VS60

VS60 INST TST PT2

CZVSBB0

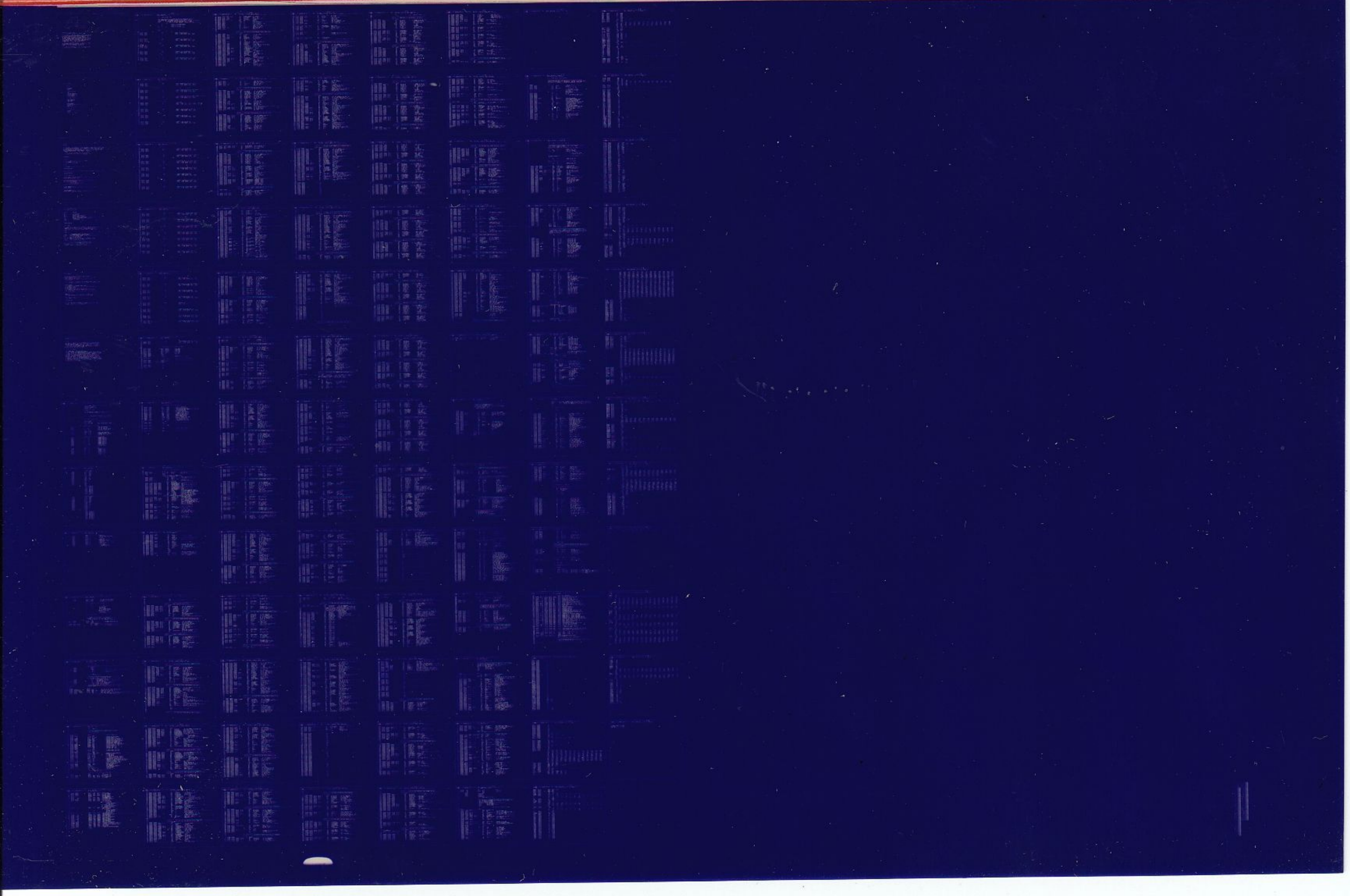
AH-9487B-MC

FICHE 1 OF 1

JAN 1980

digital

MADE IN USA



IDENTIFICATION

PRODUCT CODE: AC-9486B-MC

PRODUCT NAME: CZVSBB0 VS60 INST TST PT2

DATE: MAY 1979

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHORS: R. SHOOP & R. MOORE

0.0 TABLE OF CONTENTS

1.0 ABSTRACT

THIS PROGRAM IS THE SECOND OF A SERIES DESIGNED TO TEST STRICTLY LOGIC FUNCTIONS OF THE VS60 DISPLAY PROCESSOR. THIS DIAGNOSTIC ALL MOST EXCLUSIVELY TESTS THE VS60 GRAPHIC TYPE INSTRUCTIONS. THE BULK OF THE TESTING IS PREFORMED AT FULL SPEED WITH NO MAINTENANCE MODES ENABLED. HOWEVER, ON ODD NUMBERED PASSES, MAINTENANCE MODE SWITCH 4 (SERIAL MODE) IS ENABLED.

*** REV B INCLUDES ADDITIONAL TESTING FOR VECTOR
SCALING AFTER ECO #VT48-0012 IS INSTALLED.
G.P. MAY '79 ***

2.0 REQUIREMENTS

2.1 EQUIPMENT

A. PDP-11 COMPUTER WITH AT LEAST 8K OF MEMORY
B. I/O TERMINAL (I.E. ASR33 TTY)
C. VS60 DISPLAY PROCESSOR

2.2 STORAGE

THIS PROGRAM OCCUPIES LOWER 8K OF MEMORY.

3.0 LOADING PROCEDURE

NORMAL PROCEDURE FOR LOADING A BINARY PROGRAM INTO MEMORY SHOULD BE FOLLOWED.

4.0 STARTING PROCEDURE

4.1 STARTING ADDRESS 200

LOADING ADDRESS 200 AND STARTING WILL IDENTIFY THE TEST, INITIALIZE THE SYSTEM, AND BEGIN TESTING.

****
NOTE: SWITCH REGISTER BIT 10 MUST BE SET TO INDICATE THE PRESENCE OF ECO VT48-0012.
0 = ECO NOT INSTALLED (DEFAULT).
1 = ECO IS INSTALLED.
****

4.2 RESTART ADDRESS 204

LOADING ADDRESS 204 AND STARTING WILL INITIALIZE THE SYSTEM AND BEGIN TESTING.

4.3 STARTING ADDRESS 210

LOADING ADDRESS 210 AND STARTING IS IDENTICAL TO START 200 EXCEPT THAT EXTERNAL SYNC TIMING WILL BE TESTED.

5.0 SWITCH REGISTER SETTINGS

SWITCH	FUNCTION
SW15 = 1	HALT ON ERROR
SW14 = 1	LOOP ON TEST
SW13 = 1	INHIBIT ERROR TYPEOUTS
SW12 = 1	HALT AT END OF DIAGNOSTIC
SW11 = 1	INHIBIT ITERATIONS - NORMALLY 2000(8) AFTER 1ST PASS
SW10 = 1	SCALING ECO (VT48-0012) IS INSTALLED.
SW09 = 1	LOOP ON ERROR
SW08 = 1	LOOP ON TEST IN SWR<7:0>

6.0 ERROR REPORTING

6.1 ERROR COMMENT

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE COMMENT AS TO THE TYPE OF FAILURE. FURTHER QUALIFICATION OF THE ERROR CAN BE OBTAINED IF NEEDED FROM THE COMMENT AT THE ERROR PC.

6.2 ERROR DATA

PC	LISTING ADDRESS WHERE THE ERROR WAS DETECTED.
TSTNUM	TEST NUMBER WHERE THE ERROR OCCURRED.
BUSADRS	DISPLAY BUS ADDRESS WHERE THE RESULTANT DATA WAS EXPECTED
EXPCT	DATA THAT WAS EXPECTED.
RCVD	DATA THAT WAS RECEIVED.
TSTPC	ADDRESS OF TEST WHICH FAILED TO COMPLETE (HUNG)
SCALE	VALUE OF SCALE (1/4 TO 3 3/4) WHEN ERROR OCCURED
STARTVAL	X OR Y VALUE THAT WAS SCALED

****
NOTE: IF YOU GET SCALING ERRORS (SCALE ER), CHECK
THAT SWR BIT 10 = 1 IF ECO VT48-0012 IS INSTALLED.
OTHERWISE, SWR BIT 10 SHOULD BE 0 (DEFAULT).
****

7.0 MISCELLANEOUS

7.1 VS60 BUS/VECTOR/PRIORITY ADDRESS MODIFICATION:
MODIFY LOCATION 1252 ($VECT1) IF BASE VECTOR PRIORITY/ADDRESS
IS NOT 100320 (HI BYTE = PRI 4).
MODIFY LOCATION 1256 ($BASE) IF BASE REGISTER ADDRESS
IS NOT 172000.

NOTE: A RESTART IS REQUIRED AFTER THE ABOVE ADDRESS MODIFICATION.
THE ABOVE LOCATIONS ARE LOCATED IN THE 'APT MAILBOX-ETABLE'.

7.2 XXDP/APT NOTES:
THIS DIAGNOSTIC IS CHAINABLE UNDER XXDP.
THIS DIAGNOSTIC INCLUDES THE 'APT' SOFTWARE HOOKS HOWEVER, THEY
HAVE NOT BEEN TESTED.
****
THIS DIAGNOSTIC SOULD NOT BE CHAINED AFTER ECO VT48-0012
SINCE THE SETTING OF SWR BIT 10 MAY CONFLICT WITH OTHER
TESTS IN THE CHAIN.
****

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

7.4 SINGLE VS60 TESTING:
THIS DIAGNOSTIC DOES NOT TEST MULTIPLE VS60'S.

8.0 EXECUTION TIME

EXECUTION TIME RANGES FROM ABOUT 3 SECONDS WITH NO ITERATIONS
TO ABOUT 4 MINUTES WITH ITERATIONS ENABLED. AN 'END PASS' MESSAGE
IS TYPED EACH PASS THRU THE DIAGNOSTIC.

9.0 PROGRAM TEST DESCRIPTIONS

9.1 GENERAL

THE TABLE OF CONTENTS AT THE BEGINNING OF THE LISTING CONTAINS AT BRIEF DESCRIPTION OF EACH TEST. ADDITIONAL INFORMATION CAN BE OBTAINED FROM THE COMMENTS SUPPLIED AT EACH TEST. THE MAJORITY OF THE TESTS IN THIS DIAGNOSTIC ARE CONCERNED WITH THE GRAPHIC OR DISPLAY INSTRUCTIONS WHERE AT COMPLETION THE RESULTS CAN BE READ FROM THE X OR Y POSITION REGISTERS.

9.2 TYPICAL TEST SEQUENCE

TEST FORMAT MIGHT LOOK LIKE LIKE THIS:
LOCATION 'BUFFER' IS LOADED WITH A GRAPHIC INSTRUCTION OR INSTRUCTIONS THAT WILL PROVIDE THE EXPECTED RESULT. A DISPLAY STOP INSTRUCTION IS LOADED AS THE LAST INSTRUCTION IN 'BUFFER'. '$BDADR' IS LOADED WITH THE X OR Y REGISTER ADDRESS WHERE THE EXPECTED RESULT IS EXPECTED. LOCATION '$GDDAT' IS LOADED WITH THE EXPECTED RESULT AND THEN THE DISPLAY IS STARTED AT LOCATION 'BUFFER'. WHEN THE DISPLAY STOP FLAG IS SET THE APPROPRIATE VS60 BUS ADDRESS DEFINED BY '$BDADR' IS READ AND COMPARED TO '$GDDAT'. IF AN ERROR IS DETECTED AN ERROR TRAP (EMP INSTR) IS PREFORMED ACCORDING TO THE ITEM NUMBER IN THE '$ERRTB' (ERROR TABLE). THIS TABLE DIRECTS OR POINTS TO THE PERTINENT 'TYPEOUT' INFORMATION.

