: 0
2045
2046                                  .SBTTL  SCOPE HANDLER ROUTINE
2047
2048                                  ;;**********************************************************************
2049                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS, IT WILL INCREMENT
2050                                  ;*AND LOAD THE TEST NUMBER(STSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
2051                                  ;*AND LOAD THE ERROR FLAG (SERFLG) INTO DISPLAY<15:08>
2052                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2053                                  ;*SW14=1         LOOP ON TEST
2054                                  ;*SW11=1         INHIBIT ITERATIONS
2055                                  ;*SW09=1         LOOP ON ERROR
2056                                  ;*SW08=1         LOOP ON TEST IN SWR<7:0>
2057                                  ;*CALL
2058                                  ;*      SCOPE           ;;SCOPE=TST
2059
2060  013344                          SSCOPE:
2061  013344  104405                         CKSWR
2062  013346  032777  040000  165562  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
2063  013354  001114                         BNE     SOVER           ;;YES IF SW14=1
2064                                  ;#####START OF CODE FOR THE XOR TESTER#####
2065  013356  000416                  SXTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
2066                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
2067  013360  013746  000004                 MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
2068  013364  012737  013404  000004         MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
2069  013372  005737  177060                 TST     @#177060        ;;TIME OUT ON XOR?
2070  013376  012637  000004                 MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
2071  013402  000443                         BR      $SVLAD          ;;GO TO THE NEXT TEST
2072  013404  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
2073  013406  012637  000004                 MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
2074  013412  000423                         BR      7$              ;;LOOP ON THE PRESENT TEST
2075  013414                          6$:;#####END OF CODE FOR THE XOR TESTER#####
```

```
2076  013414  032777  000400  165514          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
2077  013422  001404                          BEQ     2$              ;;BR IF NO
2078  013424  127737  165506  001102          CMPB    @SWR,STSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
2079  013432  001465                          BEQ     $OVER           ;;BR IF YES
2080  013434  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
2081  013440  001421                          BEQ     3$              ;;BR IF NO
2082  013442  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
2083  013450  101015                          BHI     3$              ;;BR IF NO
2084  013452  032777  001000  165456          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
2085  013460  001404                          BEQ     4$              ;;BR IF NO
2086  013462  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
2087  013470  000446                          BR      $OVER
2088  013472  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
2089  013476  005037  001164                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
2090  013502  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
2091  013504  032777  004000  165424  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
2092  013512  001011                          BNE     1$              ;;BR IF YES
2093  013514  005737  001206                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
2094  013520  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
2095  013522  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
2096  013526  023737  001164  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
2097  013534  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
2098  013536  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
2099  013544  013737  013622  001164          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
2100  013552  105237  001102          $SVLAD: INCB    STSTNM          ;;COUNT TEST NUMBERS
2101  013556  113737  001102  001204          MOVB    STSTNM,STESTN   ;;SET TEST NUMBER IN APT MAILBOX
2102  013564  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
2103  013570  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
2104  013574  005037  001166                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
2105  013600  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
2106  013606  013777  001102  165324  $OVER:  MOV     STSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
2107  013614  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
2108  013620  000002                          RTI                     ;;FIXES PS
2109  013622  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
2110
2111                                  .SBTTL  ERROR HANDLER ROUTINE
2112
2113                                  ;;**************************************************************
2114                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2115                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2116                                  ;*AND GO TO $ERRTYP ON ERROR
2117                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2118                                  ;*SW15=1         HALT ON ERROR
2119                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
2120                                  ;*SW10=1         BELL ON ERROR
2121                                  ;*SW09=1         LOOP ON ERROR
2122                                  ;*CALL
2123                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
2124
2125  013624                          $ERROR:
2126  013624  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
2127  013630  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
2128  013632  013777  001102  165300          MOV     STSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
2129  013640  032777  002000  165270          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
2130  013646  001402                          BEQ     1$              ;;NO = SKIP
2131  013650  104400  001170                  TYPE    ,$BELL          ;;RING BELL
```

```
 2132  013654  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
 2133  013660  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
 2134  013664  162737  000002  001116          SUB     #2,$ERRPC
 2135  013672  117737  165220  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
 2136  013700  032777  020000  165230          BIT     #BIT13,@$SWR    ;;SKIP TYPEOUT IF SET
 2137  013706  001004                          BNE     20$             ;;SKIP TYPEOUTS
 2138  013710  004737  014006                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
 2139  013714  104400  001175                  TYPE    ,$CRLF
 2140  013720                          20$:
 2141  013720  122737  000001  001220          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
 2142  013726  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
 2143  013730  113737  001114  013742          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
 2144  013736  004737  015704                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
 2145  013742     000                  21$:    .BYTE   0
 2146  013743     000                          .BYTE   0
 2147  013744  000777                  22$:    BR      22$             ;;APT ERROR LOOP
 2148  013746  005777  165164          2$:     TST     @$SWR           ;;HALT ON ERROR
 2149  013752  100001                          BPL     3$              ;;SKIP IF CONTINUE
 2150  013754  000000                          HALT                    ;;HALT ON ERROR!
 2151  013756  032777  001000  165152  3$:     BIT     #BIT09,@$SWR    ;;LOOP ON ERROR SWITCH SET?
 2152  013764  001402                          BEQ     4$              ;;BR IF NO
 2153  013766  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
 2154  013772  005737  001166          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
 2155  013776  001402                          BEQ     5$              ;;BR IF NONE
 2156  014000  013716  001166                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
 2157  014004                          5$:
 2158  014004  000002                          RTI                     ;;RETURN
 2159
 2160                                  .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
 2161
 2162                                  ;;**************************************************************************
 2163                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 2164                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
 2165                                  ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
 2166
 2167  014006                          $ERRTYP:
 2168  014006  104400  001175                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 2169  014012  010046                          MOV     R0,-(SP)        ;;SAVE R0
 2170  014014  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
 2171  014016  153700  001114                  BISB    @$ITEMB,R0
 2172  014022  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
 2173                                                                  ;;TYPE THE PC OF THE ERROR
 2174  014024  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
 2175                                                                  ;;ERROR ADDRESS
 2176  014030  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 2177  014032  000426                          BR      6$              ;;GET OUT
 2178  014034  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
 2179  014036  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
 2180  014040  006300                          ASL     R0
 2181  014042  006300                          ASL     R0
 2182  014044  062700  001324                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
 2183  014050  012037  014060                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
 2184  014054  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
 2185  014056  104400                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
 2186  014060  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
 2187  014062  104400  001175                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
```

```
2188  014066  012037  014076          3$:   MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
2189  014072  001404                        BEQ     5$              ;;SKIP TYPEOUT IF 0
2190  014074  104400                        TYPE                    ;;TYPE THE "DATA HEADER"
2191  014076  000000                  4$:   .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
2192  014100  104400  001175                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2193  014104  011000                  5$:   MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
2194  014106  001004                        BNE     7$              ;;GO TYPE THE DATA
2195  014110  012600                  6$:   MOV     (SP)+,R0        ;;RESTORE R0
2196  014112  104400  001175                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2197  014116  000207                        RTS     PC              ;;RETURN
2198  014120                          7$:
2199  014120  013046                        MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
2200  014122  104401                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2201  014124  005710                        TST     (R0)            ;;IS THERE ANOTHER NUMBER?
2202  014126  001770                        BEQ     6$              ;;BR IF NO
2203  014130  104400  014136                TYPE    ,8$             ;;TYPE TWO(2) SPACES
2204  014134  000771                        BR      7$              ;;LOOP
2205  014136  020040     000          8$:   .ASCIZ  /  /            ;;TWO(2) SPACES
2206          014142                        .EVEN
2207
2208                                  .SBTTL  POWER DOWN AND UP ROUTINES
2209
2210                                  ;;**********************************************************************
2211                                  ;POWER DOWN ROUTINE
2212  014142  012737  014306  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
2213  014150  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2214  014156  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2215  014160  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2216  014162  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2217  014164  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2218  014166  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2219  014170  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2220  014172  017746  164740                MOV     @$WR,-(SP)      ;;PUSH @$WR ON STACK
2221  014176  010637  014312                MOV     SP,$SAVR6       ;;SAVE SP
2222  014202  012737  014214  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
2223  014210  000000                        HALT
2224  014212  000776                        BR      .-2             ;;HANG UP
2225
2226                                  ;;**********************************************************************
2227                                  ;POWER UP ROUTINE
2228  014214  012737  014306  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
2229  014222  013706  014312                MOV     $SAVR6,SP       ;;GET SP
2230  014226  005037  014312                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
2231  014232  005237  014312          1$:   INC     $SAVR6          ;;WAIT FOR THE INC
2232  014236  001375                        BNE     1$              ;;OF  WORD
2233  014240  012677  164672                MOV     (SP)+,@$WR      ;;POP STACK INTO @$WR
2234  014244  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
2235  014246  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
2236  014250  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
2237  014252  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
2238  014254  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
2239  014256  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
2240  014260  012737  014142  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
2241  014266  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
2242  014274  104400                        TYPE                    ;;REPORT THE POWER FAILURE
2243  014276  014314                  $PWRMG: .WORD   PWRMSG           ;;POWER FAIL MESSAGE POINTER
```

```
 2244  014300  012716                          MOV     (PC)+,(SP)      ;;RESTART AT BEGIN
 2245  014302  001356                SPWRAD: .WORD   BEGIN           ;;RESTART ADDRESS
 2246  014304  000002                        RTI
 2247  014306  000000                SILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
 2248  014310  000776                        BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
 2249  014312  000000                SSAVR6: 0                       ;;PUT THE SP HERE
 2250  014314  005015  042522  052123 PWRMSG: .ASCIZ  <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12><12>
 2251  014322  051101  044524  043516
 2252  014330  040440  052106  051105
 2253  014336  040440  050040  053517
 2254  014344  051105  043040  044501
 2255  014352  052514  042522  005015
 2256  014360  000012
 2257                                  .EVEN
 2258
 2259
 2260                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 2261
 2262                                  ;;**********************************************************************
 2263                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 2264                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 2265                                  ;*STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 2266                                  ;*CALL:
 2267                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 2268                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
 2269                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 2270                                  ;*      .BYTE   M               ;;M=1 OR 0
 2271                                  ;*                                      ;;1=TYPE LEADING ZEROS
 2272                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
 2273                                  ;*
 2274                                  ;*STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 2275                                  ;*STYPOS OR STYPOC
 2276                                  ;*CALL:
 2277                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 2278                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
 2279                                  ;*
 2280                                  ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 2281                                  ;*CALL:
 2282                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 2283                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
 2284
 2285  014362  017646  000000          STYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
 2286  014366  116637  000001  014605          MOVB    1(SP),$0FILL    ;;LOAD ZERO FILL SWITCH
 2287  014374  112637  014607                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
 2288  014400  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
 2289  014404  000406                          BR      STYPON
 2290  014406  112737  000001  014605  STYPOC: MOVB    #1,$0FILL       ;;SET THE ZERO FILL SWITCH
 2291  014414  112737  000006  014607          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
 2292  014422  112737  000005  014604  STYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
 2293  014430  010346                          MOV     R3,-(SP)        ;;SAVE R3
 2294  014432  010446                          MOV     R4,-(SP)        ;;SAVE R4
 2295  014434  010546                          MOV     R5,-(SP)        ;;SAVE R5
 2296  014436  113704  014607                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
 2297  014442  005404                          NEG     R4
 2298  014444  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
 2299  014450  110437  014606                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
```

```
2300  014454  113704  014605              MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
2301  014460  016605  000012              MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
2302  014464  005003                      CLR     R3              ;;CLEAR THE OUTPUT WORD
2303  014466  006105              1$:     ROL     R5              ;;ROTATE MSB INTO "C"
2304  014470  000404                      BR      3$              ;;GO DO MSB
2305  014472  006105              2$:     ROL     R5              ;;FORM THIS DIGIT
2306  014474  006105                      ROL     R5
2307  014476  006105                      ROL     R5
2308  014500  010503                      MOV     R5,R3
2309  014502  006103              3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
2310  014504  105337  014606              DECB    $OMODE          ;;TYPE THIS DIGIT?
2311  014510  100016                      BPL     7$              ;;BR IF NO
2312  014512  042703  177770              BIC     #177770,R3      ;;GET RID OF JUNK
2313  014516  001002                      BNE     4$              ;;TEST FOR 0
2314  014520  005704                      TST     R4              ;;SUPPRESS THIS 0?
2315  014522  001403                      BEQ     5$              ;;BR IF YES
2316  014524  005204              4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2317  014526  052703  000060              BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2318  014532  052703  000040      5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2319  014536  110337  014602              MOVB    R3,8$           ;;SAVE FOR TYPING
2320  014542  104400  014602              TYPE    ,8$             ;;GO TYPE THIS DIGIT
2321  014546  105337  014604      7$:     DECB    $OCNT           ;;COUNT BY 1
2322  014552  003347                      BGT     2$              ;;BR IF MORE TO DO
2323  014554  002402                      BLT     6$              ;;BR IF DONE
2324  014556  005204                      INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2325  014560  000744                      BR      2$              ;;GO DO THE LAST DIGIT
2326  014562  012605              6$:     MOV     (SP)+,R5        ;;RESTORE R5
2327  014564  012604                      MOV     (SP)+,R4        ;;RESTORE R4
2328  014566  012603                      MOV     (SP)+,R3        ;;RESTORE R3
2329  014570  016666  000002  000004      MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2330  014576  012616                      MOV     (SP)+,(SP)
2331  014600  000002                      RTI                     ;;RETURN
2332  014602     000      8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2333  014603     000              .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2334  014604     000      $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2335  014605     000      $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2336  014606  000000      $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2337
2338                      .SBTTL  TYPE ROUTINE
2339
2340                      ;;*************************************************************************
2341                      ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
2342                      ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
2343                      ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
2344                      ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
2345                      ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
2346                      ;*
2347                      ;*CALL:
2348                      ;*1) USING A TRAP INSTRUCTION
2349                      ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2350                      ;*OR
2351                      ;*      TYPE
2352                      ;*      MESADR
2353                      ;*
2354
2355  014610  105737  001155      $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
```

```
2356  014614  100002                          APL     /1$             ;;BR IF YES
2357  014616  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
2358  014620  000430                          BR      3$              ;;LEAVE
2359  014622  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
2360  014624  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
2361  014630  122737  000001  001220          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
2362  014636  001011                          BNE     6$              ;;NO,GO CHECK FOR APT CONSOLE
2363  014640  132737  000100  001221          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
2364  014646  001405                          BEQ     6$              ;;NO,GO CHECK FOR CONSOLE
2365  014650  010037  014660                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
2366  014654  004737  015674                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
2367  014660  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
2368  014662  132737  000040  001221  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
2369  014670  001003                          BNE     6$              ;;YES,SKIP TYPE OUT
2370  014672  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
2371  014674  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
2372  014676  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
2373  014700  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
2374  014702  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
2375  014706  000002                          RTI                     ;;RETURN
2376  014710  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
2377  014714  001430                          BEQ     8$
2378  014716  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
2379  014722  001006                          BNE     5$
2380  014724  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
2381  014726  104408                          TYPE                    ;;TYPE A CR AND LF
2382  014730  001175                          $CRLF
2383  014732  105037  015066                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
2384  014736  000755                          BR      2$              ;;GET NEXT CHARACTER
2385  014740  004737  015022          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
2386  014744  123726  001154          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
2387  014750  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
2388  014752  013746  001152                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
2389                                          ;;AND THE NULL CHAR.
2390  014756  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
2391  014762  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2392  014764  004737  015022                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
2393  014770  105337  015066                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2394  014774  000770                          BR      7$              ;;LOOP
2395
2396                                  ;HORIZONTAL TAB PROCESSOR
2397
2398  014776  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2399  015002  004737  015022          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
2400  015006  132737  000007  015066,         BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2401  015014  001372                          BNE     9$              ;;TAB STOP
2402  015016  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
2403  015020  000724                          BR      2$              ;;GET NEXT CHARACTER
2404  015022  105777  164120          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2405  015026  100375                          BPL     $TYPEC
2406  015030  116677  000002  164112          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2407  015036  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2408  015044  001003                          BNE     1$              ;;BRANCH IF NO
2409  015046  105037  015066                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2410  015052  000406                          BR      $TYPEX          ;;EXIT
2411  015054  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
```

```
2412  015062  001402                        BEQ     STYPEX          ;;BRANCH IF YES
2413  015064  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
2414  015066  000000                SCHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
2415  015070  000207                STYPEX: RTS     PC
2416
2417
2418                                  .SBTTL  TTY INPUT ROUTINE
2419
2420                                  ;;**************************************************************
2421                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
2422                                  ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
2423                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
2424                                  ;*WHEN OPERATING IN TTY FLAG MODE.
2425  015072  022737  000176  001136  SCKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
2426  015100  001073                        BNE     14$             ;;BRANCH IF NO
2427  015102  105777  164034                TSTB    @STKS           ;;CHAR THERE?
2428  015106  100070                        BPL     14$             ;;IF NO, DON'T WAIT AROUND
2429  015110  117746  164030          2$:   MOVB    @STKB,-(SP)     ;;SAVE THE CHAR
2430  015114  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
2431  015120  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
2432  015124  001061                        BNE     14$             ;;NO, RETURN TO USER
2433  015126  104400  015535                TYPE    ,SCNTLG         ;;YES, ECHO CONTROL G
2434
2435  015132  104400  015542          6$:   TYPE    ,SMSWR          ;;TYPE CURRENT CONTENTS
2436  015136  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
2437  015142  104401                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2438  015144  104400  015553                TYPE    ,SMNEW          ;;PROMPT FOR NEW SWR
2439  015150  005046                        CLR     -(SP)           ;;CLEAR COUNTER
2440  015152  005046                        CLR     -(SP)           ;;THE NEW SWR
2441  015154  104406                7$:   RDCHR                   ;;GET NEXT CHAR
2442
2443  015156  022716  000025          8$:   CMP     #25,(SP)        ;;IS IT A CONTROL U?
2444  015162  001005                        BNE     9$              ;;BRANCH IF NO
2445  015164  104400  015530                TYPE    ,SCNTLU         ;;YES, ECHO IT
2446  015170  062706  000006                ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
2447  015174  000756                        BR      6$              ;;LET'S TRY IT AGAIN
2448
2449  015176  022716  000015          9$:   CMP     #15,(SP)        ;;IS IT A <CR>?
2450  015202  001011                        BNE     11$             ;;BRANCH IF NO
2451  015204  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
2452  015210  001403                        BEQ     10$             ;;BRANCH IF YES
2453  015212  016677  000002  163716        MOV     2(SP),@SWR      ;;SAVE NEW SWR
2454  015220  062706  000006          10$:  ADD     #6,SP           ;;CLEAR UP STACK
2455  015224  000417                        BR      13$             ;;RETURN TO USER
2456  015226  022716  000012          11$:  CMP     #12,(SP)        ;;IS IT A <LF>
2457  015232  001017                        BNE     15$             ;;BRANCH IF NO
2458  015234  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
2459  015240  001403                        BEQ     12$             ;;YES
2460  015242  016677  000002  163666        MOV     2(SP),@SWR      ;;SAVE NEW SWR
2461  015250  062706  000006          12$:  ADD     #6,SP           ;;CLEAR UP STACK
2462  015254  013716  000046                MOV     @#46,(SP)       ;;GET RESTART
2463  015260  062716  000010                ADD     #10,(SP)        ;;ADDRESS
2464  015264  104400  001175          13$:  TYPE    ,SCRLF          ;;ECHO <CR> AND <LF>
2465  015270  000002                14$:  RTI                     ;;RETURN
2466  015272  004737  015022          15$:  JSR     PC,STYPEC       ;;ECHO CHAR
2467  015276  042726  177770                BIC     #177770,(SP)+   ;;RESTRICT TO 0-7
```