10.0 LISTING

PROGRAM LISTING FOLLOWS.

```
 13                              .TITLE  CZVSBB -- VS60 INSTRUCTION TEST PART II
(1)                              ;*COPYRIGHT (C) 1979
(1)                              ;*DIGITAL EQUIPMENT CORP.
(1)                              ;*MAYNARD, MASS. 01754
(1)                              ;*
(1)                              ;*PROGRAM BY R. SHOOP
(1)                              ;*
(1)                              ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
(1)                              ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
(1)                              ;*
 14                              ;
 15                              ; REV B -- FIXES SCALING TESTS AFTER ECO VT48-0012.
 16                              ;          SET SWR BIT 10 (002000) IF ECO IS INSTALLED.
 17                              ;                                  G.P.  MAY '79
 18                              ;
 19                              .SBTTL  BASIC DEFINITIONS
(1)
(1)                              ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
(1)          001100              STACK=  1100
(1)                              .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
(1)                              .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
(1)
(1)                              ;*MISCELLANEOUS DEFINITIONS
(1)          000011              HT=     11              ;;CODE FOR HORIZONTAL TAB
(1)          000012              LF=     12              ;;CODE FOR LINE FEED
(1)          000015              CR=     15              ;;CODE FOR CARRIAGE RETURN
(1)          000200              CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
(1)          177776              PS=     177776          ;;PROCESSOR STATUS WORD
(1)                              .EQUIV  PS,PSW
(1)          177774              STKLMT= 177774          ;;STACK LIMIT REGISTER
(1)          177772              PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
(1)          177570              DSWR=   177570          ;;HARDWARE SWITCH REGISTER
(1)          177570              DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
(1)
(1)                              ;*GENERAL PURPOSE REGISTER DEFINITIONS
(1)          000000              R0=     %0              ;;GENERAL REGISTER
(1)          000001              R1=     %1              ;;GENERAL REGISTER
(1)          000002              R2=     %2              ;;GENERAL REGISTER
(1)          000003              R3=     %3              ;;GENERAL REGISTER
(1)          000004              R4=     %4              ;;GENERAL REGISTER
(1)          000005              R5=     %5              ;;GENERAL REGISTER
(1)          000006              R6=     %6              ;;GENERAL REGISTER
(1)          000007              R7=     %7              ;;GENERAL REGISTER
(1)          000006              SP=     %6              ;;STACK POINTER
(1)          000007              PC=     %7              ;;PROGRAM COUNTER
(1)
(1)                              ;*PRIORITY LEVEL DEFINITIONS
(1)          000000              PR0=    0               ;;PRIORITY LEVEL 0
(1)          000040              PR1=    40              ;;PRIORITY LEVEL 1
(1)          000100              PR2=    100             ;;PRIORITY LEVEL 2
(1)          000140              PR3=    140             ;;PRIORITY LEVEL 3
(1)          000200              PR4=    200             ;;PRIORITY LEVEL 4
(1)          000240              PR5=    240             ;;PRIORITY LEVEL 5
(1)          000300              PR6=    300             ;;PRIORITY LEVEL 6
(1)          000340              PR7=    340             ;;PRIORITY LEVEL 7
(1)
```

```
(1)                             ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
(1)             100000          SW15=   100000
(1)             040000          SW14=   40000
(1)             020000          SW13=   20000
(1)             010000          SW12=   10000
(1)             004000          SW11=   4000
(1)             002000          SW10=   2000
(1)             001000          SW09=   1000
(1)             000400          SW08=   400
(1)             000200          SW07=   200
(1)             000100          SW06=   100
(1)             000040          SW05=   40
(1)             000020          SW04=   20
(1)             000010          SW03=   10
(1)             000004          SW02=   4
(1)             000002          SW01=   2
(1)             000001          SW00=   1
(1)                             .EQUIV  SW09,SW9
(1)                             .EQUIV  SW08,SW8
(1)                             .EQUIV  SW07,SW7
(1)                             .EQUIV  SW06,SW6
(1)                             .EQUIV  SW05,SW5
(1)                             .EQUIV  SW04,SW4
(1)                             .EQUIV  SW03,SW3
(1)                             .EQUIV  SW02,SW2
(1)                             .EQUIV  SW01,SW1
(1)                             .EQUIV  SW00,SW0
(1)
(1)                             ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)             100000          BIT15=  100000
(1)             040000          BIT14=  40000
(1)             020000          BIT13=  20000
(1)             010000          BIT12=  10000
(1)             004000          BIT11=  4000
(1)             002000          BIT10=  2000
(1)             001000          BIT09=  1000
(1)             000400          BIT08=  400
(1)             000200          BIT07=  200
(1)             000100          BIT06=  100
(1)             000040          BIT05=  40
(1)             000020          BIT04=  20
(1)             000010          BIT03=  10
(1)             000004          BIT02=  4
(1)             000002          BIT01=  2
(1)             000001          BIT00=  1
(1)                             .EQUIV  BIT09,BIT9
(1)                             .EQUIV  BIT08,BIT8
(1)                             .EQUIV  BIT07,BIT7
(1)                             .EQUIV  BIT06,BIT6
(1)                             .EQUIV  BIT05,BIT5
(1)                             .EQUIV  BIT04,BIT4
(1)                             .EQUIV  BIT03,BIT3
(1)                             .EQUIV  BIT02,BIT2
(1)                             .EQUIV  BIT01,BIT1
(1)                             .EQUIV  BIT00,BIT0
(1)
```

```
(1)                                 ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
(1)             000004              ERRVEC= 4                   ;;TIME OUT AND OTHER ERRORS
(1)             000010              RESVEC= 10                  ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)             000014              TBITVEC=14                  ;;''T'' BIT
(1)             000014              TRTVEC= 14                  ;;TRACE TRAP
(1)             000014              BPTVEC= 14                  ;;BREAKPOINT TRAP (BPT)
(1)             000020              IOTVEC= 20                  ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)             000024              PWRVEC= 24                  ;;POWER FAIL
(1)             000030              EMTVEC= 30                  ;;EMULATOR TRAP (EMT) **ERROR**
(1)             000034              TRAPVEC=34                  ;;''TRAP'' TRAP
(1)             000060              TKVEC=  60                  ;;TTY KEYBOARD VECTOR
(1)             000064              TPVEC=  64                  ;;TTY PRINTER VECTOR
(1)             000240              PIRQVEC=240                 ;;PROGRAM INTERRUPT REQUEST VECTOR
```

```
                                                                K 1
                    172000                  ABASE=  172000          ; VS60 BASE REGISTER ADDRESS.
                    100320                  AVECT1= 100320          ; PRI AND 1ST VECTOR ADDRESS.
                    165400                  $SWR=   165400          ; REDEFINE SWITCHES USED.

                                            .SBTTL  OPERATIONAL SWITCH SETTINGS
                                            ;*
                                            ;*      SWITCH                  USE
                                            ;*      ------          --------------------
                                            ;*        15            HALT ON ERROR
                                            ;*        14            LOOP ON TEST
                                            ;*        13            INHIBIT ERROR TYPEOUTS
                                            ;*        12            HALT AT END OF DIAGNOSTIC
                                            ;*        11            INHIBIT ITERATIONS
                                            ;*        10            SCALING ECO IS INSTALLED
                                            ;*         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  002050                             JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
 000204  000137  002460                             JMP     RESTRT          ; 204 = RESTART.
 000210  000137  002040                             JMP     BEGINA          ; 210 = START W/ EXTERNAL SYNC.
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
                                        $SVPC=.                 ;SAVE PC
000214                                  .=46
000046                                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000046  024470                          .=52
000052                                  .WORD   0               ;;2)SET LOC.52 TO ZERO
000052  000000                          .=$SVPC                 ;; RESTORE PC
000214

                                        .=1000
001000                          .SBTTL  APT PARAMETER BLOCK

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

001000                          $APTHD:
001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001176                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000012                  $TSTM:  .WORD   10.     ;;RUN TIM OF LONGEST TEST
001006  000036                  $PASTM: .WORD   30.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000000                  $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000032                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  .WORD   0
001102     000          $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103     000          $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114     000          $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115     001          $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134     000          $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135     000          $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154     000          $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155     002          $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156     012          $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
001157     000          $TPFLG: .BYTE   0               ;;'TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
001160  000000          $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REG0) WAS OBTAINED
001162  000000          $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000          $TMP0:  .WORD   0               ;;USER DEFINED
001166  000000          $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001170  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001172     077          $QUES:  .ASCII  /?/             ;;QUESTION MARK
001173     015          $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001174  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;*****************************************************************
                                .EVEN
001176                  $MAIL:                          ;;APT MAILBOX
001176  000000          $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001200  000000          $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001202  000000          $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001204  000000          $PASS:  .WORD   APASS           ;;PASS COUNT
```

```
(2)  001206  000000                $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
(2)  001210  000000                $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
(2)  001212  000000                $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
(2)  001214  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
(2)  001216                        $ETABLE:                ;;APT ENVIRONMENT TABLE
(2)  001216     000                $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
(2)  001217     000                $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
(2)  001220  000000                $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
(2)  001222  000000                $USWR:  .WORD   AUSWR   ;;USER SWITCHES
(2)  001224  000000                $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
(2)                                ;*                      BITS 15-11=CPU TYPE
(2)                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(2)                                ;*                              11/70=06,PDQ=07,Q=10
(2)                                ;*                      BIT 10=REAL TIME CLOCK
(2)                                ;*                      BIT  9=FLOATING POINT PROCESSOR
(2)                                ;*                      BIT  8=MEMORY MANAGEMENT
(2)  001226     000                $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
(2)  001227     000                $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
(2)                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
(2)                                ;*                              900 NSEC CORE=001
(2)                                ;*                              300 NSEC BIPOLAR=002
(2)                                ;*                              500 NSEC MOS=003
(2)  001230  000000                $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
(2)                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
(2)  001232     000                $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
(2)  001233     000                $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
(2)  001234  000000                $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
(2)  001236     000                $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
(2)  001237     000                $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
(2)  001240  000000                $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
(2)  001242     000                $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
(2)  001243     000                $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
(2)  001244  000000                $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
(2)  001246  100320                $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)  001250  000000                $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(2)  001252  172000                $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)  001254  000000                $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
(2)  001256  000000                $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
(2)  001260  000000                $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
(2)  001262                        $ETEND:
(2)                                .MEXIT
```

```
                                                                B 2
                                .SBTTL  ERROR POINTER TABLE

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

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


001262                          $ERRTB:
                                ;ITEM   1

001262  030061                          EM1                     ;MAINT. SWITCH REGISTER
001264  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001266  032470                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001270  000000                          0

                                ;ITEM   2

001272  030103                          EM2                     ;LOADING D.P.C. REGISTER <STATIC>
001274  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001276  032470                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001300  000000                          0

                                ;ITEM   3

001302  030137                          EM3                     ;LOADING RELOCATE REGISTER
001304  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001306  032470                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001310  000000                          0

                                ;ITEM   4

001312  030164                          EM4                     ;READING D.P.C. WITH RELOCATE SET
001314  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001316  032470                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001320  000000                          0

                                ;ITEM   5

001322  030231                          EM5                     ;OFFSET INDICATOR FAILED TO SET/CLEAR
001324  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001326  032470                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001330  000000                          0
```

```
                                ;ITEM   6
                                        EM6             ;OFFSET POLARITY BIT FAILED TO SET/CLEAR
001332  030270                  DH1     DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001334  032323                  DT1     DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001336  032470                  0       0
001340  000000

                                ;ITEM   7
                                        EM7             ;OFFSET INSTRUCTION FAILED TO LOAD DYNAMIC REGISTER
001342  030332                  DH1     DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001344  032323                  DT1     DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001346  032470                  0       0
001350  000000

                                ;ITEM   10
                                        EM10            ;OFFSET INSTRUCTION FAILED TO LOAD EXTENDED POSITION BIT
001352  030402                  DH1     DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001354  032323                  DT1     DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001356  032470                  0       0
001360  000000

                                ;ITEM   11
                                        EM11            ;SCALE ERROR
001362  030460                  DH11    DH11            ;ERRPC  TSTNUM  EXPCT   RCVD    SCALE   STARTVAL
001364  032406                  DT11    DT11            ;$ERRPC TSTNUM  $GDDAT  $BDDAT  $TMPC   $REG0
001366  032512                  0       0
001370  000000

                                ;ITEM 12
                                        EM12            ;ABSOLUTE VECTOR FAILED TO LOAD X OR Y POS REG
001372  030471                  DH1     DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001374  032323                  DT1     DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001376  032470                  0       0
001400  000000

                                ;ITEM   13
                                        EM13            ;GRAPHPLOT INC REG ER
001402  030544                  DH1     DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001404  032323                  DT1     DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001406  032470                  0       0
001410  000000

                                ;ITEM   14
                                        EM14            ;LOADING X POSITION REGISTER
001412  030571                  DH1     DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001414  032323                  DT1     DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001416  032470                  0       0
001420  000000
```

```
                                ;ITEM   15

001422  030613                          EM15                    ;LOADING Y POSITION REGISTER
001424  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001426  032470                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001430  000000                          0

                                ;ITEM   16

001432  030635                          EM16                    ;INCREMENTING DELTA X OR DELTA Y REGISTERS
001434  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001436  032470                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001440  000000                          0

                                ;ITEM   17

001442  030701                          EM17                    ;DECREMENTING DELTA X OR DELTA Y REGISTER
001444  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001446  032470                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001450  000000                          0

                                ;ITEM   20

001452  030745                          EM20                    ;EDGE INDICATOR UPON EXCEEDING AN EDGE
001454  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001456  032470                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001460  000000                          0

                                ;ITEM   21

001462  031013                          EM21                    ;LOADING CHARACTER REGISTER
001464  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001466  032470                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001470  000000                          0

                                ;ITEM   22

001472  031034                          EM22                    ;INCORRECT OR UNEXPECTED CHANGE IN X OR Y POSITION REGIS
001474  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001476  03247C                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001500  000000                          0

                                ;ITEM   23

001502  031123                          EM23                    ;WRONG DELTA X - SCALED CHARACTER MODE
001504  032323                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001506  032470                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001510  000000                          0