```
2468  015302  005766  000002          TST     2(SP)           ;;IS THIS THE FIRST CHAR
2469  015306  001403                  BEQ     16$             ;;BRANCH IF YES
2470  015310  006316                  ASL     (SP)            ;;NO, SHIFT PRESENT
2471  015312  006316                  ASL     (SP)            ;;    CHAR OVER TO MAKE
2472  015314  006316                  ASL     (SP)            ;;    ROOM FOR NEW ONE.
2473  015316  005266  000002  16$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
2474  015322  056616  177776          BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
2475  015326  000712                  BR      7$              ;;GET THE NEXT ONE
2476                                ;;*****************************************************************
2477                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2478                                ;*CALL:
2479                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
2480                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
2481                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
2482                                ;
2483
2484  015330  011646                SRDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
2485  015332  016666  000004  000002        MOV     4(SP),2(SP)     ;;SAVE THE PS
2486  015340  105777  163576        1$:     TSTB    @$TKS           ;;WAIT FOR
2487  015344  100375                        BPL     1$              ;;A CHARACTER
2488  015346  117766  163572  000004        MOVB    @$TKB,4(SP)     ;;READ THE TTY
2489  015354  042766  177600  000004        BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
2490  015362  026627  000004  000140        CMP     4(SP),#140      ;;IS IT UPPER CASE?
2491  015370  002407                        BLT     2$              ;;BRANCH IF YES
2492  015372  026627  000004  000175        CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
2493  015400  003003                        BGT     2$              ;;BRANCH IF YES
2494  015402  042766  000040  000004        BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
2495  015410  000002                2$:     RTI                     ;;GO BACK TO USER
2496                                ;;*****************************************************************
2497                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2498                                ;*CALL:
2499                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
2500                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2501                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2502
2503  015412  010346                SRDLIN: MOV     R3,-(SP)        ;;SAVE R3
2504  015414  012703  015520        1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
2505  015420  022703  015530        2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
2506  015424  101405                        BLOS    4$              ;;BR IF YES
2507  015426  104406                        RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
2508  015430  112613                        MOVB    (SP)+,(R3)      ;;GET CHARACTER
2509  015432  122713  000177        10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
2510  015436  001003                        BNE     3$              ;;SKIP IF NOT
2511  015440  104400  001174        4$:     TYPE    ,$QUES          ;;TYPE A '?'
2512  015444  000763                        BR      1$              ;;CLEAR THE BUFFER AND LOOP
2513  015446  111337  015516        3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2514  015452  104400  015516                TYPE    ,9$
2515  015456  122723  000015                CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2516  015462  001356                        BNE     2$              ;;LOOP IF NOT RETURN
2517  015464  105063  177777                CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2518  015470  104400  001176                TYPE    ,$LF            ;;TYPE A LINE FEED
2519  015474  012603                        MOV     (SP)+,R3        ;;RESTORE R3
2520  015476  011646                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2521  015500  016666  000004  000002        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2522  015506  012766  015520  000004        MOV     #$TTYIN,4(SP)
2523  015514  000002                        RTI                     ;;RETURN
```

```
2524  015516  000                     98:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2525  015517  000                             .BYTE   0               ;;TERMINATOR
2526  015520  000010                  STTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
2527  015530  052536  005015    000   $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
2528  015535    136   006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
2529  015542  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
2530  015550  020075    000
2531  015553    040   047040  053505  $MNEW:  .ASCIZ  /  NEW = /
2532  015560  036440  000040
2533
2534                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
2535
2536                                  ;;**************************************************************
2537                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
2538                                  ;*CHANGE IT TO BINARY.
2539                                  ;*CALL:
2540                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
2541                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
2542                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
2543
2544  015564  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
2545  015566  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
2546  015574  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2547  015576  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2548  015600  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2549  015602  104407                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
2550  015604  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
2551  015606  005001                          CLR     R1              ;;CLEAR DATA WORD
2552  015610  005002                          CLR     R2
2553  015612  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
2554  015614  001412                          BEQ     3$              ;;IF ZERO GET OUT
2555  015616  006301                          ASL     R1              ;;*2
2556  015620  006102                          ROL     R2
2557  015622  006301                          ASL     R1              ;;*4
2558  015624  006102                          ROL     R2
2559  015626  006301                          ASL     R1              ;;*8
2560  015630  006102                          ROL     R2
2561  015632  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
2562  015636  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
2563  015640  000764                          BR      2$              ;;LOOP
2564  015642  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
2565  015644  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
2566  015650  010237  015664                  MOV     R2,$HIOCT
2567  015654  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2568  015656  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2569  015660  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2570  015662  000002                          RTI                     ;;RETURN
2571  015664  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
2572
2573                                  .SBTTL  APT COMMUNICATIONS ROUTINE
2574
2575                                  ;;**************************************************************
2576  015666  112737  000001  016132  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
2577  015674  112737  000001  016130  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
2578  015702  000403                          BR      $ATYC
2579  015704  112737  000001  016132  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
```

```
2580  015712                          SATYC:
2581  015712  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2582  015714  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2583  015716  105737  016130                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
2584  015722  001450                          BEQ     5$              ;;IF NOT:  BR
2585  015724  122737  000001  001220          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
2586  015732  001031                          BNE     3$              ;;IF NOT:  BR
2587  015734  132737  000100  001221          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
2588  015742  001425                          BEQ     3$              ;;IF NOT:  BR
2589  015744  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
2590  015750  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2591  015756  005737  001200          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
2592  015762  001375                          BNE     1$              ;;IF NOT:  WAIT
2593  015764  010037  001214                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
2594  015770  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
2595  015772  001376                          BNE     2$
2596  015774  163700  001214                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
2597  016000  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
2598  016002  010037  001216                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
2599  016006  012737  000004  001200          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
2600  016014  000413                          BR      5$
2601  016016  017637  000004  016042  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
2602  016024  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
2603  016032  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
2604  016036  004737  014610                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
2605  016042  000000                  4$:     .WORD   0
2606  016044                          5$:
2607  016044  105737  016132          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
2608  016050  001416                          BEQ     12$             ;;IF NOT:  BR
2609  016052  005737  001220                  TST     $ENV            ;;RUNNING UNDER APT?
2610  016056  001413                          BEQ     12$             ;;IF NOT:  BR
2611  016060  005737  001200          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
2612  016064  001375                          BNE     11$             ;;IF NOT:  WAIT
2613  016066  017637  000004  001202          MOV     @4(SP),$FATAL   ;;GET ERROR #
2614  016074  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2615  016102  005237  001200                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
2616  016106  105037  016132          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
2617  016112  105037  016131                  CLRB    $LFLG           ;;CLEAR LOG FLAG
2618  016116  105037  016130                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
2619  016122  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2620  016124  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2621  016126  000207                          RTS     PC              ;;RETURN
2622  016130     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
2623  016131     000          $LFLG:  .BYTE   0               ;;LOG FLAG
2624  016132     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
2625                                  .EVEN
2626          016134          APTSIZE=200
2627          000200          APTENV=001
2628          000001          APTSPOOL=100
2629          000100          APTCSUP=040
2630          000040
2631                          .SBTTL  TRAP DECODER
2632
2633                          ;;*****************************************************************
2634                          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2635                          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
```

```
                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                ;*GO TO THAT ROUTINE.

016134  010046                 STRAP:  MOV     R0,-(SP)        ;;SAVE R0
016136  016600  000002                 MOV     2(SP),R0        ;;GET TRAP ADDRESS
016142  005740                         TST     -(R0)           ;;BACKUP BY 2
016144  111000                         MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
016146  006300                         ASL     R0              ;;POSITION FOR INDEXING
016150  016000  016156                 MOV     STRPAD(R0),R0   ;;INDEX TO TABLE
016154  000200                         RTS     R0              ;;GO TO ROUTINE


                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
016156                          STRPAD:
016156  014610                         $TYPE   ;;CALL*TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
016160  014406                         $TYPOC  ;;CALL*TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
016162  014362                         $TYPOS  ;;CALL*TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
016164  014422                         $TYPON  ;;CALL*TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
016166  007762                         $TYPDS  ;;CALL*TYPDS    TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)
016170  015072                         $CKSWR  ;;CALL*CKSWR    TRAP+5(104405)  TEST FOR CHANGE IN SOFT-SWR
016172  015330                         $RDCHR  ;;CALL*RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
016174  015412                         $RDLIN  ;;CALL*RDLIN    TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
016176  015564                         $RDOCT  ;;CALL*RDOCT    TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY

016200  000000                 ADBUFF: 0                       ;
        016242                          .=.+40
016242  000000                 ADTB0:  0
        016304                          .=.+40
016304  000000                 ADTB1:  0
        016346                          .=.+40
016346  000000                 ADTB2:  0
        016410                          .=.+40
016410  000000                 ADTB3:  0
        016452                          .=.+40
016452  000000                 ADTB4:  0
        016514                          .=.+40
016514  000000                 ADTB5:  0
        016556                          .=.+40
016556  000000                 ADTB6:  0
        016620                          .=.+40
016620  000000                 ADTB7:  0
        016662                          .=.+40
        000001                          .END
```

```
A       003554            680     744#     859     A68
ABASE = 170400            123#    241      285
ACDW1 = 000000            241     287
ACDW2 = 000000            241     288
ACPUOP= 000000            241     256
ACRLF   012356           1042    1153     1286    1292    1318    1625    1907#
ADBUFF  016200           1061    1130     1371    1385    1497    1499    1682    1755    1805    2666#
ADCNVT  010620           1258    1364     1369    1680#
ADCS    001336            336#    434      462*   1065*   1191*   1193*   1440    1689*   1693*   1695
ADCS1   010752            438*    439*    1685*   1710#
ADDBR   001340            337#    440     1195    1697
ADDW0 = 000000            241     289
ADDW1 = 000000            241     290
ADDW10= 000000            241     299
ADDW11= 000000            241     300
ADDW12= 000000            241     301
ADDW13= 000000            241     302
ADDW14= 000000            241     303
ADDW15= 000000            241     304
ADDW2 = 000000            241     291
ADDW3 = 000000            241     292
ADDW4 = 000000            241     293
ADDW5 = 000000            241     294
ADDW6 = 000000            241     295
ADDW7 = 000000            241     296
ADDW8 = 000000            241     297
ADDW9 = 000000            241     298
ADEVCT= 000000            241     247
ADEVM = 000000            241     286
ADHIGH  013250           1387    1758*   1761    1763*   2015#
ADLOW   013252           1299    1759*   1764    1766*   2016#
ADTB0   016242           1372*   1412    2668#
ADTB1   016304           1373*   1413    2670#
ADTB2   016346           1374*   1414    2672#
ADTB3   016410           1375*   1415    2674#
ADTB4   016452           1376*   1416    2676#
ADTB5   016514           1377*   1417    2678#
ADTB6   016556           1378*   1418    2680#
ADTB7   016620           1379*   1419    2682#
ADWRD1  006224           1047*   1051*   1055*   1058*   1191    1209#
ADWRD2  013146           1045*   1046*   1065    1255*   1256*   1257*   1360*   1361*   1362*   1365*   1366*   1367*   1685
                         1689    1693    1982#
AENV  = 000000            241     252
AENVM = 000000            241     253
AFATAL= 000000            241     244
AMADR1= 000000            241     269
AMADR2= 000000            241     273
AMADR3= 000000            241     276
AMADR4= 000000            241     279
AMAMS1= 000000            241     263
AMAMS2= 000000            241     271
AMAMS3= 000000            241     274
AMAMS4= 000000            241     277
AMSGAD= 000000            241     249
AMSGLG= 000000            241     250
AMSGTY= 000000            241     243
```

F6

```
AMTYP1= 000000            241     264
AMTYP2= 000000            241     272
AMTYP3= 000000            241     275
AMTYP4= 000000            241     27A
AN0     006072           1115    1175#
APASS = 000000            241     246
APRIOR= 000200            125#    241     282
APTCSU= 000040           2368    2629#
APTENV= 000001           2141    2361    2585    2627#
APTSIZ= 000200            390    2626#
APTSPO= 000100           2363    2587    2628#
ASWREG= 000000            241     254
ATESTN= 000000            241     245
AUNIT = 000000            241     248
AUSWR = 000000            241     255
AVECTR  013152            458*    485*   1034*   1231*   1346*   1425*   1607    1984#
AVECT1= 000340            124#    241     280
AVECT2= 000200            241     281
AVERM1  013264           2021#
AVERM2  013262           2020#
AVERM3  013260           2019#
AVERM4  013256           2018#
AVERM5  013254           1811    1814    2017#
AVERP1  013270           1784    2023#
AVERP2  013272           2024#
AVERP3  013274           2025#
AVERP4  013276           2026#
AVERP5  013300           1809    2027#
AVGCNT  013316           1315    1831    2034#
AVGDAT  011304           1772#   1775
AVGTAB  007724           1497*   1499#   1502*
AVRAGE  013266           1108    1303    1391    1757*   1758    1759    1767*   1773*   1776*   1783    1791    1792    2022#
AXIS1   006226           1079*   1137    1210#
AXIS2   006230           1080*   1133    1211#
AXIS3   006232           1081*   1134    1212#
B       003561            746#
BCDBIN  010454           1239    1250    1352    1638#   1670
BCDBN1  010470           1641#   1665
BCDBN2  010474           1643#   1659
BCDEND  010552           1644    1647    1660#
BCDERR  010574           1649    1651    1661    1667#
BCDTAB  010610           1240    1242    1244    1246    1251    1353    1354    1639    1640*   1671#
BEG     001372            349     351#
BEGIN   001356            152     348#    436     2245
BEGIN1  001364            153     350#
BIT0  = 000001            107#
BIT00 = 000001             97#    107
BIT01 = 000002             96#    106
BIT02 = 000004             95#    105
BIT03 = 000010             94#    104
BIT04 = 000020             93#    103
BIT05 = 000040             92#    102
BIT06 = 000100             91#    101
BIT07 = 000200             90#    100
BIT08 = 000400             89#     99    2076
BIT09 = 001000             88#     98    2084    2151
```

```
BIT1   = 000002          106#
BIT10  = 002000           87#    934     960    2129
BIT11  = 004000           86#   2091
BIT12  = 010000           85#    935
BIT13  = 020000           84#   2136
BIT14  = 040000           83#   2062
BIT15  = 100000           82#
BIT2   = 000004          105#
BIT3   = 000010          104#
BIT4   = 000020          103#
BIT5   = 000040          102#    882     888
BIT6   = 000100          101#   1127    1580
BIT7   = 000200          100#    639    1070
BIT8   = 000400           99#    984
BIT9   = 001000           98#
BPTVEC= 000014          114#
BRLEV1  013236          1013*   1014*   2010#
BRLEV2  013240          2011#
BRLEV3  013242          2012#
BUFADR  007660          1448*   1478#   1980
BUFNUM  007640          1434    1441    1469#
C       003566           748#    857     866
CALBRT  005104           474    1034#
CALBT1  005116          1034    1037#
CALBT2  005152          1045#   1157
CALB2A  005254          1052    1057    1059#   1158
CATORZ  011332          1260    1782#
CATR1   011432          1801#   1803
CATR2   011454          1806#   1823    1825    1827
CATR3   011514          1815#   1819
CATR4   011534          1816    1821#
CATR5   011560          1807    1831#
CHAR    003402           701     708#    838     848    1162
CHAR1   003452           717#    731
CHAR2   003460           719#    727
CHAR3   003500           720     723#    724
CHAR4   003430           712#    713
CHRCNT  013156          1581*   1593*   1594    1614    1616*   1643    1645*   1664    1986#
CHRCOL  003550           696*    702*    738#
CHTIME  005050           508     513     545     595     633     677     830    1013#
CHTMA   005056          1014#   1017
CHTMB   005072          1015    1018#
CH01    004266           836     857#
CH02    004310           846     866#
CKSWR  = 104405           971     981    1627    2061    2661#
CLRVC   004536           884     890     934#
CLRVCA  004562           939#    942     944
CLRVCB  004624           948     950     953#
CMPCNT  011330          1098*   1388*   1754*   1771    1779#
CMPTE   011152          1259    1753#
CMPTEA  011172          1390    1756#
CMPTEB  011166          1099    1755#
CNSTS1  006154          1059    1193#
CNTRLA  012351          1605    1906#
CNTRLC  012343           461    1904#
CNVTSV  006112          1086    1091    1184#   1206
```

```
COUNT   013160          1044*   1254*   1359*   1681    1987#
CPDLAY  010736           418    1703#
CPTIME  010746           418*   1686    1708#
CPTYPE  010750           415*    416*   1013    1709#
CR    = 000015            22#   2407    2417
CRLF  = 000200            23#   2378    2417
CSB     001344           340#   1189*
CSC     001354           346#   1432
CSR     001342           339#    463*   1190*
D       003573           750#
DDISP = 177570            29#    217     377
DECOUT  011052          1724    1730#
DECPNT  011140          1717*   1721    1723    1728*   1745#
DECPRT  010754          1321    1715#
DEC1    011100          1731    1733    1736#
DEC2    011114          1735    1738#
DELAY   013216          1363*   1368*   1683    1700*   2002#
DELAY1  013220          1686*   1687*   2003#
DGT0    006060          1160    1170#
DGT1    006062          1171#
DGT2    006064          1172#
DGT3    006066          1173#
DGT4    006070          1107    1174#
DH1     013075           329    1973#
DIGCNT  011134          1716*   1725*   1732    1743#
DIGIT   011132          1719*   1720*   1730    1734*   1737*   1740    1742#
DISPLA  001140           217#    377*    383*   2106*   2128*
DISPRE  000174           148#    383
DOT     012361          1908#
DSWR  = 177570            28#    216     376
DT1     013134           330    1980#
E       003600           752#    862     871
EMTVEC= 000030           117#    361*    362*
EM1     013032           328    1967#
ERRVEC= 000004           110#    374     375*    386*   2067    2068*   2070*   2073*
F       003605           754#
FILE1   011352          1785#   1788
FILE2   011410          1793#   1796
FIRST   013214          2001#
G       003612           756#
GEN1    003366           701#    703
GETDAT  011216          1760#   1770
GNS   = ****** U         147    2656    2657    2658    2659    2660    2661    2662    2663    2664
H       003617           758#    858     867
HIGH    013202           592*    615    1996#
HIORDV  013246          1756*   1768*   1772*   2014#
HT    = 000011            20#   2376    2417
I       003624           760#
INBUF   010436           466     469     472     475     478    1040    1579    1582*   1583*   1631#   1638
INCR    013204          1997#
INITA   002072           456#    458    1984
INIT1   002042           419     448#
INIT2   002076           450     458#    483    1611
IOTVEC= 000020           115#    359*    360*
J       003631           762#
K       003636           764#
```