```

```
                                ;ITEM   24
                                
001512  031164                  EM24            ;WRONG DELTA Y - SCALED CHARACTER MODE
001514  032323                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001516  032470                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001520  000000                  0

                                ;ITEM   25

001522  031225                  EM25            ;WRONG DELTA Y - SCALED CHARACTE MODE ROTATED
001524  032323                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001526  032470                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001530  000000                  0

                                ;ITEM   26

001532  031276                  EM26            ;WRONG DELTA X - SCALED CHARACTE MODE ROTATED
001534  032323                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001536  032470                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001540  000000                  0

                                ;ITEM   27

001542  031347                  EM27            ;GRAPHPLOT INC FAILED TO SET UP X OR Y POS REG
001544  032323                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001546  032470                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001550  000000                  0

                                ;ITEM   30

001552  031425                  EM30            ;BASIC VECTOR FAILED TO LOAD X OR Y POS REG
001554  032323                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001556  032470                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001560  000000                  0

                                ;ITEM   31

001562  031500                  EM31            ;BASIC SHORT VECTOR FAILED TO LOAD X OR Y POS REG
001564  032323                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001566  032470                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001570  000000                  0

                                ;ITEM   32

001572  031561                  EM32            ;RESET FAILED TO CLEAR A REGISTER
001574  032323                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001576  032470                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001600  000000                  0
```

```
                                ;ITEM   33
                                
001602  031615                        EM33                    ;EDGE FLAG INTR ER
001604  032323                        DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001606  032470                        DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001610  000000                        0

                                ;ITEM   34

001612  031637                        EM34                    ;INTR PRIORTY FAILURE USING INTERNAL STOP
001614  032323                        DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001616  032470                        DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001620  000000                        0

                                ;ITEM   35

001622  031711                        EM35                    ;SHIFT-OUT BIT SET IN ERROR
001624  032323                        DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001626  032470                        DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001630  000000                        0

                                ;ITEM   36

001632  031744                        EM36                    ;SHIFT-OUT BIT FAILED TO SET
001634  032323                        DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001636  032470                        DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001640  000000                        0

                                ;ITEM   37

001642  032000                        EM37                    ;INTERNAL STOP FLAG FAILED TO SET
001644  032370                        DH2                     ;ERRPC  TSTPC
001646  032504                        DT2                     ;$ERRPC $LPADR
001650  000000                        0

                                ;ITEM   40

001652  032041                        EM40                    ;'FRAMES PER SECOND' SYNC ER
001654  032323                        DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001656  032470                        DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001660  000000                        0

                                ;ITEM   41

001662  032075                        EM41                    ;LONG VECTOR FAILED TO SET UP X OR Y POS REG
001664  032323                        DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001666  032470                        DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001670  000000                        0

                                ;ITEM   42
001672  032151                        EM42                    ;POINT OR ABS VECTOR WITH OFFSET FAILED TO SET UP X OR Y
001674  032323                        DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001676  032470                        DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001700  000000                        0

                                ;ITEM   43
```

```
                                        EM43                    ;SUPER/SUBSCRIPT CHAR SCALE OR DELTA X OR Y ER
001702  032245                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001704  032323                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001706  032470                          0
001710  000000

                                ;COMMON PROGRAM TAGS AND STORAGE LOCATIONS

001712  032530                  DBUF:   BUFFER          ;FIRST WORD IN THE DISPLAY BUFFER
001714  032532                  DBUF1:  BUFFER+2        ;SECOND WORD
001716  032534                  DBUF2:  BUFFER+4        ;THIRD WORD
001720  032536                  DBUF3:  BUFFER+6        ;FOURTH WORD
001722  032540                  DBUF4:  BUFFER+10       ;FIFTH WORD
001724  032542                  DBUF5:  BUFFER+12       ;SIXTH WORD
001726  000000                  DSAVE:  0               ;TEMP REG.
001730  000000                  DSAVE1: 0
001732  000000                  SIZE:   0               ;BUFFER SIZE
001734  000000                  TSTNUM: 0               ;CONTAINS TEST NO. ON ERROR

001736  000200                  BRLEV1: 200             ;CONTAINS BR LEVEL TAKEN FROM $VECT1.

001740  000007                  CHSZ0:  7.              ;CHARACTER SCALE CHARACTER SIZE VALUES
001742  000016                  CHSZ1:  14.
001744  000025                  CHSZ2:  21.
001746  000034                  CHSZ3:  28.

001750  000014                  LFSZ0:  12.             ;CHARACTER SCALE LINE FEED SIZE VALUES
001752  000030                  LFSZ1:  24.
001754  000056                  LFSZ2:  46.
001756  000076                  LFSZ3:  62.
```

```
                                                    H 2
                                ;VS-60 ADDRESSES AND VECTORS POINTERS

001760  172000                  DPC:    172000      ;DISPLAY PC REGISTER
001762  172002                  SREG0:  172002      ;DISPLAY STATUS REGISTER
001764  172004                  XPOS:   172004      ;X AXIS REGISTER AND GRAPHPLOT REG <READ ONLY>
001766  172006                  YPOS:   172006      ;Y AXIS REGISTER AND CHARACTER REG <READ ONLY>
001770  172010                  RLO:    172010      ;RELOCATION REGISTER
001772  172012                  SREG1:  172012      ;MISC STATUS REGISTER #1
001774  172014                  XDOFF:  172014      ;X DYNAMIC OFFSET REGISTER
001776  172016                  YDOFF:  172016      ;Y DYNAMIC OFFSET REGISTER
002000  172020                  ANAME:  172020      ;ASSOCIATIVE NAME REGISTER
002002  172022                  CONS:   172022      ;CONSOLE STATUS REGISTER
002004  172024                  DNAME:  172024      ;DISPLAY NAME REGISTER
002006  172026                  STKVAL: 172026      ;VALUE OF CURRENT STACK WORD
002010  172030                  TERMCH: 172030      ;TERMINATE CHARACTER REGISTER
002012  172032                  STKPT:  172032      ;STACK POINTER REGISTER
002014  172034                  ZPOS:   172034      ;Z POSITION REGISTER <READ ONLY>
002016  172036                  ZDOFF:  172036      ;Z DYNAMIC OFFSET REGISTER

002020  000320                  DDONE:  320         ;DISPLAY STOP <DONE> VECTOR
002022  000322                  DDONE1: 322         ;

002024  000324                  LPVCT:  324         ;DISPLAY LIGHT PEN VECTOR
002026  000326                  LPVCT1: 326         ;

002030  000330                  TIMEVT: 330         ;DISPLAY TIME-OUT <NXM.> ERROR VECTOR
002032  000332                  TMEVT1: 332         ; OR ''SHIFT-OUT'' VECTOR

002034  000334                  NAMEVT: 334         ;NAME MATCH VECTOR
002036  000336                  NAMEV1: 336
```

```
                                                                I 2
  355                                   ;SOFTWARE INITALIZATION ROUTINE
  356
  358  002040  012737  177777  024356   BEGINA: MOV     #-1,EXSYNC      ;CHECK FOR EXT SYNC TMG (START 210)
  359  002046  000402                           BR      BEGIN1          ;SKIP OVER CLR
  360  002050  005037  024356           BEGIN:  CLR     EXSYNC          ;DON'T CHECK FOR EXT TMG (START 200)
  361  002054                           BEGIN1:
  (1)                                   .SBTTL  INITIALIZE THE COMMON TAGS
  (1)                                   ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
  (1)  002054  012706  001100                   MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
  (1)  002060  005026                           CLR     (R6)+           ;;CLEAR MEMORY LOCATION
  (1)  002062  022706  001140                   CMP     #SWR,R6 ;;DONE?
  (1)  002066  001374                           BNE     .-6             ;;LOOP BACK IF NO
  (1)  002070  012706  001100                   MOV     #STACK,SP       ;;SETUP THE STACK POINTER
  (1)                                   ;;INITIALIZE A FEW VECTORS
  (1)  002074  012737  025164  000020           MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
  (1)  002102  012737  000340  000022           MOV     #340,@#IOTVEC+2 ;;LEVEL 7
  (1)  002110  012737  025710  000030           MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
  (1)  002116  012737  000340  000032           MOV     #340,@#EMTVEC+2 ;;LEVEL 7
  (1)  002124  012737  027616  000034           MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)  002132  012737  000340  000036           MOV     #340,@#TRAPVEC+2;LEVEL 7
  (1)  002140  012737  026704  000024           MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
  (1)  002146  012737  000340  000026           MOV     #340,@#PWRVEC+2 ;;LEVEL 7
  (1)  002154  013737  024436  024430           MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
  (1)  002162  005037  001166                   CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
  (1)  002166  005037  001170                   CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
  (1)  002172  112737  000001  001115           MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
  (1)  002200  012737  002200  001106           MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
  (1)  002206  012737  002206  001110           MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
  (2)                                   ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
  (2)                                   ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
  (2)  002214  013746  000004                   MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
  (2)  002220  012737  002254  000004           MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
  (2)  002226  012737  177570  001140           MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
  (2)  002234  012737  177570  001142           MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
  (2)  002242  022777  177777  176670           CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
  (2)  002250  001012                           BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  (2)                                                                   ;;AND  THE HARDWARE SWR IS NOT = -1
  (2)  002252  000403                           BR      65$             ;;BRANCH IF NO TIMEOUT
  (2)  002254  012716  002262           64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
  (2)  002260  000002                           RTI
  (2)  002262  012737  000176  001140   65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
  (2)  002270  012737  000174  001142           MOV     #DISPREG,DISPLAY
  (2)  002276  012637  000004           66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
  (1)
  (2)  002302  005037  001204                   CLR     $PASS           ;;CLEAR PASS COUNT
  (2)  002306  132737  000200  001217           BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
  (2)  002314  001403                           BEQ     67$             ;;YES,USE NON-APT SWITCH
  (2)  002316  012737  001220  001140           MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
  (2)  002324                           67$:
  362  002324  013700  001246   SETUP1: MOV     $VECT1,R0       ; GET BUS PRIORITY LEVEL.
  363  002330  000300                   SWAB    R0
  364  002332  042700  177437           BIC     #^C340,R0       ; MASK PSW BITS
  365  002336  010037  001736           MOV     R0,BRLEV1       ; AND SET BR LEVEL.
  366  002342  012700  001760           MOV     #DPC,R0         ;SET UP VS60 REG ADRS POINTERS
  367  002346  013701  001252           MOV     $BASE,R1
  368  002352  010120           SETUP2: MOV     R1,(R0)+
```

```
   369  002354  062701  000002                    ADD     #2,R1
   370  002360  022700  002020                    CMP     #DDONE,R0       ;TEST FOR LAST ADDRESS
   371  002364  001372                            BNE     SETUP2
   372  002366  012700  002020                    MOV     #DDONE,R0       ;SET UP VS60 VECTOR ADRS POINTERS
   373  002372  013701  001246                    MOV     $VECT1,R1
   374  002376  042701  160000                    BIC     #160000,R1      ;MASK OF BITS OF PRIORITY
   375  002402  010120                  SETUP3:   MOV     R1,(R0)+
   376  002404  062701  000002                    ADD     #2,R1
   377  002410  022700  002040                    CMP     #BEGINA,R0      ;TEST FOR LAST VECTOR
   378  002414  001372                            BNE     SETUP3
   379  002416  004737  024634                    JSR     PC,RSTVEC       ;GO SET UP VS60 VECTOR ADRS'S WITH HLTS
   380  002422  004737  027124                    JSR     PC,$SIZE        ;DETERMINE LAST LOCATION
   381  002426  013737  027220  001732            MOV     $LSTAD,SIZE     ;LOAD LAST BUS ADDRESS
   382  002434  162737  006000  001732            SUB     #6000,SIZE      ;SAVE LOADERS AND XXDP
   383  002442  005737  000042                    TST     @#42            ;TEST IF RUNNING CHAIN MODE UNDER XXDP
   384  002446  001004                            BNE     RESTRT          ;BR IF SO
   385  002450  104401                            TYPE                    ;GO IDENTIFY TEST
   386  002452  027666                            HEADER
   387  002454  104401                            TYPE
   388  002456  027752                            ECOMSG                  ; AND TELL HIM ABOUT THE ECO.
   389  002460  012737  000340  177776  RESTRT:   MOV     #340,PSW        ;SET PRIORITY TO HIGHEST LEVEL
   390  002466  012706  001100                    MOV     #STACK,SP       ;ALWAYS RESET STACK POINTER
   391  002472  000005                            RESET                   ;INITILIZE VS60 BEFORE STARTING TESTS
   392  002474  105037  001102                    CLRB    $TSTNM          ;RESET TEST NUMBER
   393
```

```
                                                    .SBTTL
                                                    .SBTTL  THESE TESTS USE MAINT SWITCH BITS
                                                    .SBTTL

                                                ;;******************************************************************
                                                ;*TEST 1         FLOAT A ONE THRU RELOCATE REG
                                                ;;******************************************************************
002500  000004                                  TST1:   SCOPE
002502  012737  000001  001202                          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
002510  012737  002532  001110                          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
002516  013737  001770  001122                          MOV     RLO,$BDADR      ;SET UP RLO ADRS
002524  012737  004000  001124                          MOV     #4000,$GDDAT    ;SET UP MSB TO A ONE
002532  013777  001124  177230  1$:             MOV     $GDDAT,@RLO     ;LOAD UP RLO REG
002540  017737  177224  001126                          MOV     @RLO,$BDDAT     ;READ IT BACK
002546  023737  001124  001126                          CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
002554  001401                                        BEQ     2$              ;BR IF OK
002556  104003                                        ERROR   3               ;RELOCATE REG READ/WRITE ER
002560  006237  001124          2$:             ASR     $GDDAT          ;SHIFT ONE BIT RIGHT
002564  001362                                        BNE     1$              ;BR IF NOT DONE
002566  005077  177176                                CLR     @RLO            ;INSURE 0 RELOCATE REG BEFORE ADVANCING

                                                ;;******************************************************************
                                                ;*TEST 2         TEST THAT ALL MAINT TYPE BITS ARE READ/WRITEABLE
                                                ;;******************************************************************
002572  000004                                  TST2:   SCOPE
002574  012737  000002  001202                          MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
002602  013737  002012  001122                          MOV     STKPT,$BDADR    ;SET UP REG 32 ADRS
002610  012737  002624  001110                          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
002616  012737  170000  001124                          MOV     #170000,$GDDAT  ;EXPECT ALL BITS INITIALLY
002624  013777  001124  177160  1$:             MOV     $GDDAT,@STKPT   ;LOAD UP MAINT BITS
002632  017737  177154  001126                          MOV     @STKPT,$BDDAT   ;READ THEM BACK
002640  042737  007777  001126                          BIC     #7777,$BDDAT    ;SAVE ONLY MAINT SWITCHES
002646  023737  001124  001126                          CMP     $GDDAT,$BDDAT   ;ARE THEY CORRECT?
002654  001401                                        BEQ     2$              ;BR IF SO
002656  104001                                        ERROR   1               ;MAINT SWITCH READ/WRITE ERROR
002660  162737  010000  001124  2$:             SUB     #10000,$GDDAT   ;GO TO NEXT PATTERN
002666  100356                                        BPL     1$              ;BR IF ALL STATES NOT TESTED

                                                ;;******************************************************************
                                                ;*TEST 3         FLOAT A 1 ACROSS THE D.P.C.
                                                ;;******************************************************************
002670  000004                                  TST3:   SCOPE
002672  012737  000003  001202                          MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
002700  013737  001760  001122                          MOV     DPC,$BDADR      ;SET UP REG ADRS 00
002706  012737  002730  001110                          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
002714  012737  000002  001124                          MOV     #BIT1,$GDDAT    ;LOAD EXPECTED
002722  012777  010000  177062                          MOV     #BIT12,@STKPT           ;SET MAINT SWITCH #1
002730  013777  001124  177022  1$:             MOV     $GDDAT,@DPC     ;LOAD D.P.C.
002736  017737  177016  001126                          MOV     @DPC,$BDDAT     ;READ D.P.C. INTO $BDDAT
002744  023737  001124  001126                          CMP     $GDDAT,$BDDAT   ;COMPARE VALUE EXPECTED TO READ
002752  001401                                        BEQ     2$              ;;BR IF SAME
002754  104002                                        ERROR   2               ;FAILED TO LOAD DPC WITH A
                                                                                ;FLOATING 1
002756  006337  001124          2$:             ASL     $GDDAT          ;CHANGE DATA EXPECTED
002762  103362                                        BCC     1$              ;BR IF NOT COMPLETED
```

```
                                ;;*******************************************************************
                                ;*TEST 4         TEST THAT THE RELOCATE REG WILL CORRECTLY SET UP THE DPC
                                ;;*******************************************************************
002764  000004                  TST4:   SCOPE
002766  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
002774  012737  000004  001202          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
003002  012777  010000  177002          MOV     #BIT12,@STKPT   ;SET MAINT SW 1
003010  013737  001760  001122          MOV     DPC,$BDADR      ;SET UP REG 00 ADRS
003016  005077  176736                  CLR     @DPC            ;INSURE 0 DPC
003022  012737  000100  001124          MOV     #100,$GDDAT     ;EXPECT 100 INITIALLY
003030  012700  000001                  MOV     #1,R0           ;START WITH 1
003034  010077  176730          1$:     MOV     R0,@RLO         ;LOAD RELOCATE REG
003040  012737  003034  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