```
KSTOR1  013162          1240*   1241    1244    1252    1353*   1360    1988#
KSTOR2  013164          1048*   1155    1241*   1246*   1329    1354*   1365    1401    1989#
KSTOR3  013166          1251*   1265    1270    1276    1281    1990#
KSTOR4  013170          1252*   1255    1293    1327    1331*   1991#
KSTOR5  013172          1992#
KSTR11  013174          1993#
KSTR12  013176          1994#
L       003643           766#    863     872
LADTB   007414          1389    1412#
LDTRAP  001740           423#    448     459
LF    = 000012            21#   2411    2417
LOADVC  004626           885     955#
LODPNT  003154           649     651     660#
LODVCA  004664           962#    973
LODVCB  004672           964#    970
LODVCC  004676           963     965#
LOW     013200           542*    550     551     591*    608     616    1995#
M       003650           768#
MARKER  006222          1085*   1090*   1184    1208#
MESPRT  013244          1234*   1287    1310    1314*   2013#
MES10   012513          1038    1927#
MES13   012561          1233    1934#
MES14   012617          1237    1350    1940#
MES15   012633           489    1943#
MES16   012701          1248    1950#
MES19   012720          1298    1953#
MES2    011772          1427    1861#
MES20   012743          1313    1957#
MES3    012023           948    1866#
MES4    012070           457    1873#
MES6    012370           487    1911#
MES7    012420          1036    1915#
MES8    012454          1348    1920#
MES9    012505          1400    1925#
MINUS1  013314          1833    2033#
MINUS2  013312          1836    2032#
MINUS3  013310          1839    2031#
MINUS4  013306          1842    2030#
MINUS5  013304          1822*   2029#
MODE    003542           714     735#
N       003655           685     770#    860     861     869     870
NBEXT   001334           333#
N0      003756           690     796#   1175
N1      003763           798#    865    1176
N2      003770           800#    874    1177
N3      003775           802#   1178
N4      004002           804#   1179
N5      004007           806#   1180
N6      004014           808#   1181
N7      004021           810#   1182
N8      004026           812#
N9      004033           814#
O       003662           772#
OPS1    013224          1235*   2005#
ORHIGH  013332          1802    1824*   2040#
ORLOW   013302          1319    1800    1826*   2028#
```

```
OVRHI   011544          1810    1824#
OVRLO   011552          1812    1826#
P       003667           774#
PBB     002370           501     514     519#
PC    =%000007            41#    446#    448#    459#    465#    500#    501#    502#    513#    514#    515#    531#    545#
                         584#    595#    624#    633#    656#    677#    682#    687#    692#    693#    701#    704#    733#
                         838#    838#    848#    852#    884#    885#    886#    890#    908#    911#    921#    926     953#
                         974#    983#    989#   1018#   1039#   1086#   1091#   1099#   1123#   1126#   1144#   1154#   1162#
                        1166#   1207#   1238#   1239#   1249#   1250#   1253#   1258#   1259#   1260#   1298#   1302#   1306#
                        1321#   1351#   1352#   1355#   1364#   1369#   1390#   1405#   1406#   1454#   1492#   1498    1504#
                        1507#   1626#   1666#   1669#   1701#   1724#   1727#   1777#   1843#   1849#   1850#   2138#   2144#
                        2197#   2244    2366#   2385#   2392#   2349#   2413#   2415#   2466#   2604#   2621#
PIC0    002266           497#   1004
PIC1    002332           510#   1005
PIC12   004342           881#   1011
PIC12A  004350           882#    887
PIC3    002446           517     542#   1006
PIC4    002652           591#   1007
PIC4B   002704           596#
PIC5    003030           632#   1008
PIC6    003212           676#   1009
PIC6A   003224           678#    694
PIC6B   003346           682     687     692     696#
PIC7    004070           695     829#   1010
PIC7A   004144           837#    841
PIC7AA  004106           832#    853
PIC7B   004226           847#    851
PIRQ  = 177772            27#    401#    408
PIRQVE= 000240           121#
PLUS1   013320          1832    2035#
PLUS2   013322          1835    2036#
PLUS3   013324          1838    2037#
PLUS4   013326          1841    2038#
PLUS5   013330          2039#
PRINT1  013150          1983#
PROC    013154          1040#   1049    1053    1985#
PR0   = 000000            44#
PR1   = 000040            45#
PR2   = 000100            46#
PR3   = 000140            47#
PR4   = 000200            48#
PR5   = 000240            49#
PR6   = 000300            50#
PR7   = 000340            51#
PS    = 177776            24#     25     394#    402#    411
PSW   = 177776            25#    1680#   1715#
PWRMSG  014314          2243    2250#
PWRVEC= 000024           116#    365#    366#   2212#   2213#   2222#   2228#   2240#   2241#
P3      002510           550#    585
P3A     002534           555#    559
P3B     002556           563#    567
P3C     002600           571#    575
P3D     002622           579#    583
P4      002724           600#    625
P4A     002742           606#    611
P4B     002776           618#    623
```

```
P7CNT   004262            835*    840*    845*    850*    855#
P7PNT   004264            836*    837     839*    846*    847     849*    856#
Q       003674            776#
QMARK   012365            482     1629    1668    1910#
R       003701            778#
RDCHR * 104406            2441    2507    2662#
RDLIN * 104407            2549    2663#
RDOCT * 104410            2664#
RECVRY  007024             488    1346#
RECVY1  007036            1346    1349#
RECVY2  007066            1355#   1398    1407
REPTST  006244             477    1231#
REPT1   006256            1231    1234#   1245
REPT2   006342            1243    1247#
REPT2A  006364            1252#   1333
REPT3   006372            1253#   1332
REPT4   006570            1262    1268    1273    1279    1282    1285#
REPT5   006606            1288    1291#
REPT6   006714            1311    1314#
REPT6A  006742            1320#   1323
REPT7   006766            1264    1269    1274    1280    1284    1325    1327#
REPT7A  007014            1330    1333#
REPT8   006626            1295#
REPT8A  007020            1293*   1294*   1295    1335#
RESTST  007444             468    1425#
RESVEC* 000010             111#
ROUTPT  005030             996    1004#
R0      =%000000            32#    406*    409*    412*    415     417*    418     434*    435*    436     440*    442*    497*
                           510*    553*    558*    562*    566*    570*    574*    578*    582*    605*    610*    617*    622*
                           642*    646*    654     664*    716*    730*    918*    921     937*    943*    957*    998     1083*
                          1088*   1100*   1101*   1102*   1103*   1104*   1105    1109*   1110*   1111*   1112    1131*   1142*
                          1205*   1371*   1372    1373    1374    1375    1376    1377    1378    1379    1384*   1393*   1432*
                          1433*   1436*   1440*   1445    1448    1450    1523    1533*   1537    1553    1554    1567*   1653*
                          1658    1805*   1808    2169    2178*   2171*   2178*   2179*   2180*   2181*   2182*   2183    2188
                          2193*   2195*   2199    2201    2214    2239*   2359    2360*   2365    2370    2373*   2546    2550*
                          2553    2569*   2581    2589*   2593    2594    2596*   2597*   2598    2620*   2639    2640*   2641
                          2642*   2643*   2644*   2645*
R1      =%000001            330    424*    425*    426*    427     441*    442     443     444     498*    511*    546*    596*
                           641*    647*    661*    717*    726*    938*    941*    958*   1084*   1089*   1112*   1113*   1114*
                          1115    1186*   1196    1378*   1372*   1373*   1374*   1375*   1376*   1377*   1378*   1379*   1386*
                          1389    1392    1434*   1436    1441*   1444    1450    1524    1537*   1538    1542    1566*   1586*
                          1587*   1588    1598    1592    1598    1602    1608    1612    1617*   1620    1622    1641*   1654
                          1655*   1656*   1657*   1658*   1660    1663    1813*   1817*   1818    1821*   1822*   2215    2238*
                          2547    2551*   2555*   2557*   2559*   2562*   2565    2568*   2582    2619*
R2      =%000002            340    423*    425     427*    428     429     522*    525     547*    597*    660*    665*    680*
                           685*    698*    837*    847*    959*   1107*   1115*   1161*   1195*   1196*   1197*   1198*   1199*
                          1200    1201    1320*   1385*   1391*   1435*   1437*   1442*   1451*   1525    1536*   1540*   1543
                          1550*   1551*   1552    1557*   1565*   1642*   1654*   1814*   1815    2216    2237*   2548    2552*
                          2556*   2558*   2560*   2566    2567*
R3      =%000003            350    521*    529     548*    598*    638*    643*    648*    662     718*    719*   1072*   1076*
                          1080    1133*   1135*   1138*   1200*   1526    1534*   1535*   1549*   1552*   1561*   1562*   1564*
                          1639*   1663*   1784*   1786*   1792*   1794*   1800*   1801*   1802    2217    2236*   2293    2302*
                          2308*   2309*   2312*   2317*   2318*   2319    2328*   2503    2504*   2585    2588*   2589    2513
                          2515    2517*   2519*
R4      =%000004            360    528*    549*    555     563     571     579     599*    608     609     620     621     715*
                           725     729     732    1073*   1077*   1081    1134*   1136*   1141*   1204*   1319*   1320    1322
```

```
                    1327*  1328*  1329   1389*  1579*  1592*  1618   1638*  1646   1648   1650   1652*  1653
                    1662   1682*  1697*  1755*  1757   1760   2218   2235*  2294   2296*  2297*  2298*  2299
                    2308*  2314   2316*  2324*  2327*
R5      =%000005      37#    649*   651*   661    667    668*   697*   831*   1061*  1120*  1130*  1138   1201*
                    1527   1529*  1531*  1538*  1542*  1557   1563*  2219   2234*  2295   2301*  2303*  2305*
                    2306*  2307*  2308   2326*
R6      =%000006      38#     40    353*   354*   355
R7      =%000007      39#     41
S         003706     788#
SP      =%000006      40#    357*   374*   384*   386    915*   998    1149*  1194   1295*  1299*  1303*  1307*
                    1401*  1460*  1498*  1523*  1524*  1525*  1526*  1527*  1528*  1529   1532*  1545   1547*
                    1549   1559   1561   1563   1564   1565   1566   1567   1569*  1570*  1606   1610   2067*
                    2070   2072   2073   2102   2103   2107*  2133   2153*  2156*  2169*  2174*  2195   2199*
                    2214*  2215*  2216*  2217*  2218*  2219*  2220*  2221   2229*  2233   2234   2235   2236
                    2237   2238   2239   2244*  2285*  2286   2287   2288*  2293*  2294*  2295*  2301   2326
                    2327   2328   2329*  2330*  2359*  2360   2370*  2372   2373   2374*  2376   2378   2380
                    2386   2388*  2390*  2398*  2402   2406   2407   2411   2429*  2430*  2431   2436*  2439*
                    2440*  2443   2446*  2449   2451   2453   2454*  2456   2458   2460   2461*  2462*  2463*
                    2467*  2468   2470*  2471*  2472*  2473*  2474*  2484*  2485*  2488*  2489*  2490   2492
                    2494*  2503*  2508   2519   2520*  2521*  2522*  2544*  2545*  2546*  2547*  2548*  2558
                    2553*  2561*  2562   2564   2565*  2567   2568   2569   2581*  2582*  2589   2590*  2601
                    2602*  2603*  2613   2614*  2619   2620   2639*  2640
SPACEA  004040      816#    864    873
SPACEX  007712     1484*  1489*  1491*  1493*  1503*
STACK = 001100      15#    357
STCHAN  013212     2000#
STKLMT= 177774      26#
SWR     001136      216#   376*   378    382*   392*   639    882    888    945    949    982    1045   1048
                    1070   1127   1147   1155   1261   1263   1324   1397   2062   2076   2078   2084   2091
                    2129   2136   2148   2151   2220   2233*  2425   2453*  2460*
SWREG   000176     149#    382    2425   2436
SW0     = 000001    79#
SW00    = 000001    69#     79
SW01    = 000002    68#     78
SW02    = 000004    67#     77
SW03    = 000010    66#     76
SW04    = 000020    65#     75
SW05    = 000040    64#     74
SW06    = 000100    63#     73
SW07    = 000200    62#     72
SW08    = 000400    61#     71
SW09    = 001000    60#     70
SW1     = 000002    78#
SW10    = 002000    59#   1147    1261    1397
SW11    = 004000    58#
SW12    = 010000    57#    945
SW13    = 020000    56#   1263
SW14    = 040000    55#
SW15    = 100000    54#
SW2     = 000004    77#
SW3     = 000010    76#
SW4     = 000020    75#
SW5     = 000040    74#
SW6     = 000100    73#
SW7     = 000200    72#
SW8     = 000400    71#
```

5

M6

```
SW9    = 001000            70#
T        003713            782#
TAGA     011256           1765    1767#
TBITVE= 000014            112#
TEMP     013226            348*    350*    449    2006#
TEMP1    013230            956*    961    1097*   1387*   1681*   1698*   1753*   1769*   1771*   1774*   1782*   1787*   1790*
                          1795*   1804*   1806*   2007#
TEMP2    013232           1760*   1761    1763    1764    1766    1767    1783*   1785*   1786    1791*   1793*   1794    1808*
                          1809    1811    1815    2008#
TEMP3    013234           1146*   1358*   1505*   2009#
TICKS    013210            499*    512*    544*    594*    632*    676*    829*    881*    986*   1016*   1999#
TIMER    004726            502     515     584     624     656     693     852     886     981#
TIMERA   004742            984#
TIMER1   004764            987     989#
TIMER2   004766            985     994#
TIMER4   005024           1002#
TIMSV    013206            982*    984     994*    995*    996*    997*   1002    1998#
TITLE    011706            455    1852#
TKVEC  = 000060            119#
TPVEC  = 000064            120#
TRAPVE= 000034            118#    363*    364*
TRTVEC= 000014            113#
TSLO     011240           1762    1764#
TSTCT1   006550           1277    1281#
TSTCT2   006526           1271    1276#
TSTCT3   006504           1266    1270#
TSTCT4   006452           1263#
TSTFLG   011672            983    1144    1154    1253    1355    1454    1847#
TST1     002256            495#
TST10    004332            854     879#
TST11    005074           1022#
TST12    006234           1218#
TST13    007022           1340#
TST14    007434           1423#
TST2     002322            508#
TST3     002436            537#
TST4     002642            589#
TST5     003020            630#
TST6     003202            658     674#
TST7     004060            827#
TYPDS  = 104404            916    1461    2660#
TYPE   = 104400            454     456     460     481     486     488     914     917     947    1035    1037    1041    1152
                          1232    1236    1247    1285    1289    1291    1312    1317    1347    1349    1399    1426    1457
                          1462    1568    1604    1624    1628    1667    2131    2139    2168    2185    2187    2190    2192
                          2196    2203    2242    2320    2381    2433    2435    2438    2445    2464    2511    2514    2518
                          2656#
TYPOC  = 104401           2176    2200    2437    2657#
TYPON  = 104403           2659#
TYPOS  = 104402           1150    1296    1300    1304    1308    1402    1500    2658#
TYPT1    011002           1719#   1729
TYPT2    011010           1720#   1722
TYPT3    011042           1726    1728#
U        003720            784#
USECLK   013222           1043*   1056*   1187    2004#
V        003725            786#
VCSTAT   001346            342#    464*    519*    520     543*    548     593*    598     635*    638     681*    686*    691*
```

A1

```
                      697     712     723     831     832*    842*    934*    935*    939     955*    957     1067*   1074*
                     1078*   1079    1119*   1120    1132*   1137*   1139    1202    1453*
VCXREG  001350        343#    497     511     546     550*    596     636*    642     647     710*    721*    958     1068*
                     1072    1077
VCYREG  001352        344#    498     510     547     551*    597     637*    641     646     711*    722*    959     1069*
                     1073    1076
VISAU1  002250        485     488#
VISUAL  002236        471     485#
VSUAL0  002254        453     490#    927
W       003732        788#
X       003737        790#
XPOS    003546        678*    683*    688*    710     721     729*    732*    737#    833*    843*    1121*   1124*
XPRTAV  007714       1406    1497#
XPTA1   007722       1498#   1506
XSPACE  007662       1298    1302    1306    1405    1486#   1490    1504
XSPRD1  013334       1283    1322    1831*   1832*   1833*   1834    2041#
XSPRD2  013336       1278    1834*   1835*   1836*   1837    2042#
XSPRD3  013340       1272    1837*   1838*   1839*   1840    2043#
XSPRD4  013342       1267    1840*   1841*   1842*   2044#
XTTYIN  010206        465    1039    1238    1249    1351    1579#   1630    1669    1849
Y       003744        792#
YPOS    003544        679*    684*    689*    708     711     722     725*    728*    736#    834*    844*    1122*   1125*
YPT     003552        708*    728     739#
Z       003751        794#
ZERO    011136       1718*   1734    1736*   1744#
SAPYHD  001000        177     183#
SARG1   001616        388#
SASTAT* ****** U      2607    2622
SATYC   015712        2578    2580#
SATY1   015666        2576#
SATY3   015674        2366    2577#
SATY4   015704        2144    2579#
SBASE   001254        285#    435     438
SBDADR  001122        210#
SBDDAT  001126        212#    355    1445*   1446    1980
SBELL   001170        232#   2131    2159
SCDW1   001260        287#
SCDW2   001262        288#
SCHARC  015066       2383*   2393*   2400    2409*   2414#
SCKSWR  015072       2425#   2661
SCMTAG  001100        198#    352     353     361     367     368
SCM1  = 000002        228#    229#    230#
SCM2  = 000004        228#    229#    230#
SCM3  = 000002        226#    228
SCNTLG  015535       2433    2528#
SCNTLU  015530       2445    2527#
SCPUDP  001226        256#
SCRLF   001175        234#   2139    2159    2168    2187    2192    2196    2382    2417    2464    2527
SDBLK   010176       1534    1568    1576#
SDDW0   001264        289#
SDDW1   001266        290#
SDDW10  001310        299#
SDDW11  001312        300#
SDDW12  001314        301#
SDDW13  001316        302#
SDDW14  001320        303#
```

```
$DDW15  001322            304#
$DDW2   001270            291#
$DDW3   001272            292#
$DDW4   001274            293#
$DDW5   001276            294#
$DDW6   001300            295#
$DDW7   001302            296#
$DDW8   001304            297#
$DDW9   001306            298#
$DEVCT  001210            247#
$DEVM   001256            286#
$DOAGN  004512            918     919     925#
$DTBL   010166           1537    1572#
$ENDAD  004502            162     921#
$ENDCT  004450            912#
$ENDMG  004521            914     929#   1458
$ENULL  004516            917     928#   1462
$ENV    001220            252#   2141    2361    2585    2609
$ENVM   001221            253#    390    2363    2368    2587
$EOP    004414            902#
$EOPCT  004442            909#    913
$ERFLG  001103            201#   2051    2080    2082    2088*   2110    2126*   2159
$ERMAX  001115            207#    369*   2082    2105*   2110
$ERROR  013624            361    2125#
$ERRPC  001116            208#   1980    2133*   2134*   2135    2159    2174
$ERRTB  001324            324#   2182
$ERRTY  014006           2138    2167#
$ERTTL  001112            205#   2132*   2159
$ESCAP  001166            231#    368*   2104*   2154    2156    2159
$ETABL  001220            251#
$ETEND  001324            189     307#
$FATAL  001202            244#   2613*
$FFLG   016132           2576*   2579*   2607    2616*   2624#
$FILLC  001154            224#   2386    2417
$FILLS  001153            223#   2417
$GDADR  001120            209#
$GDDAT  001124            211#   1444*   1446    1980
$GET42  004472            918#
$HD   = 000000             11
$HIBTS  001000            184#
$HIOCT  015664           2566*   2571#
$ICNT   001104            202#   2095*   2096    2098*   2109
$ILLUP  014306           2212    2228    2247#
$ITEMB  001114            206#   2135*   2143    2159    2171
$LF     001176            235#   2159    2417    2518    2527
$LFLG   016131           2617*   2623#
$LPADR  001106            203#    370*   2086*   2102*   2107    2109
$LPERR  001110            204#    371*   2086    2103*   2109    2153
$MADR1  001232            269#
$MADR2  001236            273#
$MADR3  001242            276#
$MADR4  001246            279#
$MAIL   001200            185     189     242#    388     2101    2141    2361
$MAMS1  001230            263#
$MAMS2  001234            271#
$MAMS3  001240            274#
```

c1