003046  017737  176706  001126          MOV     @DPC,$BDDAT     ;READ RELOCATE REG-SB TIMES 64
003054  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
003062  001401                          BEQ     2$              ;BR IF SO
003064  104004                          ERROR   4               ;RELOCATE FAILED TO SET UP DPC CORRECTLY
003066  062737  000100  001124  2$:     ADD     #100,$GDDAT     ;SET UP NEXT EXPECTED VALUE
003074  001402                          BEQ     3$              ;BR IF ALL 15 BITS TESTED
003076  005200                          INC     R0              ;SET UP NEXT PATTERN
003100  000755                          BR      1$              ;GO TRY IT
003102  005077  176662          3$:     CLR     @RLO            ;ZERO RELOCATE BEFORE ADVANCING

                                ;;*******************************************************************
                                ;*TEST 5         TEST THAT THE OFFSET INDICATOR WILL SET & RESET
                                ;;*******************************************************************
003106  000004                  TST5:   SCOPE
003110  012737  000005  001202          MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
003116  012777  020000  176666          MOV     #20000,@STKPT   ;SET MAINT SW 2
003124  012737  002000  001124          MOV     #2000,$GDDAT    ;EXPECT OFFSET TO BE SET INITIALLY
003132  012737  114000  032530          MOV     #114000,BUFFER  ;SET UP OFFSET INSTR
003140  012737  100000  032536          MOV     #100000,BUFFER+6        ;LOAD ''CHAR'' INST. (TO RESET INDICATOR)
003146  012777  032530  173604          MOV     #BUFFER,@DPC    ;START
003154  012737  010000  032532          MOV     #10000,BUFFER+2 ;X=OFFSET 0
003162  005277  176572                  INC     @DPC            ;PICK UP X
003166  012737  010000  032534          MOV     #10000,BUFFER+4 ;Y=OFFSET 0
003174  005277  176560                  INC     @DPC            ;PICK UP Y
003200  013737  002012  001122          MOV     STKPT,$BDADR    ;SET UP REG ADRS 32
003206  032777  002000  176576          BIT     #2000,@STKPT    ;IS THE OFFSET INDICATOR SET?
003214  001003                          BNE     1$              ;BR IF OK
003216  005037  001126                  CLR     $BDDAT          ;INDICATE IT WAS NOT SET
003222  104005                          ERROR   5               ;OFFSET INSTR INDICATOR FAILED TO SET
003224  005277  176530          1$:     INC     @DPC            ;PICK UP CHAR INSTR - SHOULD CLR OFFSET INDICATOR
003230  005037  001124                  CLR     $GDDAT          ;EXPECT 0
003234  032777  002000  176550          BIT     #2000,@STKPT    ;DID THE OFFSET INDICATOR CLR?
003242  001404                          BEQ     TST6            ;;GO TO NEXT TEST IF OFFSET INDICATOR RESET
003244  012737  002000  001126          MOV     #2000,$BDDAT    ;INDICATE IT WAS STILL SET
003252  104005                          ERROR   5               ;OFFSET INSTR INDICATOR FAILED TO CLR

                                        .SBTTL
                                        .SBTTL  THESE TEST RUN AT FULL SPEED
                                        .SBTTL

                                ;;*******************************************************************
                                ;*TEST 6         TEST THAT OFFSET INSTR CAN SET & RESET X POLARITY BIT AT REG 36
                                ;;*******************************************************************
```

```
(2)  003254  000004                    TST6:   SCOPE
(2)  003256  012737  000006  001202             MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
484  003264  012777  000040  176520             MOV     #40,@STKPT      ;RESET MAINT. BITS & 'TOP OF STACK'
485  003272  013737  002016  001122             MOV     ZDOFF,$BDADR    ;SET UP REG 36 ADRS
486  003300  012737  100000  001124             MOV     #100000,$GDDAT  ;EXPECT POLARITY BIT TO BE SET INITIALLY
487  003306  012737  114000  032530             MOV     #114000,BUFFER  ;SET UP OFFSET INSTR
488  003314  012737  030000  032532             MOV     #30000,BUFFER+2 ;SET UP LD OFFSET POLARITY BIT AT X DATA
489  003322  012737  010000  032534             MOV     #10000,BUFFER+4 ;Y OFFSET=0
490  003330  012737  172000  032536             MOV     #172000,BUFFER+6        ;LD STOP INSTR
491  003336  012737  161773  032540             MOV     #161773,BUFFER+10       ;JUMP TO REPEAT TEST
492  003344  004737  024524                     JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
493  003350  005777  176442                     TST     @ZDOFF          ;IS X POLARITY SET?
494  003354  100403                             BMI     1$              ;BR IF SO
495  003356  005037  001126                     CLR     $BDDAT          ;SHOW IT WAS NOT SET
496  003362  104006                             ERROR   6               ;OFFSET INSTR FAILED TO SET X POLARITY
497  003364  005037  001124             1$:     CLR     $GDDAT          ;PREPARE FOR RESET STATE
498  003370  042737  020000  032532             BIC     #20000,BUFFER+2 ;RESET POLARITY AT INSTR LOC
499  003376  004737  024552                     JSR     PC,CONT         ;RESUME
500  003402  005777  176410                     TST     @ZDOFF          ;IS X POLARITY RESET?
501  003406  100004                             BPL     TST7            ;;NEXT TEST IF SO
502  003410  012737  100000  001126             MOV     #100000,$BDDAT  ;SHOW IT WAS SET
503  003416  104006                             ERROR   6               ;OFFSET INSTR FAILED TO RESET X POLARITY
504
505                                     ;;**********************************************************************
(3)                                     ;*TEST 7         TEST THAT OFFSET INSTR CAN SET & RESET Y POLARITY BIT AT REG 36
(3)                                     ;;**********************************************************************
(2)  003420  000004                     TST7:   SCOPE
(2)  003422  012737  000007  001202             MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
506  003430  013737  002016  001122             MOV     ZDOFF,$BDADR    ;SET UP REG 36 ADRS
507  003436  012737  040000  001124             MOV     #40000,$GDDAT   ;EXPECT POLARITY BIT TO BE SET INITIALLY
508  003444  012737  114000  032530             MOV     #114000,BUFFER  ;SET UP OFFSET INSTR
509  003452  012737  010000  032532             MOV     #10000,BUFFER+2 ;SET UP X VALUE
510  003460  012737  172000  032536             MOV     #172000,BUFFER+6        ;LOAD "STOP" INST.
511  003466  012737  030000  032534             MOV     #30000,BUFFER+4 ;SET UP LD OFFSET & POLARITY AT Y DATA
512  003474  012737  161773  032540             MOV     #161773,BUFFER+10       ;JUMP TO REPEAT TEST
513  003502  004737  024524                     JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
514  003506  032777  040000  176302             BIT     #40000,@ZDOFF   ;IS Y POLARITY BIT SET?
515  003514  001003                             BNE     1$              ;BR IF SO
516  003516  005037  001126                     CLR     $BDDAT          ;SHOW IT WAS NOT SET
517  003522  104006                             ERROR   6               ;OFFSET INSTR FAILED TO SET Y POLARITY
518  003524  005037  001124             1$:     CLR     $GDDAT          ;ADVANCE TO RESET STATE
519  003530  042737  020000  032534             BIC     #20000,BUFFER+4 ;RESET POLARITY AT INSTR LOC
520  003536  004737  024552                     JSR     PC,CONT         ;RESUME
521  003542  032777  040000  176246             BIT     #40000,@ZDOFF   ;IS Y POLARITY RESET?
522  003550  001404                             BEQ     TST10           ;;NEXT TEST IF SO
523  003552  012737  040000  001126             MOV     #40000,$BDDAT   ;SHOW IT WAS SET
524  003560  104006                             ERROR   6               ;OFFSET INSTR FAILED TO RESET Y POLARITY
525
526                                     ;;**********************************************************************
(3)                                     ;*TEST 10        TEST THAT OFFSET INSTR CAN LOAD X & Y DYNAMIC OFFSET REG
(3)                                     ;;**********************************************************************
(2)  003562  000004                     TST10:  SCOPE
(2)  003564  012737  000010  001202             MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
527  003572  012737  003624  001110             MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
528  003600  012737  116000  032530             MOV     #116000,BUFFER  ;SET UP THE OFFSET INSTR
529  003606  012737  172000  032536             MOV     #172000,BUFFER+6        ;LOAD "STOP" INST.
```

```
530  003614  012700  004000            MOV     #4000,R0        ;FLOAT A ONE IN X POS
531  003620  012701  173777            MOV     #173777,R1      ;FLOAT A ZERO IN Y POS
532  003624  010037  032532    1$:     MOV     R0,BUFFER+2     ;SET UP X POS
533  003630  010137  032534            MOV     R1,BUFFER+4     ;SET UP Y POS
534  003634  042737  170000  032532    BIC     #170000,BUFFER+2        ;PLUS 12 BITS
535  003642  042737  170000  032534    BIC     #170000,BUFFER+4        ;PLUS 12 BITS
536  003650  052737  010000  032532    BIS     #BIT12,BUFFER+2 ;INDICATE AN OFFSET INSTR IN X
537  003656  004737  024524            JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
538  003662  013737  001774  001122    MOV     XDOFF,$BDADR    ;SET UP REG ADRS 14
539  003670  013737  032532  001124    MOV     BUFFER+2,$GDDAT ;LD EXPECTED
540  003676  042737  010000  001124    BIC     #BIT12,$GDDAT   ;DON'T WANT OFFSET BIT
541  003704  017737  176064  001126    MOV     @XDOFF,$BDDAT   ;READ X DYNAMIC OFFSET REG
542  003712  023737  001124  001126    CMP     $GDDAT,$BDDAT   ;CORRECT?
543  003720  001401                    BEQ     2$              ;BR IF SO
544  003722  104007                    ERROR   7               ;OFFSER INSTR FAILED TO SET UP X DYNAMIC OFFSET REG
545  003724  013737  001776  001122  2$:     MOV     YDOFF,$BDADR    ;SET UP REG ADRS 16
546  003732  013737  032534  001124        MOV     BUFFER+4,$GDDAT ;LD EXPECTED
547  003740  017737  176032  001126        MOV     @YDOFF,$BDDAT   ;READ Y DYNAMIC OFFSET REG
548  003746  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;CORRECT?
549  003754  001401                        BEQ     3$              ;BR IF SO
550  003756  104007                        ERROR   7               ;OFFSET INSTR FAILED TO SET UP Y DYNAMIC OFFSET REG
551  003760  006200                3$:     ASR     R0              ;FLOAT PAT RIGHT
552  003762  001403                        BEQ     4$              ;BR IF X DONE WITH FLOAT ONE
553  003764  006201                        ASR     R1              ;FLOAT PAT RIGHT
554  003766  001406                        BEQ     TST11           ;;NEXT TEST IF ONES AND ZEROS TESTED
555  003770  000715                        BR      1$              ;REPEAT TEST THIS PATTERN
556  003772  012700  173777        4$:     MOV     #173777,R0      ;NOW FLOAT A ZERO IN X
557  003776  012701  004000                MOV     #4000,R1        ;NOW FLOAT A ONE IN Y
558  004002  000710                        BR      1$              ;REPEAT TEST
559
560                                ;;*****************************************************************
(3)                                ;*TEST 11       TEST THAT X & Y POSITION REGS CAN BE LOADED CORRECTLY USING POINT MODE 1
(3)                                ;;*****************************************************************
(2)  004004  000004                TST11:  SCOPE
(2)  004006  012737  000011  001202        MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
561  004014  005077  175754                CLR     @XDOFF          ;CLR REG 14
562  004020  005077  175752                CLR     @YDOFF          ;CLR REG 16
563  004024  012737  004056  001110        MOV     #1$,$LPERR      ;SET UP SCOPE LOOP
564  004032  012737  116000  032530        MOV     #116000,BUFFER  ;LD POINT INSTR
565  004040  012737  172000  032536        MOV     #172000,BUFFER+6  ;LD STOP INSTR
566  004046  012700  004000                MOV     #4000,R0        ;FLOAT A ONE IN X POS
567  004052  012701  173777                MOV     #173777,R1      ;FLOAT A ZERO IN Y POS
568  004056  010037  032532        1$:     MOV     R0,BUFFER+2     ;SET UP X POS
569  004062  010137  032534                MOV     R1,BUFFER+4     ;SET UP Y POS
570  004066  042737  170000  032532        BIC     #170000,BUFFER+2  ;12 BITS
571  004074  042737  170000  032534        BIC     #170000,BUFFER+4  ;12 BITS
572  004102  004737  024524                JSR     PC,EXECUTE      ;START
573  004106  013737  001764  001122        MOV     XPOS,$BDADR     ;SET UP REG 04 ADRS
574  004114  013737  032532  001124        MOV     BUFFER+2,$GDDAT ;LD EXPECTED
575  004122  017737  175636  001126        MOV     @XPOS,$BDDAT    ;READ X POS
576  004130  017705  175640                MOV     @XDOFF,R5       ;GET HI ORDER X POS
577  004134  004737  025136                JSR     PC,POCONV       ;GO MAKE LP 14 BIT X POS
578  004140  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;CORRECT?
579  004146  001401                        BEQ     2$              ;BR IF SO
580  004150  104014                        ERROR   14              ;POINT INSTR FAILED TO LOAD X POS CORRECTLY
581  004152  013737  001766  001122  2$:     MOV     YPOS,$BDADR     ;SET UP REG 06 ADRS
```

```
                                MOV     BUFFER+4,$GDDAT ;LD  EXPECTED
                                MOV     @YPOS,$BDDAT    ;READ Y POS
                                MOV     @YDOFF,R5       ;GET HI ORDER Y POS
                                JSR     PC,POCONV       ;GO MAKE UP 14 BIT Y POS
                                CMP     $GDDAT,$BDDAT   ;CORRECT?
                                BEQ     3$              ;BR IF SO
                                ERROR   15              ;POINT INSTR FAILED TO LOAD Y POS CORRECTLY
                        3$:     ASR     R0              ;FLOAT PAT RIGHT
                                BEQ     4$              ;BR IF X DONE WITH FLOAT 1
                                ASR     R1              ;FLOAT PAT RIGTH
                                BEQ     TST12           ;;NEXT TEST IF ONES + ZEROS FLOATED
                                BR      1$              ;REPEAT TEST THIS PAT
                        4$:     MOV     #173777,R0      ;NOW FLOAT ZERO IN X
                                MOV     #4000,R1        ;NOW FLOAT ONE IN Y
                                BR      1$              ;REPEAT TEST

                        ;;*******************************************************************
                        ;*TEST 12       TEST THAT X & Y POSITION REGS CAN BE LOADED CORRECTLY USING ABS VECTOR
                        ;;*******************************************************************
                        TST12:  SCOPE
                                MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                CLR     @XDOFF          ;CLR REG 14
                                CLR     @YDOFF          ;CLR REG 16
                                MOV     #1$,$LPERR      ;SET UP SCOPE LOOP
                                MOV     #BUFFER,R0      ;SET INSTR ADRS
                                MOV     #116000,(R0)+   ;POINT INSTR
                                CLR     (R0)+           ;X = 0
                                CLR     (R0)+           ;Y = 0
                                MOV     #146000,(R0)+   ;ABSOLUTE VECTOR INSTR
                                CMP     (R0)+,(R0)+     ;SKIP OVER VECTORS PTS
                                MOV     #172000,(R0)    ;SET UP HALT
                                MOV     #4000,R0        ;FLOAT A ONE IN X VECTOR POINT
                                MOV     #173777,R1      ;FLOAT A ZERO IN Y VECTOR POINT
                        1$:     MOV     R0,BUFFER+10    ;SET UP X POINT
                                MOV     R1,BUFFER+12    ;SET UP Y POINT
                                BIC     #170000,BUFFER+10  ;12 BITS
                                BIC     #170000,BUFFER+12  ;12 BITS
                                JSR     PC,EXECUTE      ;START
                                MOV     XPOS,$BDADR     ;SET UP REG 04 ADRS
                                MOV     BUFFER+10,$GDDAT  ;LD EXPECTED
                                MOV     @XPOS,$BDDAT    ;READ X POS
                                MOV     @XDOFF,R5       ;GET HI ORDER X POS
                                JSR     PC,POCONV       ;GO MAKE UP 14 BIT X POS
                                CMP     $GDDAT,$BDDAT   ;CORRECT?
                                BEQ     2$              ;BR IF SO
                                ERROR   12              ;ABSOLUTE VECTOR INSTR FAILED TO LOAD X POS
                        2$:     MOV     YPOS,$BDADR     ;SET UP REG 06 ADRS
                                MOV     BUFFER+12,$GDDAT        ;LD EXPECTED
                                MOV     @YPOS,$BDDAT    ;READ Y POS
                                MOV     @YDOFF,R5       ;GET HI ORDER Y POS
                                JSR     PC,POCONV       ;GO MAKE UP 14 BIT X POS
                                CMP     $GDDAT,$BDDAT   ;CORRECT?
                                BEQ     3$              ;BR IF SO
                                ERROR   12              ;ABSOLUTE VECTOR INSTR FAILED TO LOAD Y POS
                        3$:     ASR     R0              ;FLOAT PAT RIGHT
                                BEQ     4$              ;BR IF X DONE WITH FLOAT 1
```

```
   634  004472  006201                          ASR     R1              ;FLOAT PAT RIGHT
   635  004474  001406                          BEQ     TST13           ;;NEXT TEST IF ONES + ZEROES FLOATED
   636  004476  000713                          BR      1$              ;REPEAT TEST THIS PAT
   637  004500  012700  173777          4$:     MOV     #173777,R0      ;NOW FLOAT ZERO IN X
   638  004504  012701  004000                  MOV     #4000,R1        ;NOW FLOAT ONE IN Y
   639  004510  000706                          BR      1$              ;REPEAT TEST
   640
   641                                  ;;*****************************************************************
   (3)                                  ;*TEST 13       TEST THAT X POSITION BITS 10 & 11 WILL SET & RESET VIA OFFSET INSTR
   (3)                                  ;;*****************************************************************
   (2)  004512  000004          TST13:  SCOPE
   (2)  004514  012737  000013  001202          MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
   642  004522  012737  004600  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
   643  004530  013737  001774  001122          MOV     XDOFF,$BDADR    ;SET UP REG 14 ADRS
   644  004536  012737  036000  001124          MOV     #36000,$GDDAT   ;EXPECT BOTH BITS INITIALLY
   645  004544  012700  032530                  MOV     #BUFFER,R0      ;GET INSTR ADRS POINTER
   646  004550  012720  114000                  MOV     #114000,(R0)+   ;LOAD OFFSET INSTR
   647  004554  012720  016000                  MOV     #16000,(R0)+    ;SET UP MAX X OFFSET
   648  004560  012720  010000                  MOV     #10000,(R0)+    ;SET UP Y OFFSET
   649  004564  012720  114000                  MOV     #114000,(R0)+   ;LOAD POINT INSTR
   650  004570  005020                          CLR     (R0)+           ;X OF 0
   651  004572  005020                          CLR     (R0)+           ;Y OF 0
   652  004574  012710  172000                  MOV     #172000,(R0)    ;LOAD STOP INSTR
   653  004600  005077  175170          1$:     CLR     @XDOFF          ;CLR OFFSET REG
   654  004604  004737  024524                  JSR     PC,EXECUTE      ;GO START DISPLAY
   655  004610  017737  175160  001126          MOV     @XDOFF,$BDDAT   ;READ REG 14
   656  004616  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY CORRECT?
   657  004624  001401                          BEQ     2$              ;BR IF SO
   658  004626  104010                          ERROR   10              ;OFFSET INSTR FAILED TO LOAD HIGH ORDER X POSITION BIT
   659  004630  162737  012000  001124  2$:     SUB     #12000,$GDDAT   ;SET UP FOR NEXT ROUTINE
   660  004636  162737  002000  032532          SUB     #2000,BUFFER+2  ;SET UP NEXT VALUE
   661  004644  022737  006000  032532          CMP     #6000,BUFFER+2  ;ALL VALUES TESTED?
   662  004652  001352                          BNE     1$              ;BR IF NOT
   663
   664                                  ;;*****************************************************************
   (3)                                  ;*TEST 14       TEST THAT Y POSITION BITS 10 & 11 WILL SET & RESET VIA OFFSET INSTR
   (3)                                  ;;*****************************************************************
   (2)  004654  000004          TST14:  SCOPE
   (2)  004656  012737  000014  001202          MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
   665  004664  012737  004742  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
   666  004672  013737  001776  001122          MOV     YDOFF,$BDADR    ;SET UP REG 16 ADRS
   667  004700  012737  036000  001124          MOV     #36000,$GDDAT   ;EXPECT BOTH BITS INITIALLY
   668  004706  012700  032530                  MOV     #BUFFER,R0      ;GET INSTR ADRS POINTER
   669  004712  012720  114000                  MOV     #114000,(R0)+   ;LOAD OFFSET INSTR
   670  004716  012720  010000                  MOV     #10000,(R0)+    ;SET UP X OFFSET
   671  004722  012720  016000                  MOV     #16000,(R0)+    ;SET UP MAX Y OFFSET
   672  004726  012720  114000                  MOV     #114000,(R0)+   ;LOAD POINT INSTR
   673  004732  005020                          CLR     (R0)+           ;X OF 0
   674  004734  005020                          CLR     (R0)+           ;Y OF 0
   675  004736  012710  172000                  MOV     #172000,(R0)    ;LOAD STOP INSTR
   676  004742  005077  175030          1$:     CLR     @YDOFF          ;CLR OFFSET REG
   677  004746  004737  024524                  JSR     PC,EXECUTE      ;GO START DISPLAY
   678  004752  017737  175020  001126          MOV     @YDOFF,$BDDAT   ;READ REG 16
   679  004760  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY CORRECT?