```
$MAMS4  001244            277#
$MBADR  001002            185#
$MFLG   016130           2577*   2583    2618*   2622#
$MNEW   015553           2438    2531#
$MSGAD  001214            249#   2593*   2596
$MSGLG  001216            250#   2598*
$MSGTY  001200            243#   2591    2599*   2611    2615*
$MSWR   015542           2435    2529#
$MTYP1  001231            264#
$MTYP2  001235            272#
$MTYP3  001241            275#
$MTYP4  001245            278#
$MXCNT  013622           2099    2109#
$NULL   001152            222#   2388    2417
$NWTST= 000001            492#    505#    534#    586#    627#    671#    824#    876#   1019#   1215#   1337#   1420#
$OCNT   014604           2292*   2321*   2334#
$OMODE  014606           2287*   2291*   2296    2299*   2310*   2336#
$OVER   013606           2063    2079    2087    2097    2106#
$PASS   001206            246#    389*    906*    907*    915     928    2093    2110
$PASTM  001006            187#
$PRIOR  001252            282#
$PWRAD  014302           2245#
$PWRDN  014142            365    2212#   2240
$PWRMG  014276           2243#
$PWRUP  014214           2222    2228#
$QUES   001174            233#   2159    2417    2511    2527
$RDCHR  015330           2484#   2662
$RDDEC= ****** U         2665
$RDLIN  015412           2503#   2663
$RDOCT  015564           2544#   2664
$RDSZ = 000010           2496#
$REGAD  001156            226#
$REG0   001160            228#
$REG1   001162            229#
$RTNAD  004514            927#
$R2A  = ****** U         2665
$SAVRE= ****** U         2665
$SAVR6  014312           2221*   2229    2230*   2231*   2249#
$SCOPE  013344            359    2060#
$SETUP= 000017            326#    352#    358     359     361     363     365     367     368     370     387     904    2061
                         2126    2151    2158
$STUP = 177777            326#    352#
$SVLAD  013552           2071    2100#
$SVPC = 000210            160#    165
$SWR  = 167400              1#     11     132     133     134     135     136     137     138     139     230     231     232
                          367     368     370     371     496     509     538     590     631     675     828     880     899
                          905     920     926     928    1023    1219    1341    1424    2052    2053    2054    2055    2056
                         2062    2074    2076    2077    2080    2081    2082    2089    2090    2091    2103    2106    2109
                         2117    2118    2119    2120    2121    2129    2136    2148    2151    2159    2246
$SWREG  001222            254#    392
$SWRMK= 000000            139     140    2056    2057    2078
$TESTN  001204            245#   2101*
$TIMES  001164            230#    367*    496*    509*    538*    590*    631*    675*    828*    880*    905*   1023*   1219*
                         1424*   2089*   2096    2099*   2109
$TKB    001144            219#   1586    2421    2429    2488
$TKS    001142            218#   1580*   1584    1847    2421    2427    2486
```

```
STN    = 000015            1#     11     492     496#    505     509#    534     538#    586     590#    627     631#    658
                         671     675#    824     828#    854     876     880#   1019    1023#   1215    1219#   1337    1341#
                        1420    1424#
STPB     001150           221#   1488*   1598*   1740*   2406*   2417
STPFLG   001155           225#   2355    2417
STPS     001146           220#   1486    1596    1738    2404    2417
STRAP    016134           363    2639#
STRP   = 000011          2647#   2657#   2658#   2659#   2660#   2661#   2662#   2663#   2664#   2665#
STRPAD   016156          2644    2655#
STSTM    001004           186#
STSTNM   001102           200#    904*   2051    2078    2100*   2101    2106    2110    2128    2159
STTYIN   015520          2504    2505    2522    2526#
STYPBN=  ******  U       2661
STYPDS   007762          1522#   2660
STYPE    014610          2355#   2604    2647    2656
STYPEC   015022          2385    2392    2399    2404#   2405    2466
STYPEX   015070          2410    2412    2415#
STYPOC   014406          2290#   2657
STYPON   014422          2289    2292#   2659
STYPOS   014362          2285#   2658
SUNIT    001212           248#
SUNITM   001010           188#
SUSWR    001224           255#
SVECT1   001250           280#   1059*
SVECT2   001251           281#
SXTSTR   013356          2065#
SSGET4=  000000           920#
S0FILL   014605          2286*   2290*   2300    2335#
S40CAT=  ******  U       2062    2138
.      = 016662           143#    147#    160     161#    163#    165#    166#    173     174#    176#    178#    197#    236
                          356     370     371     528     557     565     573     581     607     613     619    823#     928
                          932     968    1487    1576#   1634#   1739    1745    2109    2110    2159   2206#    2224    2248
                         2417    2421    2526#   2527    2533    2625#   2667#   2669#   2671#   2673#   2675#   2677#   2679#
                         2681#   2683#
.$ASTA=  ******  U       2577    2580
.$X    = 001000           173#    178
```

```
COMMEN    122#
ENDCOM    122#
ERROR      16#   1449
ESCAPE    122#
GETPRI    122#
MULT      122#
NEWTST    122#    492    505    534    586    627    671    824    876   1019   1215   1337   1420
POP       122#   1563   2233   2234   2567   2619   2620
PUSH      122#   1522   2214   2220   2546   2580   2582   2603
REPORT    122#
SCOPE      17#    495    508    537    589    630    674    827    879    903   1022   1218   1340   1423
SETPRI    122#
SETTRA   2647#   2657   2658   2659   2660   2661   2662   2663   2664
SETUP     122#    352
SKIP      122#    658    854   1447
SLASH     122#
SPACE     122#
STARS     122#    158    170    172    179    193    236    240    492    494    505    507    534    536    586
          588    627    629    671    673    824    826    876    878    895   1019   1021   1215   1217   1337
         1339   1420   1422   1512   2048   2113   2162   2210   2226   2262   2340   2420   2476   2496   2536
         2575   2633
SWRSU     122#    372#
TRMTRP   2647#
TYPBIN    122#
TYPDEC    122#    915
TYPNAM    122#
TYPNUM    122#
TYPOCS    122#
TYPOCT    122#   2174   2198   2436
TYPTXT    122#
$SCMRE    190#    228    229
$SCMTM    190#
$SESCA    122#
$SNEWT    122#    492    505    534    586    627    671    824    876   1019   1215   1337   1420
$SSET    2647#   2657   2658   2659   2660   2661   2662   2663   2664
$SSETM    388#    389
$SSKIP    122#    658    854
.EQUAT      1#     11
.HEADE      1#
.SETUP      1#    326    352
.SWRHI      1#    127
.SWRLO    140#
.SACT1      1#    155
.SAPTB      1#    237#
.SAPTH      1#    167
.SAPTY      1#   2572
.SCATC      1#    140
.SCMTA      1#    190
.SEOP       1#    892
.SERRO      1#   2110
.SERRT      1#   2159
.SPARM      1#
.SPOWE      1#   2207
.SRDOC      1#   2533
.SREAD      1#   2417
.SSAVE      1#
```

```
.$SCOP      1#   2045
.$SPAC      1#
.$SWDO      1#
.$TRAP      1#   2630
.$TYPD      1#   1509
.$TYPE      1#   2337
.$TYPO      1#   2259
```

DZARBB.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
        1116  1142  1164  1186  1205  1381  1393  1437  1451  1455  1489  1505  1416  1645  1687
        1698  1769  1774  1787  1793  1795  1806  2178
DECB    2310  2321  2390  2393
EMT       16
HALT     147   951  1326  1820  2150  2223  2247  2357
INC      416   439   526   702   726   730   906   965  1090  1103  1331  1363  136A  1459  1540
        1593  1720  1725  1785  1817  1822  1824  1826  2095  2132  2231  2316  2324  2473  2615
INCB    2100  2126  2413
IOT       17
JMP      152   153   419   453   468   471   474   477   480   695   926  1002  1157  1158  1332
        1333  1607  1611  1670
JSR      448   459   465   500   501   502   513   514   515   545   584   595   624   633   649
         651   656   677   682   687   692   693   701   830   838   848   852   884   885   886
         890   921   983  1039  1086  1091  1099  1123  1126  1144  1154  1162  1238  1239  1249
        1250  1253  1258  1259  1260  1298  1302  1306  1321  1351  1352  1355  1364  1369  1390
        1405  1406  1454  1504  1669  1724  1849  2138  2144  2364  2385  2392  2399  2466  2604
MOV      350   353   357   359   360   361   362   363   364   365   366   370   371   374   375
         376   377   382   383   384   386   392   404   405   406   415   418   423   424   425
         427   434   435   438   440   441   458   485   496   497   498   499   509   510   511
         512   520   521   522   523   524   538   544   546   547   548   549   550   551   553
         554   561   562   569   570   577   578   590   591   592   594   596   597   598   599
         600   601   604   605   615   616   617   631   632   638   641   642   643   646   647
         648   660   661   675   676   678   679   680   683   684   685   688   689   690   696
         697   708   710   711   715   716   717   721   722   728   828   829   831   832   833
         834   835   836   837   842   843   844   845   846   847   880   881   911   915   918
         934   937   956   957   958   959   960   961   962   982   997  1013  1023  1034  1040
        1044  1048  1056  1059  1061  1065  1072  1073  1074  1076  1077  1078  1079  1080  1081
        1082  1083  1084  1088  1089  1097  1098  1100  1105  1106  1107  1112  1115  1120  1121
        1122  1124  1125  1130  1131  1133  1134  1137  1138  1146  1149  1159  1160  1161  1189
        1190  1195  1200  1201  1219  1231  1240  1241  1246  1251  1252  1254  1293  1295  1299
        1303  1307  1319  1320  1327  1346  1353  1354  1356  1358  1359  1370  1371  1372  1373
        1374  1375  1376  1377  1378  1379  1384  1385  1386  1387  1388  1389  1391  1401  1404
        1424  1425  1430  1432  1434  1435  1436  1440  1441  1442  1444  1445  1448  1460  1488
        1497  1498  1503  1523  1524  1525  1526  1527  1528  1529  1534  1537  1557  1563  1564
        1565  1566  1567  1569  1570  1579  1586  1592  1617  1638  1639  1653  1654  1663  1681
        1682  1686  1693  1697  1716  1717  1718  1719  1734  1736  1740  1753  1754  1755  1757
        1758  1759  1760  1763  1766  1771  1782  1783  1784  1786  1790  1791  1792  1794  1800
        1804  1805  1808  1814  1831  1834  1837  1840  2067  2068  2070  2073  2086  2098  2099
        2102  2103  2106  2107  2128  2133  2153  2156  2169  2174  2183  2188  2193  2195  2199
        2212  2213  2214  2215  2216  2217  2218  2219  2220  2221  2222  2228  2229  2233  2234
        2235  2236  2237  2238  2239  2240  2241  2244  2285  2293  2294  2295  2301  2308  2326
        2327  2328  2329  2330  2359  2360  2365  2373  2388  2436  2453  2460  2462  2484  2485
        2503  2504  2519  2520  2521  2522  2544  2545  2546  2547  2548  2550  2565  2566  2567
        2568  2569  2581  2582  2589  2593  2598  2599  2601  2603  2613  2619  2620  2639  2640
        2644
MOVB     369   718  1045  1255  1360  1365  1532  1535  1549  1552  1561  1598  1685  1689  2101
        2105  2135  2143  2286  2287  2290  2291  2292  2296  2299  2300  2319  2370  2398  2406
        2429  2488  2508  2513  2553  2576  2577  2579  2642
NEG     1531  2297
NOP      490   698   699   700   922   923   924   936   966   999  1000  1001  1060  1062  1063
        1064  1066  1316  1431  1691  1692
RESET    351   891   920  1429
ROL     2303  2305  2306  2307  2309  2556  2558  2560
ROLB     719
ROR     1773
RTI      385   404  1571  2108  2158  2246  2331  2375  2465  2495  2523  2570
```

```
RTS      446   531   668   704   733   953   974   989  1018  1166  1207  1492  1507  1426  1666
        1701  1727  1741  1777  1843  1850  2197  2415  2621  2645
SBC      409   412
SEC      407   410
SUB      571   579   620   964   972  1538  1721  2134  2596
TRAP    2647  2657  2658  2659  2660  2661  2662  2663  2664
TST      381   408   428   443   449   451   654   667   949  1184  1187  1242  1287  1310  1324
        1392  1543  1553  1606  1610  1614  1618  1643  1662  1664  1683  1730  2069  2093  2148
        2154  2201  2314  2372  2380  2402  2451  2458  2468  2564  2591  2609  2611  2641
TSTB     411   527   556   564   572   580   606   612   618   662   712   723   939   967  1139
        1202  1486  1545  1559  1584  1596  1695  1738  1847  2080  2355  2404  2427  2486  2583
        2594  2607
WAIT    1192
.ASCII   233   234  1873  1880  1885  1890  1895
.ASCIZ   232   235   929  1852  1861  1866  1900  1904  1906  1910  1911  1915  1920  1925  1927
        1934  1940  1943  1950  1953  1957  1967  1973  2205  2250  2527  2528  2529  2531
.BLKB   2526
.BLKW   1576
.BYTE    200   201   206   207   222   223   224   225   252   253   263   264   271   272   274
         275   277   278   280   281   282   283   744   746   748   750   752   754   756   758
         760   762   764   766   768   770   772   774   776   778   780   782   784   786   788
         790   792   794   796   798   800   802   804   806   808   810   812   814   816   818
         820   928  1151  1297  1301  1305  1309  1403  1501  1907  1908  2145  2146  2332  2333
        2334  2335  2524  2525  2622  2623  2624
.ENABL     1
.END    2684
.ENDC      6    16   108   122   136   138   139   140   153   159   163   165   171   173   180
         194   198   200   226   230   231   232   233   237   241   263   271   274   277   280
         281   282   283   286   287   288   289   290   291   292   293   294   295   296   297
         298   299   300   301   302   303   304   305   309   326   352   357   358   361   363
         365   367   368   370   372   394   493   494   495   496   497   506   507   508   509
         510   535   536   537   538   539   587   588   589   590   591   628   629   630   631
         632   659   672   673   674   675   676   825   826   827   828   829   855   877   878
         879   880   881   896   898   899   901   904   910   913   914   918   920   926   928
         929   932  1020  1021  1022  1023  1024  1216  1217  1218  1219  1220  1338  1339  1340
        1341  1421  1422  1423  1424  1425  1448  1513  2049  2052  2057  2062  2064  2075  2078
        2079  2080  2082  2084  2091  2095  2100  2102  2106  2109  2110  2114  2117  2126  2133
        2138  2139  2140  2148  2158  2159  2163  2178  2207  2211  2220  2221  2227  2233  2234
        2244  2246  2250  2263  2341  2370  2421  2477  2496  2497  2504  2506  2509  2511  2527
        2537  2539  2572  2576  2577  2580  2607  2622  2634  2640  2643  2656  2657  2658  2659
        2660  2661  2662  2663  2664  2665
.EQUIV    16    17    25    40    41    70    71    72    73    74    75    76    77    78    79
          98    99   100   101   102   103   104   105   106   107
.EVEN    241   284   823  1979  2206  2257  2625
.IF        2    14    80   108   135   137   138   139   140   150   158   161   163   170   172
         179   193   197   199   226   230   231   232   236   237   240   263   271   274   277
         280   281   282   283   286   287   288   289   290   291   292   293   294   295   296
         297   298   299   300   301   302   303   304   305   309   326   352   357   359   361
         363   365   367   368   370   388   492   494   496   497   505   507   509   510   534
         536   538   539   586   588   590   591   627   629   631   632   658   671   673   675
         676   824   826   828   829   854   876   878   880   881   895   896   897   898   899
         900   901   903   909   912   914   918   920   926   928   929  1019  1021  1023  1024
        1215  1217  1219  1220  1337  1339  1341  1420  1422  1424  1425  1447  1512  2048  2051
        2056  2061  2062  2074  2076  2077  2078  2080  2081  2082  2091  2093  2101  2103  2108
        2109  2110  2113  2116  2126  2129  2136  2138  2139  2141  2148  2151  2158  2159  2162
        2177  2193  2210  2220  2221  2226  2233  2234  2242  2244  2246  2250  2262  2340  2361
```

```
          2420    2421    2476    2496    2504    2505    2509    2510    2526    2527    2536    2539    2551    2575    2577
          2580    2607    2622    2633    2639    2643    2647    2657    2658    2659    2660    2661    2662    2663    2664
          2665
.IFF        14     135     138     139     140     159     163     165     171     173     180     194     197     200     226
           237     241     357     493     494     495     496     497     506     507     508     509     510     535     536
           537     538     539     587     588     589     590     591     628     629     630     631     632     659     672
           673     674     675     676     825     826     827     828     829     855     877     878     879     880     881
           896     900     904     909     912     928    1020    1021    1022    1023    1024    1216    1217    1218    1219
          1220    1338    1339    1340    1341    1421    1422    1423    1424    1447    1513    2049    2075    2078    2079
          2082    2109    2110    2114    2116    2129    2158    2159    2163    2178    2207    2211    2227    2242    2263
          2341    2421    2443    2477    2479    2484    2496    2497    2506    2510    2527    2537    2576    2634    2640
.IFT      2090    2139    2421    2443    2479    2484    2555    2571    2572
.IFTF     2088    2138    2435    2449    2477    2480    2551    2555    2571
.IIF         1       6      11     132     133     134     136     139     140     147     236     241     358     361     367
           368     370     371     387     898     904     905     916     928     932    2052    2053    2054    2055    2056
          2057    2061    2089    2090    2106    2109    2110    2117    2118    2119    2120    2121    2126    2151    2158
          2159    2175    2200    2417    2421    2437    2519    2527    2533    2656    2657    2658    2659    2660    2661
          2662    2663    2664
.IRP       326     352     492     505     534     586     627     671     824     876    1019    1215    1337    1420    1523
          1563    2061    2214    2220    2233    2234    2546    2567    2581    2582    2603    2619    2620
.LIST        1     122     139     147     226     228     229     230     237     241     326     352     372     492     496
           505     509     534     538     586     590     627     631     671     675     824     828     876     880     904
           920    1019    1023    1215    1219    1337    1341    1420    1424    2056    2158    2496    2647    2656    2657
          2658    2659    2660    2661    2662    2663    2664    2665
.MACRO     190     198     388    2647
.MCALL       1     122     237     372
.NLIST       1     122     139     147     226     228     229     230     237     241     326     352     372     492     496
           505     509     534     538     586     590     627     631     671     675     824     828     876     880     904
           920    1019    1023    1215    1219    1337    1341    1420    1424    2056    2158    2496    2647    2656    2657
          2658    2659    2660    2661    2662    2663    2664    2665
.PAGE      190     236
.REPT      147     228
.SBTTL      12     128     141     151     156     168     191     238     310     492     505     534     586     627     671
           824     876     893    1019    1215    1337    1420    1510    2046    2111    2160    2208    2260    2338    2418
          2534    2573    2631    2648
.TITLE       1
.WORD      147     148     149     164     184     185     186     187     188     189     199     202     203     204     205
           208     209     210     211     212     213     214     215     216     217     226     228     229     243     244
           245     246     247     248     249     250     254     255     256     269     273     276     279     285     286
           287     288     289     290     291     292     293     294     295     296     297     298     299     300     301
           302     303     304     909     912     927    2186    2191    2243    2245    2336    2367    2414    2571    2605
```

```
ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*,DZARBB/SOL/CRF<DZARBB.P11
RUN-TIME:  55 24 6 SECONDS
RUN-TIME RATIO:  390/86=4.4
CORE USED:  26K  (51 PAGES)
```

R7