   680  004766  001401                          BEQ     2$              ;BR IF SO
   681  004770  104010                          ERROR   10              ;OFFSET INSTR FAILED TO LOAD HIGH ORDER Y POSITION BITS
```

```
   682  004772  162737  012000  001124  2$:     SUB     #12000,$GDDAT   ;SET UP FOR NEXT VALUE
   683  005000  162737  002000  032532          SUB     #2000,BUFFER+2  ;SET UP NEXT VALUE
   684  005006  022737  006000  032532          CMP     #6000,BUFFER+2  ;ALL VALUES TESTED?
   685  005014  001352                          BNE     1$              ;BR IF NOT
   686
   687                                  ;;*****************************************************************
   (3)                                  ;*TEST 15       TEST X AND Y POS BITS 10-13 (+ DIR) USING THE POINT & OFFSET INSTRS
   (3)                                  ;;*****************************************************************
   (2)  005016  000004                  TST15:  SCOPE
   (2)  005020  012737  000015  001202          MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
   688  005026  012700  032530                  MOV     #BUFFER,R0      ;GET ADRS WHERE INSTR'S GO
   689  005032  012720  116000                  MOV     #116000,(R0)+   ;OFFSET INSTR
   690  005036  012720  011000                  MOV     #11000,(R0)+    ;X OFFSET OF 1000
   691  005042  012720  017000                  MOV     #17000,(R0)+    ;Y OFFSET OF 7000
   692  005046  012720  116000                  MOV     #116000,(R0)+   ;POINT INSTR
   693  005052  012720  001000                  MOV     #1000,(R0)+     ;X=1000
   694  005056  012720  007000                  MOV     #7000,(R0)+     ;Y=7000
   695  005062  012720  172000                  MOV     #172000,(R0)+   ;STOP INSTR
   696  005066  012700  010000                  MOV     #10000,R0       ;LD EXPECTED X POS
   697  005072  012701  070000                  MOV     #70000,R1       ;LD EXPECTED Y POS
   698  005076  012737  005104  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
   699  005104                       1$:
   (1)  005104  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
   700  005110  013737  001774  001122          MOV     XDOFF,$BDADR    ;SET UP REG ADRS 14
   701  005116  010037  001124                  MOV     R0,$GDDAT       ;LD EXPECTED
   702  005122  017737  174646  001126          MOV     @XDOFF,$BDDAT   ;READ HI X POS BITS
   703  005130  042737  007777  001126          BIC     #7777,$BDDAT    ;GET RID OF OFFSET
   704  005136  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
   705  005144  001401                          BEQ     2$              ;BR IF SO
   706  005146  104010                          ERROR   10              ;X POS BITS 10-13 FAILED ON POINT/OFFSET
   707  005150  013737  001776  001122  2$:     MOV     YDOFF,$BDADR    ;SET UP REG ADRS 16
   708  005156  010137  001124                  MOV     R1,$GDDAT       ;LD EXPECTED
   709  005162  017737  174610  001126          MOV     @YDOFF,$BDDAT   ;READ HI Y POS BITS
   710  005170  042737  007777  001126          BIC     #7777,$BDDAT    ;GET RID OF OFFSET
   711  005176  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
   712  005204  001401                          BEQ     3$              ;BR IF SO
   713  005206  104010                          ERROR   10              ;Y POS 10-13 FAILED ON POINT/OFFSET
   714  005210  062737  001000  032532  3$:     ADD     #1000,BUFFER+2  ;INCREASE X POINT
   715  005216  062737  001000  032540          ADD     #1000,BUFFER+10 ;INCREASE X OFFSET
   716  005224  162737  001000  032534          SUB     #1000,BUFFER+4  ;DECREASE Y POINT
   717  005232  162737  001000  032542          SUB     #1000,BUFFER+12 ;DECREASE Y OFFSET
   718  005240  062700  010000                  ADD     #10000,R0       ;UPDATE EXPECTED X
   719  005244  162701  010000                  SUB     #10000,R1       ;UPDATE EXPECTED Y
   720  005250  001315                          BNE     1$              ;BR IF NOT DONE WITH 7 CALCULATIONS
   721
   722                                  ;;*****************************************************************
   (3)                                  ;*TEST 16       TEST ALL POLARITY COMBINATIONS FOR X & Y DELTA AND X & Y OFFSET
   (3)                                  ;;*****************************************************************
   (2)  005252  000004                  TST16:  SCOPE
   (2)  005254  012737  000016  001202          MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
   723                                          ;USE POINT INSTR FIRST THEN REPLACE POINT
   724                                          ;INSTR WITH ABSOLUTE VECTOR INSTR.
   725  005262  012700  032530                  MOV     #BUFFER,R0      ;SET UP ADRS WHERE INSTRS GO
   726  005266  012720  116000                  MOV     #116000,(R0)+   ;OFFSET INSTR
   727  005272  012720  010001                  MOV     #10001,(R0)+    ;X OFFSET OF 1
   728  005276  012720  010003                  MOV     #10003,(R0)+    ;Y OFFSET OF 3
```

```
                                                MOV     #116000,(R0)+   ;POINT INSTR
                                                MOV     #2,(R0)+        ;X=2
                                                MOV     #4,(R0)+        ;Y=4
                                                MOV     #172000,(R0)+   ;STOP INSTR
                                                MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
                                                CLR     R3              ;R3 WILL CONTROL POLARITY OF ALL X & Y DATA
                                        1$:
(1)                                             JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
                                                MOV     XPOS,$BDADR     ;REPORT ER'S AT REG 02
                                                MOV     BUFFER+10,R0    ;GET X POINT
                                                MOV     BUFFER+2,R1     ;GET X OFFSET
                                                JSR     PC,CALPT        ;GO COMPUTE 14 BIT POS
                                                MOV     @XPOS,$BDDAT    ;READ X POS
                                                MOV     @XDOFF,R5       ;READ X OFFSET
                                                JSR     PC,POCONV       ;GO MAKE UP 14 BIT POS
                                                CMP     $GDDAT,$BDDAT   ;CORRECT?
                                                BEQ     2$              ;BR IF SO
                                                ERROR   42              ;POINT FAILED AT DELTA X & OFFSET X IN BUFFER+2 & +10
                                        2$:     MOV     YPOS,$BDADR     ;REPORT ER'S AT REG 04
                                                MOV     BUFFER+12,R0    ;GET Y POINT
                                                MOV     BUFFER+4,R1     ;GET Y OFFSET
                                                JSR     PC,CALPT        ;GO COMPUTE 14 BIT POS
                                                MOV     @YPOS,$BDDAT    ;READ Y POS
                                                MOV     @YDOFF,R5       ;READ Y HI POS
                                                JSR     PC,POCONV       ;GO MAKE UP 14 BIT POS
                                                CMP     $GDDAT,$BDDAT   ;CORRECT?
                                                BEQ     3$              ;BR IF SO
                                                ERROR   42              ;POINT FAILED AT DELTA Y & OFFSET Y IN BUFFER+4 & +12
                                        3$:     INC     R3              ;BUMP POLARITY CONTROL
                                                CMP     #20,R3          ;DONE 16 POSIBILITIES?
                                                BEQ     9$      ;SEE IF ABS VECTOR WAS TESTED
                                        4$:     BIC     #BIT13,BUFFER+2 ;RESET X POL OFFSET
                                                BIT     #BIT0,R3        ;WANT -POL X OFFSET?
                                                BEQ     5$              ;BR IF NOT
                                                BIS     #BIT13,BUFFER+2 ;SET -POL X OFFSET
                                        5$:     BIC     #BIT13,BUFFER+4 ;RESET Y POL OFFSET
                                                BIT     #BIT1,R3        ;WANT -POL Y OFFSET?
                                                BEQ     6$              ;BR IF NOT
                                                BIS     #BIT13,BUFFER+4 ;SET -POL Y OFFSET
                                        6$:     BIC     #BIT13,BUFFER+10        ;RESET X POL
                                                BIT     #BIT2,R3        ;WANT -POL X?
                                                BEQ     7$              ;BR IF NOT
                                                BIS     #BIT13,BUFFER+10        ;SET -POL X
                                        7$:     BIC     #BIT13,BUFFER+12        ;RESET POL Y
                                                BIT     #BIT3,R3        ;WANT -POL Y?
                                                BEQ     8$              ;BR IF NOT
                                                BIS     #BIT13,BUFFER+12        ;SET -POL X
                                        8$:     BR      1$              ;REPEAT TEST
                                        9$:     CMP     #146000,BUFFER+6        ;JUST DO ABS VEC?
                                                BEQ     TST17           ;;NEXT TEST IF SO
                                                MOV     #146000,BUFFER+6        ;NOW SET UP ASB VEC INSTR
                                                CLR     R3              ;RESET POLARITY CONTROL
                                                BR      4$              ;REPEAT TEST WITH ABS VEC

                                        ;;*****************************************************************
(3)                                     ;*TEST 17       TEST THAT RESET WILL CLEAR X & Y POS BITS 10-13
```

```
(3)
(2)  005626  000004                        ;;*****************************************************************
(1)  005630  012737  000040  001166  TST17: SCOPE
(2)  005636  012737  000017  001202        MOV     #40,$TIMES      ;;DO 40 ITERATIONS
783  005644  012700  032530                MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
784  005650  012720  116000                MOV     #BUFFER,R0      ;SET UP ADRS FOR INSTRS
785  005654  012720  010000                MOV     #116000,(R0)+   ;SET UP OFFSET INSTR
786  005660  012720  010000                MOV     #10000,(R0)+    ;X OFFSET
787  005664  012720  116000                MOV     #10000,(R0)+    ;Y OFFSET
788  005670  012720  020001                MOV     #116000,(R0)+   ;SET UP POINT INSTR
789  005674  012720  020001                MOV     #20001,(R0)+    ;X POINT
790  005700  012710  172000                MOV     #20001,(R0)+    ;Y POINT
791  005704  005037  001124                MOV     #172000,(R0)    ;SET UP HALT INSTR
792  005710  004737  024524                CLR     $GDDAT          ;EXPECT ZERO
793  005714  000005                        JSR     PC,EXECUTE      ;START
794  005716  017737  174052  001126        RESET                   ;CLR X + Y POS
795  005724  001404                        MOV     @XDOFF,$BDDAT   ;DID IT CLR?
796  005726  013737  001774  001122        BEQ     1$              ;BR IF SO
797  005734  104032                        MOV     XDOFF,$BDADR    ;SET UP REG ADRS 14
798  005736  017737  174034  001126  1$:   ERROR   32              ;HI ORDER X POS BITS FAILED TO CLEAR
799  005744  001404                        MOV     @YDOFF,$BDDAT   ;DID IT CLR?
800  005746  013737  001776  001122        BEQ     TST20           ;;NEXT TEST IF SO
801  005754  104032                        MOV     YDOFF,$BDADR    ;SET UP REG ADRS 16
802                                        ERROR   32              ;HI ORDER Y POS BITS FAILED TO CLEAR
803                                     ;;*****************************************************************
(3)                                     ;*TEST 20       TEST THAT LONG VECTOR MODE INCREMENT X AND Y AXIS COUNT 1-1777
(3)                                     ;;*****************************************************************
(2)  005756  000004                  TST20: SCOPE
(1)  005760  012737  000100  001166        MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(2)  005766  012737  000020  001202        MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
804  005774  012737  006050  001110        MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
805  006002  012703  001777                MOV     #1777,R3        ;SET UP A COUNTER
806  006006  012737  000001  001124        MOV     #1,$GDDAT       ;PRESET THE COMPARED VALUE
807  006014  013700  001712          1$:   MOV     DBUF,R0         ;SET UP R0
808  006020  012720  116000                MOV     #116000,(0)+    ;LOAD 'POINT MODE''
809  006024  005020                        CLR     (0)+            ;CLEAR X AXIS
810  006026  005020                        CLR     (0)+            ;CLEAR Y AXIS
811  006030  012720  110000                MOV     #110000,(0)+    ;LOAD 'LONG VECTOR MODE''
812  006034  013720  001124                MOV     $GDDAT,(0)+     ;PRESET 'DELTA X AXIS''
813  006040  013720  001124                MOV     $GDDAT,(0)+     ;PRESET 'DELTA Y AXIS''
814  006044  012720  172000                MOV     #172000,(R0)+   ;LOAD 'DISPLAY STOP''
815  006050                           2$:
(1)  006050  004737  024524                JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
816
817  006054  017737  173704  001126        MOV     @XPOS,$BDDAT    ;READ X POS
818  006062  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;ARE THEY EQUAL ?
819  006070  001404                        BEQ     3$              ;;YES
820  006072  013737  001764  001122        MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
821  006100  104016                        ERROR   16              ;NO, LONG VECTOR MODE FAILED
822
823  006102  017737  173660  001126  3$:   MOV     @YPOS,$BDDAT    ;READ Y POS
824  006110  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;COMPARE
825  006116  001404                        BEQ     4$              ;YES
826  006120  013737  001766  001122        MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
827  006126  104016                        ERROR   16              ;NO, INCREMENT Y AXIS VIA
828
```

```
  829  006130  005237  001124          4$:     INC     $GDDAT          ;INCREMENT EXPECTED VALUE
  830  006134  005303                          DEC     R3              ;FINISHED?
  831  006136  001326                          BNE     1$              ;;NO, TEST MORE DATA
  832
  833                                  ;;*****************************************************************
  (3)                                  ;*TEST 21       TEST THAT LONG VECTOR MODE DECREMENTS X AND Y AXIS COUNT 1-1777
  (3)                                  ;;*****************************************************************
  (2)  006140  000004                  TST21:  SCOPE
  (1)  006142  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
  (2)  006150  012737  000021  001202          MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  834  006156  012737  006232  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
  835  006164  012703  002000                  MOV     #2000,R3        ;SET UP A COUNTER
  836  006170  012737  001777  001124          MOV     #1777,$GDDAT    ;PRESET THE COMPARED VALUE
  837  006176  012705  020001                  MOV     #20001,R5
  838  006202  013700  001712          1$:     MOV     DBUF,R0         ;SET UP R0
  839  006206  012720  116000                  MOV     #116000,(0)+    ;LOAD 'POINT MODE'
  840  006212  005020                          CLR     (0)+            ;CLEAR X AXIS
  841  006214  005020                          CLR     (0)+            ;CLEAR Y AXIS
  842  006216  012720  110000                  MOV     #110000,(0)+    ;LOAD 'LONG VECTOR MODE'
  843  006222  010520                          MOV     R5,(0)+         ;PRESET 'DELTA X AXIS'
  844  006224  010520                          MOV     R5,(0)+         ;PRESET 'DELTA Y AXIS'
  845  006226  012720  172000                  MOV     #172000,(R0)+   ;PRESET 'DISPLAY STOP'
  846  006232                          2$:
  (1)  006232  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
  847
  848  006236  017737  173522  001126          MOV     @XPOS,$BDDAT    ;READ X AXIS
  849  006244  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY EQUAL?
  850  006252  001404                          BEQ     3$              ;YES
  851  006254  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
  852  006262  104017                          ERROR   17              ;NO, DECREMENT X AXIS VIA
  853
  854  006264  017737  173476  001126  3$:     MOV     @YPOS,$BDDAT    ;READ Y AXIS
  855  006272  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY EQUAL?
  856  006300  001404                          BEQ     4$              ;YES
  857  006302  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
  858  006310  104017                          ERROR   17              ;NO, DECREMENT Y AXIS VIA
  859
  860  006312  005205                  4$:     INC     R5              ;INCREMENT 'DELTA X-Y'
  861  006314  005337  001124                  DEC     $GDDAT          ;DECREMENT EXPECTED VALUE
  862  006320  005303                          DEC     R3              ;FINISHED?
  863  006322  001327                          BNE     1$              ;;NO, TEST MORE DATA
  864
  865                                  ;;*****************************************************************
  (3)                                  ;*TEST 22       TEST LONG VECTOR FOR ALL VALUES OF SCALE
  (3)                                  ;;*****************************************************************
  (2)  006324  000004                  TST22:  SCOPE
  (2)  006326  012737  000022  001202          MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  866  006334  012700  032530                  MOV     #BUFFER,R0      ;GET 1ST ADRS WHERE INTRS' GO
  867  006340  012720  154024                  MOV     #154024,(R0)+   ;SCALE OF UNITY
  868  006344  012720  116000                  MOV     #116000,(R0)+   ;POINT INSTR
  869  006350  005020                          CLR     (R0)+           ;X = 0
  870  006352  005020                          CLR     (R0)+           ;Y = 0
  871  006354  012720  154021                  MOV     #154021,(R0)+   ;SCALE TO BE UPDATED
  872  006360  012720  112000                  MOV     #112000,(R0)+   ;LONG VECTOR INSTR
  873  006364  012720  000100                  MOV     #100,(R0)+      ;X = 100
  874  006370  012720  000200                  MOV     #200,(R0)+      ;Y = 200
```

```
 875  006374  012710  172000                        MOV     #172000,(R0)    ;STOP INSTR
 876  006400  012737  000001  001164                MOV     #1,$TMP0        ;SET UP SCALE FOR ER TYPE
 877  006406  012701  000020                        MOV     #20,R1          ;SET UP X EXPECTED DATA
 878  006412  012702  000040                        MOV     #40,R2          ;SET UP Y EXPECTED DATA
 879  006416  012737  006424  001110                MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
 880  006424  004737  024524              1$:       JSR     PC,EXECUTE      ;START DISPLAY
 881  006430  013737  001764  001122                MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
 882  006436  010137  001124                        MOV     R1,$GDDAT       ;LD EXPECTED
 883  006442  012737  000100  001162                MOV     #100,$REG0      ;SET UP X VECTOR FOR ER TYPE
 884  006450  017737  173310  001126                MOV     @XPOS,$BDDAT    ;READ X POS REG
 885  006456  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;CORRECT?
 886  006464  001401                                BEQ     2$              ;BR IF SO
 887  006466  104011                                ERROR   11              ;X VECTOR POS FAILED TO SCALE
 888  006470  013737  001766  001122        2$:     MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 889  006476  010237  001124                        MOV     R2,$GDDAT       ;LD EXPECTED
 890  006502  012737  000200  001162                MOV     #200,$REG0      ;SET UP Y VECTOR FOR ER TYPE
 891  006510  017737  173252  001126                MOV     @YPOS,$BDDAT    ;READ Y POS REG
 892  006516  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;CORRECT?
 893  006524  001401                                BEQ     3$              ;BR IF SO
 894  006526  104011                                ERROR   11              ;Y VECTOR POS FAILED TO SCALE
 895  006530  005237  001164              3$:       INC     $TMP0           ;ADVANCE SCALE BY 1/4
 896  006534  022737  000020  001164                CMP     #20,$TMP0       ;ALL SCALES TESTED?
 897  006542  001407                                BEQ     TST23           ;;NEXT TEST IF SO
 898  006544  005237  032540                        INC     BUFFER+10       ;BUMP SCALE BY 1/4 AT INSTR LOC
 899  006550  062701  000020                        ADD     #20,R1          ;BUMP EXPECTED X POS BY 1/4
 900  006554  062702  000040                        ADD     #40,R2          ;BUMP EXPECTED Y POS BY 1/4
 901  006560  000721                                BR      1$              ;TRY NEXT SCALE
 902
 903                                        ;;**********************************************************************
 (3)                                        ;*TEST 23        TEST THAT X AND Y AXIS INCREMENT PROPERLY USING SHORT VECTOR MODE COUNT
 (3)                                        ;;**********************************************************************
 (2)  006562  000004                        TST23:  SCOPE
 (1)  006564  012737  000200  001166                MOV     #200,$TIMES     ;;DO 200 ITERATIONS
 (2)  006572  012737  000023  001202                MOV     #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 904  006600  012737  006656  001110                MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
 905  006606  012703  000077                        MOV     #77,R3          ;SET UP A COUNT LOCATION
 906  006612  012737  000001  001124                MOV     #1,$GDDAT       ;SET UP THE COMPARED LOCATION
 907  006620  012704  000201                        MOV     #201,R4         ;SET UP 'DELTA X-Y'
 908  006624  013700  001712              1$:       MOV     DBUF,R0         ;SET UP R0
 909  006630  012720  154024                        MOV     #154024,(0)+    ;SCALE OF UNITY
 910  006634  012720  116000                        MOV     #116000,(0)+    ;LOAD 'SET POINT DATA MODE'
 911  006640  005020                                CLR     (0)+            ;CLEAR X AXIS
 912  006642  005020                                CLR     (0)+            ;CLEAR Y AXIS
 913  006644  012720  106000                        MOV     #106000,(0)+    ;LOAD 'SET SHORT VECTOR MODE'
 914  006650  010420                                MOV     R4,(0)+         ;PRESET 'DELTA X AND DELTA Y'
 915  006652  012720  172000                        MOV     #172000,(R0)+   ;LOAD DISPLAY STOP
 916  006656                              2$:
 (1)  006656  004737  024524                        JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 917
 918  006662  017737  173076  001126                MOV     @XPOS,$BDDAT    ;READ X POSITION
 919  006670  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;ARE THEY EQUAL
 920  006676  001404                                BEQ     3$              ;YES
 921  006700  013737  001764  001122                MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
 922  006706  104016                                ERROR   16              ;INCREMENT X AXIS FAILED USING
 923
 924  006710  013737  001766  001122  3$:           MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
```

```
  925  006716  017737  173044  001126          MOV     @YPOS,$BDDAT    ;READ Y POSITION
  926  006724  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY EQUAL ?
  927  006732  001401                          BEQ     4$              ;YES
  928  006734  104016                          ERROR   16              ;INCREMENT Y AXIS FAILED USING
  929
  930  006736  062704  000201          4$:     ADD     #201,R4         ;ADD DELTA X-Y
  931  006742  005237  001124                  INC     $GDDAT          ;INCREMENT EXPECTED VALUE
  932  006746  005303                          DEC     R3              ;DECREMENT COUNT, FINISHED?
  933  006750  001325                          BNE     1$              ;;NO, TEST MORE DATA
  934
  935                                  ;;*****************************************************************
  (3)                                  ;*TEST 24       TEST THAT X AND Y AXIS DECREMENT PROPERLY USING SHORT VECTOR MODE COUNT
  (3)                                  ;;*****************************************************************
  (2)  006752  000004                  TST24:  SCOPE
  (1)  006754  012737  000200  001166          MOV     #200,$TIMES     ;;DO 200 ITERATIONS
  (2)  006762  012737  000024  001202          MOV     #24,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  936  006770  012737  007042  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
  937  006776  012703  000077                  MOV     #77,R3          ;SET UP A COUNT LOCATION
  938  007002  012737  001777  001124          MOV     #1777,$GDDAT    ;SET UP THE COMPARED LOCATION
  939  007010  012704  020301                  MOV     #20301,R4       ;PRESET THE 'DELTA X-Y'
  940  007014  013700  001712          1$:     MOV     DBUF,R0         ;SET UP R0
  941  007020  012720  116000                  MOV     #116000,(0)+    ;LOAD 'SET POINT DATA MODE'
  942  007024  005020                          CLR     (0)+            ;CLEAR X AXIS
  943  007026  005020                          CLR     (0)+            ;CLEAR Y AXIS
  944  007030  012720  106000                  MOV     #106000,(0)+    ;LOAD 'SET SHORT VECTOR MODE'
  945  007034  010420                          MOV     R4,(0)+         ;PRESET 'DELTA X AND DELTA Y'
  946  007036  012720  172000                  MOV     #172000,(R0)+   ;LOAD DISPLAY STOP
  947  007042                          2$:
  (1)  007042  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
  948
  949  007046  017737  172712  001126          MOV     @XPOS,$BDDAT    ;READ X POSITION
  950  007054  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY EQUAL
  951  007062  001404                          BEQ     3$              ;YES
  952  007064  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
  953  007072  104017                          ERROR   17              ;DECREMENT X AXIS FAILED USING
  954
  955  007074  017737  172666  001126  3$:     MOV     @YPOS,$BDDAT    ;READ Y POSITION
  956  007102  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY EQUAL ?
  957  007110  001404                          BEQ     4$              ;YES DECREMENT
  958  007112  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
  959  007120  104017                          ERROR   17              ;DECREMENT Y AXIS FAILED USING
  960
  961  007122  062704  000201          4$:     ADD     #201,R4         ;ADD 'DELTA X-Y'
  962  007126  005337  001124                  DEC     $GDDAT          ;DECREMENT EXPECTED VALUE
  963  007132  005303                          DEC     R3              ;DECREMENT COUNT, FINISHED?
  964  007134  001327                          BNE     1$              ;;NO, TEST MORE DATA
  965
  966                                  ;;*****************************************************************
  (3)                                  ;*TEST 25       TEST SHORT VECTOR AT ORIGIN 1000/1000 FOR ALL VALUES OF SCALE
  (3)                                  ;;*****************************************************************
  (2)  007136  000004                  TST25:  SCOPE
  (2)  007140  012737  000025  001202          MOV     #25,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  967  007146  012700  032530                  MOV     #BUFFER,R0      ;GET 1ST ADRS WHERE INSTR'S GO
  968  007152  012720  154024                  MOV     #154024,(R0)+   ;'LD STATUS C' (SCALE OF UNITY)
  969  007156  012720  116000                  MOV     #116000,(R0)+   ;POINT INSTR
  970  007162  012720  001000                  MOV     #1000,(R0)+     ;X = 1000
```

```
  971  007166  012720  001000                    MOV     #1000,(R0)+     ;Y = 1000
  972  007172  012720  154021                    MOV     #154021,(R0)+   ;'LD STATUS C'' (SCALE TO BE UPDATED)
  973  007176  012720  106000                    MOV     #106000,(R0)+   ;SHORT VECTOR INSTR
  974  007202  012720  022104                    MOV     #22104,(R0)+    ;X = 10 & Y = 4 IN MINUS DIRECTION
  975  007206  012710  172000                    MOV     #172000,(R0)    ;LD STOP INSTR
  976  007212  012701  000776                    MOV     #776,R1         ;SET UP X EXPECTED DATA
  977  007216  012702  000777                    MOV     #777,R2         ;SET UP Y EXPECTED DATA
  978  007222  012737  000001  001164            MOV     #1,$TMP0        ;SET UP SCALE FOR ER TYPE
  979  007230  012737  007236  001110            MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
  980  007236  004737  024524          1$:       JSR     PC,EXECUTE      ;START DISPLAY
  981  007242  013737  001764  001122            MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
  982  007250  010137  001124                    MOV     R1,$GDDAT       ;SET UP EXPECTED
  983  007254  012737  000010  001162            MOV     #10,$REG0       ;SET UP VECTOR LENGTH FOR ER TYPE
  984  007262  017737  172476  001126            MOV     @XPOS,$BDDAT    ;READ X POS REG
  985  007270  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
  986  007276  001401                            BEQ     2$              ;BR IF SO
  987  007300  104011                            ERROR   11              ;X VECTOR POS FAILED TO SCALE
  988  007302  013737  001766  001122  2$:       MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
  989  007310  010237  001124                    MOV     R2,$GDDAT       ;LD EXPECTED
  990  007314  012737  000004  001162            MOV     #4,$REG0        ;SET UP VECTOR LENGTH FOR ER TYPE
  991  007322  017737  172440  001126            MOV     @YPOS,$BDDAT    ;READ Y POS REG
  992  007330  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
  993  007336  001401                            BEQ     3$              ;BR IF SO
  994  007340  104011                            ERROR   11              ;Y VECTOR POS FAILED TO SCALE
  995  007342  005237  001164          3$:       INC     $TMP0           ;ADVANCE SCALE BY 1/4
  996  007346  022737  000020  001164            CMP     #20,$TMP0       ;ALL SCALES TESTED?
  997  007354  001406                            BEQ     TST26           ;;NEXT TEST IF SO
  998  007356  005237  032540                    INC     BUFFER+10       ;ADVANCE SCALE BY 1/4 AT ISTR LOC.
  999  007362  162701  000002                    SUB     #2,R1           ;DECREASE EXPECTED X POS BY 1/4
 1000  007366  005302                            DEC     R2              ;DECREASE EXPECTED Y POS BY 1/4
 1001  007370  000722                            BR      1$              ;TRY NEXT SCALE
 1002
 1003                                    ;;*******************************************************************
  (3)                                    ;*TEST 26       TEST THAT X AND Y AXIS INCREMENT PROPERLY USING RELATIVE POINT MODE COUN
  (3)                                    ;;*******************************************************************
  (2)  007372  000004                    TST26:  SCOPE
  (1)  007374  012737  000200  001166            MOV     #200,$TIMES     ;;DO 200 ITERATIONS
  (2)  007402  012737  000026  001202            MOV     #26,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1004  007410  012737  007466  001110            MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
 1005  007416  012703  000077                    MOV     #77,R3          ;SET UP A COUNT LOCATION
 1006  007422  012737  000001  001124            MOV     #1,$GDDAT       ;SET UP THE COMPARED LOCATION
 1007  007430  012704  000201                    MOV     #201,R4         ;SET UP 'DELTA X-Y''
 1008  007434  013700  001712          1$:       MOV     DBUF,R0         ;SET UP R0
 1009  007440  012720  154024                    MOV     #154024,(0)+    ;SCALE OF UNITY
 1010  007444  012720  116000                    MOV     #116000,(0)+    ;LOAD 'SET POINT DATA MODE''
 1011  007450  005020                            CLR     (0)+            ;CLEAR X AXIS
 1012  007452  005020                            CLR     (0)+            ;CLEAR Y AXIS
 1013  007454  012720  130000                    MOV     #130000,(0)+    ;LOAD 'SET RELATIVE POINT MODE''
 1014  007460  010420                            MOV     R4,(0)+         ;PRESET 'DELTA X AND DELTA Y''
 1015  007462  012720  172000                    MOV     #172000,(R0)+   ;LOAD DISPLAY STOP
 1016  007466                          2$:
  (1)  007466  004737  024524                    JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 1017
 1018  007472  017737  172266  001126            MOV     @XPOS,$BDDAT    ;READ X POSITION
 1019  007500  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;ARE THEY EQUAL
 1020  007506  001404                            BEQ     3$              ;YES
```

```
 1021  007510  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
 1022  007516  104016                          ERROR   16              ;INCREMENT X AXIS FAILED USING
 1023
 1024  007520  017737  172242  001126  3$:     MOV     @YPOS,$BDDAT    ;READ Y POSITION
 1025  007526  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY EQUAL ?
 1026  007534  001404                          BEQ     4$              ;YES
 1027  007536  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 1028  007544  104016                          ERROR   16              ;INCREMENT Y AXIS FAILED USING
 1029
 1030  007546  062704  000201          4$:     ADD     #201,R4         ;ADD DELTA X-Y
 1031  007552  005237  001124                  INC     $GDDAT          ;INCREMENT EXPECTED VALUE
 1032  007556  005303                          DEC     R3              ;DECREMENT COUNT, FINISHED?
 1033  007560  001325                          BNE     1$              ;;NO, TEST MORE DATA
 1034
 1035                                  ;;**********************************************************************
  (3)                                  ;*TEST 27       TEST THAT X AND Y AXIS DECREMENT PROPERLY USING RELATIVE POINT MODE COUN
  (3)                                  ;;**********************************************************************
  (2)  007562  000004                  TST27:  SCOPE
  (1)  007564  012737  000200  001166          MOV     #200,$TIMES     ;;DO 200 ITERATIONS
  (2)  007572  012737  000027  001202          MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1036  007600  012737  007652  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
 1037  007606  012703  000077                  MOV     #77,R3          ;SET UP A COUNT LOCATION
 1038  007612  012737  001777  001124          MOV     #1777,$GDDAT    ;SET UP THE COMPARED LOCATION
 1039  007620  012704  020301                  MOV     #20301,R4       ;PRESET THE 'DELTA X-Y'
 1040  007624  013700  001712          1$:     MOV     DBUF,R0         ;SET UP R0
 1041  007630  012720  116000                  MOV     #116000,(0)+    ;LOAD 'SET POINT DATA MODE'
 1042  007634  005020                          CLR     (0)+            ;CLEAR X AXIS
 1043  007636  005020                          CLR     (0)+            ;CLEAR Y AXIS
 1044  007640  012720  130000                  MOV     #130000,(0)+    ;LOAD 'SET RELATIVE POINT MODE'
 1045  007644  010420                          MOV     R4,(0)+         ;PRESET 'DELTA X AND DELTA Y'
 1046  007646  012720  172000                  MOV     #172000,(R0)+   ;LOAD DISPLAY STOP
 1047  007652                          2$:
  (1)  007652  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 1048
 1049  007656  017737  172102  001126          MOV     @XPOS,$BDDAT    ;READ X POSITION
 1050  007664  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY EQUAL
 1051  007672  001404                          BEQ     3$              ;YES
 1052  007674  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
 1053  007702  104017                          ERROR   17              ;DECREMENT X AXIS FAILED USING
 1054
 1055  007704  017737  172056  001126  3$:     MOV     @YPOS,$BDDAT    ;READ Y POSITION
 1056  007712  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY EQUAL ?
 1057  007720  001404                          BEQ     4$              ;YES DECREMENT
 1058  007722  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 1059  007730  104017                          ERROR   17              ;DECREMENT Y AXIS FAILED USING
 1060
 1061  007732  062704  000201          4$:     ADD     #201,R4         ;ADD 'DELTA X-Y'
 1062  007736  005337  001124                  DEC     $GDDAT          ;DECREMENT EXPECTED VALUE
 1063  007742  005303                          DEC     R3              ;DECREMENT COUNT, FINISHED?
 1064  007744  001327                          BNE     1$              ;;NO, TEST MORE DATA
 1065
 1066                                  ;;**********************************************************************
  (3)                                  ;*TEST 30       TEST RELATIVE POINT FOR ALL VALUES OF SCALE
  (3)                                  ;;**********************************************************************
  (2)  007746  000004                  TST30:  SCOPE
  (2)  007750  012737  000030  001202          MOV     #30,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
```

```
1067  007756  012700  032530            MOV     #BUFFER,R0      ;GET 1ST ADRS WHERE INSTR'S GO
1068  007762  012720  154024            MOV     #154024,(R0)+   ;'LD STATUS C'' (SCALE OF UNITY)
1069  007766  012720  116000            MOV     #116000,(R0)+   ;POINT INSTR
1070  007772  012720  001000            MOV     #1000,(R0)+     ;X = 1000
1071  007776  012720  001000            MOV     #1000,(R0)+     ;Y = 1000
1072  010002  012720  154021            MOV     #154021,(R0)+   ;'LD STATUS C'' (SCALE TO BE UPDATED)
1073  010006  012720  130000            MOV     #130000,(R0)+   ;RELATIVE POINT INSTR
1074  010012  012720  001010            MOV     #1010,(R0)+     ;X = 4 & Y = 10
1075  010016  012710  172000            MOV     #172000,(R0)    ;LD STOP INSTR
1076  010022  012701  001001            MOV     #1001,R1        ;SET UP X EXPECTED DATA
1077  010026  012702  001002            MOV     #1002,R2        ;SET UP Y EXPECTED DATA
1078  010032  012737  000001  001164    MOV     #1,$TMP0        ;SET UP SCALE FOR ER TYPE
1079  010040  012737  010046  001110    MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
1080  010046  004737  024524        1$: JSR     PC,EXECUTE      ;START DISPLAY
1081  010052  013737  001764  001122    MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
1082  010060  010137  001124            MOV     R1,$GDDAT       ;LD EXPECTED
1083  010064  012737  000004  001162    MOV     #4,$REG0        ;SET UP X POINT FOR ER TYPE
1084  010072  017737  171666  001126    MOV     @XPOS,$BDDAT    ;READ X POS REG
1085  010100  023737  001124  001126    CMP     $GDDAT,$BDDAT   ;CORRECT?
1086  010106  001401                    BEQ     2$              ;BR IF SO
1087  010110  104011                    ERROR   11              ;X POINT POS FAILED TO SCALE
1088  010112  013737  001766  001122 2$: MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
1089  010120  010237  001124            MOV     R2,$GDDAT       ;LD EXPECTED
1090  010124  012737  000010  001162    MOV     #10,$REG0       ;SET UP Y POINT FOR ER TYPE
1091  010132  017737  171630  001126    MOV     @YPOS,$BDDAT    ;READ Y POS REG
1092  010140  023737  001124  001126    CMP     $GDDAT,$BDDAT   ;CORRECT?
1093  010146  001401                    BEQ     3$              ;BR IF SO
1094  010150  104011                    ERROR   11              ;Y POINT POS FAILED TO SCALE
1095  010152  005237  001164        3$: INC     $TMP0           ;ADVANCE SCALE BY 1/4
1096  010156  022737  000020  001164    CMP     #20,$TMP0       ;ALL SCALES TESTED?
1097  010164  001406                    BEQ     TST31           ;;NEXT TEST IF SO
1098  010166  005237  032540            INC     BUFFER+10       ;ADVANCE SCALE BY 1/4 AT INSTR LOC.
1099  010172  005201                    INC     R1              ;INCREASE EXPECTED X POS BY 1/4
1100  010174  062702  000002            ADD     #2,R2           ;INCREASE EXPECTED Y POS BY 1/4
1101  010200  000722                    BR      1$              ;TRY NEXT SCALE
1102
1103                                ;;*****************************************************************
 (3)                                ;*TEST 31        TEST POINT INSTR (10 BITS) FOR ALL VALUES OF SCALE
 (3)                                ;;*****************************************************************
 (2)  010202  000004            TST31:  SCOPE
 (1)  010204  012737  000004  001166    MOV     #4,$TIMES       ;;DO 4 ITERATIONS
 (2)  010212  012737  000031  001202    MOV     #31,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1104  010220  012700  032530            MOV     #BUFFER,R0      ;SET UP ADRS WHERE INTR'S GO
1105  010224  012720  154020            MOV     #154020,(R0)+   ;LD SCALE INSTR
1106  010230  012720  116000            MOV     #116000,(R0)+   ;LD POINT INSTR
1107  010234  005020                    CLR     (R0)+           ;X = 0
1108  010236  005020                    CLR     (R0)+           ;Y = 0
1109  010240  012720  172000            MOV     #172000,(R0)+   ;LD STOP INSTR
1110  010244  005037  001164            CLR     $TMP0           ;SET UP SCALE FOR ER TYPE
1111  010250  005000                    CLR     R0              ;START WITH 0 POINT
1112  010252  005037  001162            CLR     $REG0           ;PUT POINT IN $REG0 FOR ER TYPE
1113  010256  005001                    CLR     R1              ;START WITH 0 SCALE
1114  010260  012737  010266  001110    MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
1115  010266  004737  024674        1$: JSR     PC,MULSCL       ;GO MUL POINT BY SCALE - RESULT IN $GDDAT
1116  010272  005077  171476            CLR     @XDOFF          ;INSURE ZERO X OFFSET
1117  010276  005077  171474            CLR     @YDOFF          ;INSURE ZERO Y OFFSET
```

```
     010302  004737  024524                    JSR    PC,EXECUTE        ;START DISPLAY
     010306  013737  001764  001122            MOV    XPOS,$BDADR       ;SET UP REG ADRS 04
     010314  017737  171444  001126            MOV    @XPOS,$BDDAT      ;READ LOW ORDER X POS BITS
     010322  017705  171446                    MOV    @XDOFF,R5         ;READ HI ORDER X POS BITS
     010326  004737  025136                    JSR    PC,POCONV         ;GO MAKE UP 14 BIT X POINT POS
     010332  023737  001124  001126            CMP    $GDDAT,$BDDAT     ;CORRECT?
     010340  001401                            BEQ    2$                ;BR IF SO
     010342  104011                            ERROR  11                ;X POINT POS FAILED TO SCALE
     010344  013737  001766  001122  2$:       MOV    YPOS,$BDADR       ;SET UP REG ADRS 06
     010352  017737  171410  001126            MOV    @YPOS,$BDDAT      ;READ LOW ORDER Y POS BITS
     010360  017705  171412                    MOV    @YDOFF,R5         ;READ HI ORDER Y POS BITS
     010364  004737  025136                    JSR    PC,POCONV         ;GO MAKE UP 14 BIT Y POINT POS
     010370  023737  001124  001126            CMP    $GDDAT,$BDDAT     ;CORRECT?
     010376  001401                            BEQ    3$                ;BR IF SO
     010400  104011                            ERROR  11                ;Y POINT POS FAILED TO SCALE
     010402  005237  001162          3$:       INC    $REG0             ;UPDATE POINT FOR ER TYPE
     010406  005200                            INC    R0                ;UPDATE POINT BY 1
     010410  022700  002000                    CMP    #2000,R0          ;DONE 10 BITS THIS SCALE?
     010414  001405                            BEQ    5$                ;BR IF SO
     010416  010037  032534          4$:       MOV    R0,BUFFER+4       ;SET UP X POINT
     010422  010037  032536                    MOV    R0,BUFFER+6       ;SET UP Y POINT
     010426  000717                            BR     1$                ;REPEAT TEST SAME SCALE AT POINT +1
     010430  005037  001162          5$:       CLR    $REG0             ;RESET POINT FOR ER TYPE
     010434  005000                            CLR    R0                ;RESET POINT
     010436  005237  001164                    INC    $TMP0             ;BUMP SCALE BY 1/4
     010442  005237  032530                    INC    BUFFER            ;NOW AT INSTR LOC
     010446  005201                            INC    R1                ;NOW AT R1
     010450  022701  000020                    CMP    #20,R1            ;ALL SCALES TESTED?
     010454  001360                            BNE    4$                ;REPEAT TEST AT NEW SCALE


                                     ;;*****************************************************************
 (3)                                 ;*TEST 32        TEST ABSOLUTE VECTOR FOR ALL VALUES OF SCALE
 (3)                                 ;;*****************************************************************
 (2) 010456  000004                  TST32:    SCOPE
 (2) 010460  012737  000032  001202            MOV    #32,$TESTN        ;;SET TEST NUMBER IN APT MAIL BOX
     010466  012700  032530                    MOV    #BUFFER,R0        ;GET 1ST ADRS WHERE INSTR'S GO
     010472  012720  154024                    MOV    #154024,(R0)+     ;SCALE OF UNITY
     010476  012720  116000                    MOV    #116000,(R0)+     ;POINT INSTR
     010502  005020                            CLR    (R0)+             ;X = 0
     010504  005020                            CLR    (R0)+             ;Y = 0
     010506  012720  154021                    MOV    #154021,(R0)+     ;SCALE TO BE UPDATED
     010512  012720  146000                    MOV    #146000,(R0)+     ;ABSOLUTE VECTOR INSTR
     010516  012720  000400                    MOV    #400,(R0)+        ;X = 400
     010522  012720  000200                    MOV    #200,(R0)+        ;Y = 200
     010526  012710  172000                    MOV    #172000,(R0)      ;STOP INSTR
     010532  012737  000001  001164            MOV    #1,$TMP0          ;SET UP SCALE OF ER TYPE
     010540  012701  000100                    MOV    #100,R1           ;SET UP X EXPECTED DATA
     010544  012702  000040                    MOV    #40,R2            ;SET UP Y EXPECTED DATA
     010550  012737  010556  001110            MOV    #1$,$LPERR        ;SET UP SCOPE LOOP ADRS
     010556  004737  024524          1$:       JSR    PC,EXECUTE        ;START DISPLAY
     010562  013737  001764  001122            MOV    XPOS,$BDADR       ;SET UP REG ADRS 04
     010570  010137  001124                    MOV    R1,$GDDAT         ;LD EXPECTED
     010574  012737  000400  001162            MOV    #400,$REG0        ;SET UP X VECTOR FOR ER TYPE
     010602  017737  171156  001126            MOV    @XPOS,$BDDAT      ;READ X POS REG
     010610  023737  001124  001126            CMP    $GDDAT,$BDDAT     ;CORRECT?
     010616  001401                            BEQ    2$                ;BR IF SO
```

```
1170  010620  104011                          ERROR   11              ;X VECTOR POS FAILED TO SCALE
1171  010622  013737  001766  001122  2$:     MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
1172  010630  010237  001124                  MOV     R2,$GDDAT       ;LD EXPECTED
1173  010634  012737  000200  001162          MOV     #200,$REG0      ;SET UP Y VECTOR FOR ER TYPE
1174  010642  017737  171120  001126          MOV     @YPOS,$BDDAT    ;READ Y POS REG
1175  010650  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1176  010656  001401                          BEQ     3$              ;BR IF SO
1177  010660  104011                          ERROR   11              ;Y VECTOR POS FAILED TO SCALE
1178  010662  005237  001164          3$:     INC     $TMP0           ;ADVANCE SCALE BY 1/4
1179  010666  022737  000020  001164          CMP     #20,$TMP0       ;ALL SCALES TESTED?
1180  010674  001407                          BEQ     TST33           ;;NEXT TEST IF SO
1181  010676  005237  032540                  INC     BUFFER+10       ;BUMP SCALE AT INSTR LOC BY 1/4
1182  010702  062701  000100                  ADD     #100,R1         ;BUMP EXPECTED X POS BY 1/4
1183  010706  062702  000040                  ADD     #40,R2          ;BUMP EXPECTED Y POS BY 1/4
1184  010712  000721                          BR      1$              ;TRY NEXT SCALE
1185
1186                                  ;;*******************************************************************
 (3)                                  ;*TEST 33       USE GRAPHPLOT X MODE TO TEST Y AXIS IS INCREMENTED BY ''GRAPHPLOT INCREME
 (3)                                  ;;*******************************************************************
 (2)  010714  000004                  TST33:  SCOPE
 (1)  010716  012737  000400  001166          MOV     #400,$TIMES     ;;DO 400 ITERATIONS
 (2)  010724  012737  000033  001202          MOV     #33,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1187  010732  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
1188  010740  012703  000077                  MOV     #77,R3          ;SET UP EXECUTION COUNTER
1189  010744  012737  000001  001124          MOV     #1,$GDDAT       ;SET UP COMPARED DATA
1190  010752  012737  174101  001726          MOV     #174101,DSAVE   ;SET UP BASIC 'LOAD STATUS B''
1191  010760  013700  001712          1$:     MOV     DBUF,R0         ;SET UP R0
1192  010764  012720  154024                  MOV     #154024,(R0)+   ;LD STATUS C - VECTOR SCALE OF UNITY
1193  010770  012720  116000                  MOV     #116000,(0)+    ;LOAD 'POINT MODE''
1194  010774  005020                          CLR     (0)+            ;CLEAR X AXIS
1195  010776  005020                          CLR     (0)+            ;CLEAR Y AXIS
1196  011000  013720  001726                  MOV     DSAVE,(0)+      ;LOAD 'SET STATUS B''
1197  011004  012720  120000                  MOV     #120000,(0)+    ;LOAD 'SET GRAPHPLOT X MODE''
1198  011010  005020                          CLR     (0)+            ;LOAD 'X GRAPHPLOT DATA''
1199  011012  012720  172000                  MOV     #172000,(R0)+   ;LOAD DISPLAY STOP
1200  011016  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
1201
1202  011022  017737  170740  001126          MOV     @YPOS,$BDDAT    ;READ Y AXIS
1203  011030  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE TO EXPECTED VALUE
1204  011036  001402                          BEQ     2$              ;ARE THEY EQUAL?
1205  011040  104027                          ERROR   27              ;GRAPHPLOT INC FAILED TO SET UP X POS REG
1206  011042  000406                          BR      TST34           ;;
1207  011044  005237  001726          2$:     INC     DSAVE
1208  011050  005237  001124                  INC     $GDDAT          ;INCREMENT THE STATUS B COUNT
1209  011054  005303                          DEC     R3              ;DECREMENT THE EXECUTION COUNT
1210  011056  001340                          BNE     1$              ;;TEST MORE DATA
1211
1212                                  ;;*******************************************************************
 (3)                                  ;*TEST 34       USE GRAPHPLOT Y MODE TO TEST X AXIS IS INCREMENTED BY ''GRAPHPLOT INCREME
 (3)                                  ;;*******************************************************************
 (2)  011060  000004                  TST34:  SCOPE
 (1)  011062  012737  000400  001166          MOV     #400,$TIMES     ;;DO 400 ITERATIONS
 (2)  011070  012737  000034  001202          MOV     #34,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1213  011076  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
1214  011104  012703  000077                  MOV     #77,R3          ;SET UP EXECUTION COUNTER
1215  011110  012737  000001  001124          MOV     #1,$GDDAT       ;SET UP COMPARED DATA
```

```
 1216  011116  012737  174101  001726          MOV     #174101,DSAVE   ;SET UP BASIC 'LOAD STATUS B''
 1217  011124  013700  001712          1$:     MOV     DBUF,R0         ;SET UP R0
 1218  011130  012720  116000                  MOV     #116000,(0)+    ;LOAD 'POINT MODE''
 1219  011134  005020                          CLR     (0)+            ;CLEAR X AXIS
 1220  011136  005020                          CLR     (0)+            ;CLEAR Y AXIS
 1221  011140  013720  001726                  MOV     DSAVE,(0)+      ;LOAD 'SET STATUS B''
 1222  011144  012720  124000                  MOV     #124000,(0)+    ;LOAD 'SET GRAPHPLOT Y MODE''
 1223  011150  005020                          CLR     (0)+            ;LOAD 'Y GRAPHPLOT DATA''
 1224  011152  012720  172000                  MOV     #172000,(R0)+   ;LOAD DISPLAY STOP
 1225  011156  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 1226
 1227  011162  017737  170576  001126          MOV     @XPOS,$BDDAT    ;READ X AXIS
 1228  011170  042737  176000  001126          BIC     #176000,$BDDAT  ;MASK TO BITS 0-9
 1229  011176  023737  0C1124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE TO EXPECTED VALUE
 1230  011204  001402                          BEQ     2$              ;ARE THEY EQUAL?
 1231  011206  104027                          ERROR   27              ;GRAPHPLOT INC FAILED TO SET UP Y POS REG
 1232  011210  000416                          BR      TST35           ;;
 1233
 1234  011212  005237  001726          2$:     INC     DSAVE
 1235  011216  005237  001124                  INC     $GDDAT          ;INCREMENT THE STATUS B COUNT
 1236  011222  005303                          DEC     R3              ;DECREMENT THE EXECUTION COUNT
 1237  011224  001337                          BNE     1$              ;;TEST MORE DATA
 1238
 1239
 1240  011226  012737  174100  032530          MOV     #174100,BUFFER
 1241  011234  012737  172000  032532          MOV     #172000,BUFFER+2
 1242  011242  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 1243
 1244                                  ;;*********************************************************************
  (3)                                  ;*TEST 35       TEST THAT GRAPHPLOT X WILL SET UP X & Y FOR ALL VALUES OF SCALE
  (3)                                  ;;*********************************************************************
  (2)  011246  000004          TST35:  SCOPE
  (2)  011250  012737  000035  001202          MOV     #35,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1245  011256  012700  032530                  MOV     #BUFFER,R0      ;GET 1ST ADRS WHERE INSTR'S GO
 1246  011262  012720  154021                  MOV     #154021,(R0)+   ;'LD STATUS C'' (VECTOR SCALE TO BE UPDATED)
 1247  011266  012720  174120                  MOV     #174120,(R0)+   ;'LD STATUS B'' (GRAPH PLOT INC OF 20)
 1248  011272  012720  116000                  MOV     #116000,(R0)+   ;POINT INSTR
 1249  011276  005020                          CLR     (R0)+           ;X OF ZERO
 1250  011300  005020                          CLR     (R0)+           ;Y OF ZERO
 1251  011302  012720  120000                  MOV     #120000,(R0)+   ;GRAPH PLOT X INSTR
 1252  011306  012720  000100                  MOV     #100,(R0)+      ;X DATA OF 100
 1253  011312  012710  172000                  MOV     #172000,(R0)    ;STOP INSTR
 1254  011316  012701  000004                  MOV     #4,R1           ;SET UP EXPECTED Y GRAPH PLOT INC VALUE
 1255  011322  012702  000020                  MOV     #20,R2          ;SET UP EXPECTED X GRAPH PLOT INC VALUE
 1256  011326  012737  000001  001164          MOV     #1,$TMP3                ;SET UP SCALE FOR ER TYPE
 1257  011334  012737  011342  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
 1258  011342  004737  024524          1$:     JSR     PC,EXECUTE      ;START DISPLAY
 1259  011346  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
 1260  011354  010237  001124                  MOV     R2,$GDDAT       ;LD EXPECTED
 1261  011360  012737  000100  001162          MOV     #100,$REG0      ;SET UP X DATA FOR ER TYPE
 1262  011366  017737  170372  001126          MOV     @XPOS,$BDDAT    ;READ X POS REG
 1263  011374  042737  176000  001126          BIC     #176000,$BDDAT  ;ONLY WANT XPOS BITS
 1264  011402  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
 1265  011410  001401                          BEQ     2$              ;BR IF SO
 1266  011412  104011                          ERROR   11              ;X GRAPHPLOT DATA FAILED TO SCALE
 1267  011414  013737  001766  001122  2$:     MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
```

```
 1268  011422  010137  001124                       MOV     R1,$GDDAT       ;LD EXPECTED
 1269  011426  012737  000020  001162               MOV     #20,$REG0       ;SET UP Y GRAPHPLOT INC FOR ER TYPE
 1270  011434  017737  170326  001126               MOV     @YPOS,$BDDAT    ;READ Y POS REG
 1271  011442  023737  001124  001126               CMP     $GDDAT,$BDDAT   ;CORRECT?
 1272  011450  001401                               BEQ     3$              ;BR IF SO
 1273  011452  104011                               ERROR   11              ;Y GRAPHPLOT INC FAILED TO SCALE
 1274  011454  005237  001164               3$:     INC     $TMP0           ;ADVANCE SCALE BY 1/4
 1275  011460  022737  000020  001164               CMP     #20,$TMP0       ;ALL SCALE VALUES TESTED?
 1276  011466  001407                               BEQ     TST36           ;;NEXT TEST IF SO
 1277  011470  005237  032530                       INC     BUFFER          ;ADVANCE SCALE BY 1/4 AT INSTR LOC.
 1278  011474  062702  000020                       ADD     #20,R2          ;BUMP EXPECTED X GRAPHPLOT BY 1/4
 1279  011500  062701  000004                       ADD     #4,R1           ;BUMP EXPECTED Y GRAPHPLOT INC BY 1/4
 1280  011504  000716                               BR      1$              ;TRY NEXT SCALE
 1281
 1282                                       ;;*******************************************************************
  (3)                                       ;*TEST 36        TEST THAT GRAPHPLOT Y WILL SET UP X & Y FOR ALL VALUES OF SCALE
  (3)                                       ;;*******************************************************************
  (2)  011506  000004                       TST36:  SCOPE
  (2)  011510  012737  000036  001202               MOV     #36,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1283  011516  012700  032530                       MOV     #BUFFER,R0      ;GET 1ST ADRS WHERE INSTR'S GO
 1284  011522  012720  154021                       MOV     #154021,(R0)+   ;"LD STATUS C" (VECTOR SCALE TO BE UPDATED)
 1285  011526  012720  174120                       MOV     #174120,(R0)+   ;"LD STATUS B" (GRAPH PLOT INC OF 20)
 1286  011532  012720  116000                       MOV     #116000,(R0)+   ;POINT INSTR
 1287  011536  005020                               CLR     (R0)+           ;X OF ZERO
 1288  011540  005020                               CLR     (R0)+           ;Y OF ZERO
 1289  011542  012720  124000                       MOV     #124000,(R0)+   ;GRAPH PLOT Y INSTR
 1290  011546  012720  000100                       MOV     #100,(R0)+      ;Y DATA OF 100
 1291  011552  012710  172000                       MOV     #172000,(R0)    ;STOP INSTR
 1292  011556  012701  000004                       MOV     #4,R1           ;SET UP EXPECTED X GRAPH PLOT INC VALUE
 1293  011562  012702  000020                       MOV     #20,R2          ;SET UP EXPECTED Y GRAPH PLOT DATA
 1294  011566  012737  000001  001164               MOV     #1,$TMP0        ;SET UP SCALE FOR ER TYPE
 1295  011574  012737  011602  001110               MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
 1296  011602  004737  024524               1$:     JSR     PC,EXECUTE      ;START DISPLAY
 1297  011606  013737  001766  001122               MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 1298  011614  010237  001124                       MOV     R2,$GDDAT       ;LD EXPECTED
 1299  011620  012737  000100  001162               MOV     #100,$REG0      ;SET UP DATA FOR ER TYPE
 1300  011626  017737  170134  001126               MOV     @YPOS,$BDDAT    ;READ Y POS REG
 1301  011634  023737  001124  001126               CMP     $GDDAT,$BDDAT   ;CORRECT?
 1302  011642  001401                               BEQ     2$              ;BR IF SO
 1303  011644  104011                               ERROR   11              ;Y GRAPHPLOT DATA FAILED TO SCALE
 1304  011646  013737  001764  001122       2$:     MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
 1305  011654  010137  001124                       MOV     R1,$GDDAT       ;LD EXPECTED
 1306  011660  012737  000020  001162               MOV     #20,$REG0       ;SET UP X GRAPH PLOT INC FOR ER TYPE
 1307  011666  017737  170072  001126               MOV     @XPOS,$BDDAT    ;READ X POS REG
 1308  011674  042737  176000  001126               BIC     #176000,$BDDAT  ;SAVE ONLY X POS BITS
 1309  011702  023737  001124  001126               CMP     $GDDAT,$BDDAT   ;CORRECT?
 1310  011710  001401                               BEQ     3$              ;BR IF SO
 1311  011712  104011                               ERROR   11              ;X GRAPHPLOT INC FAILED TO SCALE
 1312  011714  005237  001164               3$:     INC     $TMP0           ;ADVANCE SCALE BY 1/4
 1313  011720  022737  000020  001164               CMP     #20,$TMP0       ;ALL SCALE VALUES TESTED?
 1314  011726  001407                               BEQ     TST37           ;;NEXT TEST IF SO
 1315  011730  005237  032530                       INC     BUFFER          ;ADVANCE SCALE BY 1/4 AT INSTR LOC.
 1316  011734  062702  000020                       ADD     #20,R2          ;BUMP EXPECTED Y GRAPH PLOT BY 1/4
 1317  011740  062701  000004                       ADD     #4,R1           ;BUMT EXPECTED X GRAPH PLOT INC BY 1/4
 1318  011744  000716                               BR      1$              ;TRY NEXT SCALE
 1319
```

```
                                        ;;*****************************************************************
                                        ;*TEST 37       TEST BASIC VECTOR (GRAPH X & Y) AT ORIGIN 1000/1000
                                        ;;*****************************************************************
011746  000004                  TST37:  SCOPE
011750  012737  000037  001202          MOV     #37,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
011756  012737  012034  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
011764  012700  032530                  MOV     #BUFFER,R0      ;SET UP INSTR ADRS
011770  012720  174100                  MOV     #174100,(R0)+   ;SET GRAPHPLOT INC TO 0
011774  012720  154024                  MOV     #154024,(R0)+   ;'LD STATUS C'' (VECTOR SCALE TO UNITY)
012000  012720  116000                  MOV     #116000,(R0)+   ;SET UP POINT INSTR
012004  012720  001000                  MOV     #1000,(R0)+     ;X = 1000
012010  012720  001000                  MOV     #1000,(R0)+     ;Y = 1000
012014  012720  122000                  MOV     #122000,(R0)+   ;BASIC VECTOR INSTR (GRAPH X)
012020  012720  036400                  MOV     #36400,(R0)+    ;VECTOR LENGTH OF 400
012024  012710  172000                  MOV     #172000,(R0)    ;STOP INSTR
012030  012700  012206                  MOV     #TBL2R,R0       ;SET UP ADRS OF EXPECTED X & Y POS VALUES
012034  004737  024524          1$:     JSR     PC,EXECUTE      ;START
012040  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
012046  011037  001124                  MOV     (R0),$GDDAT     ;LD EXPECTED
012052  017737  167706  001126          MOV     @XPOS,$BDDAT    ;READ X POS
012060  042737  176000  001126          BIC     #176000,$BDDAT  ;RID TRASH
012066  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
012074  001401                        BEQ     2$              ;BR IF SO
012076  104030                        ERROR   30              ;BASIC VECTOR FAILED TO SET UP X POS
012100  013737  001766  001122  2$:     MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
012106  016037  000002  001124          MOV     2(R0),$GDDAT    ;LD EXPECTED
012114  017737  167646  001126          MOV     @YPOS,$BDDAT    ;READ Y POS
012122  042737  176000  001126          BIC     #176000,$BDDAT  ;RID TRASH
012130  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
012136  001401                        BEQ     3$              ;BR IF SO
012140  104030                        ERROR   30              ;BASIC VECTOR FAILED TO SET UP Y POS
012142  022020                  3$:     CMP     (R0)+,(R0)+     ;MOV ADRS PTR TO NEXT EXPECTED X & Y POS
012144  162737  004000  032544          SUB     #4000,BUFFER+14 ;SET UP NEXT DIRECTION
012152  100330                        BPL     1$              ;REPEAT TEST IF THIS DIRECTION NOT TESTED
012154  062737  004000  032542          ADD     #4000,BUFFER+12 ;NOW SET UP BASIC VECTOR (GRAPH Y)
012162  032737  010000  032542          BIT     #10000,BUFFER+12  ;HAS IT BEEN DONE ALREADY?
012170  001426                        BEQ     TST40           ;;NEXT TEST IF SO
012172  012737  036400  032544          MOV     #36400,BUFFER+14        ;RESET DIRECTION TO PATH 7
012200  012700  012206                  MOV     #TBL2R,R0       ;RESET ADRS PTR OF RESULTS
012204  000713                        BR      1$              ;REPEAT TEST

012206  001400                  TBL2R:  1400                    ;EXPECT X & Y BASIC VECTOR POS RESULTS
012210  000400                          400
012212  001000                          1000
012214  000400                          400
012216  000400                          400
012220  000400                          400
012222  000400                          400
012224  001000                          1000
012226  000400                          400
012230  001400                          1400
012232  001000                          1000
012234  001400                          1400
012236  001400                          1400
012240  001400                          1400
012242  001400                          1400
```

```
 1372  012244  001000                          1000
 1373
 1374                                  ;;*****************************************************************
  (3)                                  ;*TEST 40       TEST BASIC VECTOR (GRAPH X & Y) AT ORIGIN 1000/1000, PATH 7 FOR ALL VALU
  (3)                                  ;;*****************************************************************
  (2)  012246  000004                  TST40:  SCOPE
  (2)  012250  012737  000040  001202          MOV     #40,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1375  012256  012700  032530                  MOV     #BUFFER,R0      ;GET 1ST ADRS WHERE INSTR'S GO
 1376  012262  012720  154024                  MOV     #154024,(R0)+   ;'LD STATUS C'' (VECTOR SCALE OF UNITY)
 1377  012266  012720  116000                  MOV     #116000,(R0)+   ;POINT INSTR
 1378  012272  012720  001000                  MOV     #1000,(R0)+     ;X = 1000
 1379  012276  012720  001000                  MOV     #1000,(R0)+     ;Y = 1000
 1380  012302  012720  154021                  MOV     #154021,(R0)+   ;'LD STATUS C'' (VECTOR SCALE TO BE UPDATED)
 1381  012306  012720  122000                  MOV     #122000,(R0)+   ;BASIC VECTOR INSTR (GRAPH X)
 1382  012312  012720  036200                  MOV     #36200,(R0)+    ;VECTOR LENGTH OF 200 - PATH 7
 1383  012316  012720  172000                  MOV     #172000,(R0)+   ;STOP INSTR
 1384  012322  012701  001040          1$:     MOV     #1040,R1        ;SET UP X EXPECTED DATA
 1385  012326  012702  000740                  MOV     #740,R2         ;SET UP Y EXPECTED DATA
 1386  012332  012737  000001  001164          MOV     #1,$TMP0        ;SET UP SCALE FOR ER TYPE
 1387  012340  012737  012354  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
 1388  012346  012737  000200  001162          MOV     #200,$REG0      ;SET UP VECTOR LENGTH FOR ER TYPE
 1389  012354  004737  024524          2$:     JSR     PC,EXECUTE      ;START DISPLAY
 1390  012360  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
 1391  012366  010137  001124                  MOV     R1,$GDDAT       ;LD EXPECTED
 1392  012372  017737  167366  001126          MOV     @XPOS,$BDDAT    ;READ X POS REG
 1393  012400  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
 1394  012406  001401                          BEQ     3$              ;BR IF SO
 1395  012410  104011                          ERROR   11              ;X VECTOR POS FAILED TO SCALE
 1396  012412  013737  001766  001122  3$:     MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 1397  012420  010237  001124                  MOV     R2,$GDDAT       ;LD EXPECTED
 1398  012424  017737  167336  001126          MOV     @YPOS,$BDDAT    ;READ Y POS REG
 1399  012432  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
 1400  012440  001401                          BEQ     4$              ;BR IF SO
 1401  012442  104011                          ERROR   11              ;Y VECTOR POS FAILED TO SCALE
 1402  012444  005237  001164          4$:     INC     $TMP0           ;ADVANCE SCALE BY 1/4
 1403  012450  022737  000020  001164          CMP     #20,$TMP0       ;ALL SCALE VALUES TESTED?
 1404  012456  001407                          BEQ     5$              ;BR IF SO
 1405  012460  005237  032540                  INC     BUFFER+10       ;ADVANCE SCALE BY 1/4 AT INSTR LOC.
 1406  012464  062701  000040                  ADD     #40,R1          ;BUMP EXPECTED X POS BY 1/4
 1407  012470  162702  000040                  SUB     #40,R2          ;DECREASE EXPECTED Y POS BY 1/4
 1408  012474  000727                          BR      2$              ;TRY NEXT SCALE
 1409  012476  062737  004000  032542  5$:     ADD     #4000,BUFFER+12 ;NOW SET UP BASIC VECTOR (GRAPH Y)
 1410  012504  032737  010000  032542          BIT     #10000,BUFFER+12  ;HAS IT BEEN DONE ALREADY?
 1411  012512  001404                          BEQ     TST41           ;;NEXT TEST IF SO
 1412  012514  012737  154021  032540          MOV     #154021,BUFFER+10  ;RESTORE SCALE INSTR TO 1/4
 1413  012522  000677                          BR      1$              ;REPEAT TEST
 1414
 1415                                  ;;*****************************************************************
  (3)                                  ;*TEST 41       TEST BASIC SHORT VECTOR AT ORIGIN 1000/1000
  (3)                                  ;;*****************************************************************
  (2)  012524  000004                  TST41:  SCOPE
  (2)  012526  012737  000041  001202          MOV     #41,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1416  012534  012700  032530                  MOV     #BUFFER,R0      ;SET UP INSTR ADRS PTR
 1417  012540  012720  154024                  MOV     #154024,(R0)+   ;'LD STATUS C'' (VECTOR SCALE TO UNTIY)
 1418  012544  012720  116000                  MOV     #116000,(R0)+   ;POINT INSTR
 1419  012550  012720  001000                  MOV     #1000,(R0)+     ;X OF 1000
```

```
1420  012554  012720  001000                    MOV     #1000,(R0)+     ;Y OF 1000
1421  012560  012720  136000                    MOV     #136000,(R0)+   ;BASIC SHORT VECTOR INSTR
1422  012564  012720  000177                    MOV     #177,(R0)+      ;LOW BYTE, 17 COUNTS, PATH 7 INTITIALLY
1423  012570  012720  172000                    MOV     #172000,(R0)+   ;STOP INSTR
1424  012574  005001                            CLR     R1              ;0 = LOW BYTE, 1 = HI BYTE UNDER TEST
1425  012576  012700  012764                    MOV     #TBL3R,R0       ;SET UP ADRS PTR OF X & Y POS RESULTS
1426  012602  012737  012610  001110            MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
1427  012610  004737  024524          1$:       JSR     PC,EXECUTE      ;START
1428  012614  013737  001764  001122            MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
1429  012622  011037  001124                    MOV     (R0),$GDDAT     ;SET UP EXPECTED
1430  012626  017737  167132  001126            MOV     @XPOS,$BDDAT    ;READ X POS
1431  012634  042737  176000  001126            BIC     #176000,$BDDAT  ;RID TRASH
1432  012642  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
1433  012650  001401                            BEQ     2$              ;BR IF SO
1434  012652  104031                            ERROR   31              ;X POS FAILED ON SHORT VECTOR
1435  012654  013737  001766  001122  2$:       MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
1436  012662  016037  000002  001124            MOV     2(R0),$GDDAT    ;SET UP EXPECTED
1437  012670  017737  167072  001126            MOV     @YPOS,$BDDAT    ;READ Y POS
1438  012676  042737  176000  001126            BIC     #176000,$BDDAT  ;RID TRASH
1439  012704  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
1440  012712  001401                            BEQ     3$              ;BR IF SO
1441  012714  104031                            ERROR   31              ;Y POS FAILED ON SHORT VECTOR
1442  012716  022020                  3$:       CMP     (R0)+,(R0)+     ;MOVE ADRS PTR TO NEXT EXPECTED X & Y POS
1443  012720  005701                            TST     R1              ;LOW BYTE?
1444  012722  001013                            BNE     4$              ;BR IF HI BYTE
1445  012724  162737  000020  032542            SUB     #20,BUFFER+12   ;SET UP NEXT VECTOR PATH
1446  012732  100326                            BPL     1$              ;BR IF ALL DIRECTIONS NOT TESTED - LOW BYTE
1447  012734  012737  037600  032542            MOV     #37600,BUFFER+12  ;SET UP HI BYTE NOW
1448  012742  012700  012764                    MOV     #TBL3R,R0       ;RESET PTR OF EXPECTED X & Y POS RESULTS
1449  012746  005201                            INC     R1              ;R1 NON-ZERO MEANS HI BYTE
1450  012750  000717                            BR      1$              ;NOW TEST HI BYTE
1451  012752  162737  004000  032542  4$:       SUB     #4000,BUFFER+12 ;SET UP NEXT VECTOR PATH
1452  012760  100421                            BMI     TST42           ;;NEXT TEST IF HI BYTE TESTED ALSO
1453  012762  000712                            BR      1$              ;BR TO TEST NEXT HI BYTE DIRECTION
1454
1455  012764  001017                  TBL3R:    1017                    ;TABLE CONTAINS EXPECTED X & Y POS RESULTS FOR BASIC SHO
1456  012766  000761                            761
1457  012770  001000                            1000
1458  012772  000761                            761
1459  012774  000761                            761
1460  012776  000761                            761
1461  013000  000761                            761
1462  013002  001000                            1000
1463  013004  000761                            761
1464  013006  001017                            1017
1465  013010  001000                            1000
1466  013012  001017                            1017
1467  013014  001017                            1017
1468  013016  001017                            1017
1469  013020  001017                            1017
1470  013022  001000                            1000
1471
1472
 (3)                                    ;;*****************************************************************
 (3)                                    ;*TEST 42       TEST BASIC SHORT VECTOR AT ORIGIN 1000/1000, PATH 3 FOR ALL VALUES OF SC
 (3)                                    ;;*****************************************************************
 (2)  013024  000004                    TST42:  SCOPE
```

```
  (2)  013026  012737  000042  001202          MOV     #42,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1473  013034  012700  032530                  MOV     #BUFFER,R0      ;GET 1ST ADRS WHERE INSTR'S GO
 1474  013040  012720  154024                  MOV     #154024,(R0)+   ;'LD STATUS C'' (VECTOR SCALE OF UNITY)
 1475  013044  012720  116000                  MOV     #116000,(R0)+   ;POINT INSTR
 1476  013050  012720  001000                  MOV     #1000,(R0)+     ;X = 1000
 1477  013054  012720  001000                  MOV     #1000,(R0)+     ;Y = 1000
 1478  013060  012720  154021                  MOV     #154021,(R0)+   ;'LD STATUS C'' (VECTOR SCALE TO BE UPDATED)
 1479  013064  012720  136000                  MOV     #136000,(R0)+   ;BASIC SHORT VECTOR INSTR
 1480  013070  012720  000070                  MOV     #70,(R0)+       ;LOW BYTE, 10 COUNTS, PATH 3
 1481  013074  012720  172000                  MOV     #172000,(R0)+   ;LD STOP INSTR
 1482  013100  012701  000776                  MOV     #776,R1         ;SET UP X EXPECTED DATA
 1483  013104  012702  001002                  MOV     #1002,R2        ;SET UP Y EXPECTED DATA
 1484  013110  012737  000001  001164          MOV     #1,$TMP0        ;SET UP SCALE FOR ER TYPE
 1485  013116  012737  000010  001162          MOV     #10,$REG0       ;SET UP VECTOR LENGTH FOR ER TYPE
 1486  013124  012737  013132  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
 1487  013132  004737  024524          1$:     JSR     PC,EXECUTE      ;START DISPLAY
 1488  013136  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
 1489  013144  010137  001124                  MOV     R1,$GDDAT       ;LD EXPECTED
 1490  013150  017737  166610  001126          MOV     @XPOS,$BDDAT    ;READ X POS REG
 1491  013156  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
 1492  013164  001401                          BEQ     2$              ;BR IF SO
 1493  013166  104011                          ERROR   11              ;X VECTOR POS FAILED TO SCALE
 1494  013170  013737  001766  001122  2$:     MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 1495  013176  010237  001124                  MOV     R2,$GDDAT       ;LD EXPECTED
 1496  013202  017737  166560  001126          MOV     @YPOS,$BDDAT    ;READ Y POS REG
 1497  013210  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
 1498  013216  001401                          BEQ     3$              ;BR IF SO
 1499  013220  104011                          ERROR   11              ;Y VECTOR POS FAILED TO SCALE
 1500  013222  005237  001164          3$:     INC     $TMP0           ;ADVANCE SCALE BY 1/4
 1501  013226  022737  000020  001164          CMP     #20,$TMP0       ;ALL SCALE VALUES TESTED?
 1502  013234  001407                          BEQ     TST43           ;;NEXT TEST IF SO
 1503  013236  005237  032540                  INC     BUFFER+10       ;ADVANCE SCALE BY 1/4 AT INSTR LOC.
 1504  013242  162701  000002                  SUB     #2,R1           ;DECREASE EXPECTED X POS BY 1/4
 1505  013246  062702  000002                  ADD     #2,R2           ;BUMP EXPECTED Y POS BY 1/4
 1506  013252  000727                          BR      1$              ;TRY NEXT SCALE
 1507
 1508                                  ;;*****************************************************************
  (3)                                  ;*TEST 43       TEST VECTOR SCALE SUMMING LOGIC USING BASIC VECTOR MODE
  (3)                                  ;;*****************************************************************
  (2)  013254  000004                  TST43:  SCOPE
  (2)  013256  012737  000043  001202          MOV     #43,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1509                                  ; DRAW A SERIES OF 8 VECTORS, 31 UNITS LONG.  TEST THAT SUMMER
 1510                                  ; ADDS AND SUBTRACTS CORRECTLY FOR EACH SCALE VALUE.
 1511                                  ; OMIT IF SCALING ECO (#    ) IS NOT INSTALLED (SWR BIT 10).
 1512                                  ;
 1513  013264  032777  002000  165646          BIT     #SW10,@SWR
 1514  013272  001002                          BNE     .+6
 1515  013274  000137  013562                  JMP     4$              ; EXIT IF SWR10 NOT SET.
 1516  013300  012700  032530                  MOV     #BUFFER,R0      ; SET DISPLAY CODE IN BUFFER.
 1517  013304  012720  154024                  MOV     #154024,(R0)+   ; SCALE UNITY (1).
 1518  013310  012720  116000                  MOV     #116000,(R0)+   ; POINT.
 1519  013314  005020                          CLR     (R0)+           ;       X 0
 1520  013316  005020                          CLR     (R0)+           ;       Y 0
 1521  013320  012720  154021                  MOV     #154021,(R0)+   ; SCALE 1/4 (VARIABLE).
 1522  013324  012720  122000                  MOV     #122000,(R0)+   ; BASIC VECTOR MODE.
 1523  013330  012701  000010                  MOV     #8.,R1
```

```
1524  013334  012720  006037          1$:     MOV     #6037,(R0)+     ; DIRECTION 1, LENGTH 31...
1525  013340  005301                          DEC     R1
1526  013342  001374                          BNE     1$              ;       ...8 TIMES.
1527  013344  012720  172000                  MOV     #172000,(R0)+   ; STOP AND TEST X/Y.
1528  013350  012701  000010                  MOV     #8.,R1
1529  013354  012720  026037          2$:     MOV     #26037,(R0)+    ; DIRECTION 5, LENGTH 31...
1530  013360  005301                          DEC     R1
1531  013362  001374                          BNE     2$              ;       ...8 TIMES.
1532  013364  012720  172000                  MOV     #172000,(R0)+   ; STOP AND TEST X/Y AGAIN.
1533
1534  013370  012737  013414  001110          MOV     #3$,$LPERR      ; SET SCOPE LOOP ADDRESS.
1535  013376  012737  000001  001164          MOV     #1,$TMP0        ; SCALE FACTOR FOR ERROR OUTPUT.
1536  013404  005037  001162                  CLR     $REG0           ; START VAL IS 0.
1537  013410  012701  000076                  MOV     #62.,R1         ; 31. X 8. X 1/4 = UNIT LENGTH.
1538  013414  010137  001124          3$:     MOV     R1,$GDDAT       ; SET EXPECTED END POINT.
1539  013420  004737  024524                  JSR     PC,EXECUTE      ; EXECUTE 8 VECTORS (POS,POS).
1540  013424  017737  166334  001126          MOV     @XPOS,$BDDAT
1541  013432  023737  001124  001126          CMP     $GDDAT,$BDDAT
1542  013440  001401                          BEQ     .+4
1543  013442  104011                          ERROR   11              ; X SCALE (ADDING) FAILED.
1544  013444  017737  166316  001126          MOV     @YPOS,$BDDAT
1545  013452  023737  001124  001126          CMP     $GDDAT,$BDDAT
1546  013460  001401                          BEQ     .+4
1547  013462  104011                          ERROR   11              ; Y SCALE (ADDING) FAILED.
1548
1549  013464  005037  001124                  CLR     $GDDAT          ; RESET EXPECTED X/Y.
1550  013470  004737  024552                  JSR     PC,CONT         ; EXECUTE 8 VECTORS (NEG,NEG).
1551  013474  017737  166264  001126          MOV     @XPOS,$BDDAT
1552  013502  023737  001124  001126          CMP     $GDDAT,$BDDAT
1553  013510  001401                          BEQ     .+4
1554  013512  104011                          ERROR   11              ; X SCALE (SUBTRACTING) FAILED.
1555  013514  017737  166246  001126          MOV     @YPOS,$BDDAT
1556  013522  023737  001124  001126          CMP     $GDDAT,$BDDAT
1557  013530  001401                          BEQ     .+4
1558  013532  104011                          ERROR   11              ; Y SCALE (SUBTRACTING) FAILED.
1559
1560  013534  023727  001164  000017          CMP     $TMP0,#17
1561  013542  001407                          BEQ     TST44           ;;EXIT WHEN ALL SCALES DONE
1562  013544  005237  001164                  INC     $TMP0           ; OTHERWISE, NEXT SCALE VALUE...
1563  013550  005237  032540                  INC     BUFFER+10       ;...IN DISPLAY CODE TOO.
1564  013554  062701  000076                  ADD     #62.,R1         ; NEXT EXPECTED END POINT.
1565  013560  000715                          BR      3$              ; AND CONTINUE.
1566  013562                          4$:
1567
1568                                  ;;*****************************************************************
 (3)                                  ;*TEST 44       TEST THAT EXCEEDING +X AXIS SETS EDGE INDICATOR
 (3)                                  ;;*****************************************************************
 (2)  013562  000004                  ;
 (2)  013564  012737  000044  001202  TST44:  SCOPE
1569  013572  013737  001762  001122          MOV     #44,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1570  013600  013700  001712                  MOV     $REG0,$BDADR    ;SET UP REG ADRS 02
1571  013604  012720  154024                  MOV     DBUF,R0
1572  013610  012720  116000                  MOV     #154024,(R0)+   ;SCALE OF UNITY
1573  013614  012720  001777                  MOV     #116000,(0)+    ;LOAD POINT
1574  013620  005020                          MOV     #1777,(0)+      ;LOAD MAX X
1575  013622  012720  172000                  CLR     (0)+            ;LOAD Y
                                              MOV     #172000,(R0)+   ;LOAD STOP
```

```
1576  013626  012720  110000                  MOV     #110000,(0)+    ;LOAD LONG VECTOR
1577  013632  012720  000001                  MOV     #1,(0)+         ;LOAD DETA X
1578  013636  005020                          CLR     (0)+            ;LOAD DELTA Y
1579  013640  012720  172000                  MOV     #172000,(0)+    ;LOAD STOP
1580  013644  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
1581
1582  013650  005037  001124                  CLR     $GDDAT          ;CLEAR EXPECTED
1583  013654  017737  166102  001126          MOV     @SREG0,$BDDAT   ;READ STATUS
1584  013662  032737  000040  001126          BIT     #40,$BDDAT      ;TEST BIT 5
1585  013670  001402                          BEQ     1$
1586  013672  104020                          ERROR   20              ;EDGE INDICATOR SET IN ERROR
1587  013674  000432                          BR      TST45           ;;
1588
1589
1590  013676                          1$:
 (1)  013676  004737  024552                  JSR     PC,CONT                 ;RESUME DISPLAY AND WAIT FOR STOP FLAG
1591  013702  012737  000040  001124          MOV     #BIT5,$GDDAT    ;LOAD EXPECTED
1592  013710  017737  166046  001126          MOV     @SREG0,$BDDAT   ;READ STATUS
1593  013716  032737  000040  001126          BIT     #40,$BDDAT      ;TEST BIT 5
1594  013724  001002                          BNE     2$
1595  013726  104020                          ERROR   20              ;EDGE INDICATOR FAILED TO SET
1596  013730  000414                          BR      TST45           ;;
1597
1598  013732                          2$:
 (1)  013732  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
1599  013736  005037  001124                  CLR     $GDDAT          ;CLEAR EXPECTED
1600  013742  017737  166014  001126          MOV     @SREG0,$BDDAT   ;READ STATUS
1601  013750  032737  000040  001126          BIT     #40,$BDDAT      ;TEST BIT 5
1602  013756  001401                          BEQ     TST45           ;;BR IF EQUAL
1603  013760  104020                          ERROR   20              ;EDGE INDICATOR FAILED TO CLEAR
1604
1605                            ;;*****************************************************************
 (3)                            ;*TEST 45       TEST THAT EXCEEDING -X AXIS SETS EDGE INDICATOR
 (3)                            ;;*****************************************************************
 (2)  013762  000004            TST45:  SCOPE
 (2)  013764  012737  000045  001202          MOV     #45,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1606  013772  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
1607  014000  013700  001712                  MOV     DBUF,R0
1608  014004  012720  116000                  MOV     #116000,(0)+    ;LOAD POINT
1609  014010  005020                          CLR     (0)+            ;LOAD MAX X
1610  014012  005020                          CLR     (0)+            ;LOAD Y
1611  014014  012720  172000                  MOV     #172000,(R0)+   ;LOAD STOP DISPLAY
1612  014020  012720  110000                  MOV     #110000,(0)+    ;LOAD LONG VECTOR
1613  014024  012720  020001                  MOV     #20001,(0)+     ;LOAD DETA X
1614  014030  005020                          CLR     (0)+            ;LOAD DELTA Y
1615  014032  012720  172000                  MOV     #172000,(0)+    ;LOAD STOP
1616  014036  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
1617  014042  004737  024552                  JSR     PC,CONT                 ;RESUME DISPLAY AND WAIT FOR STOP FLAG
1618
1619  014046  012737  000040  001124          MOV     #BIT5,$GDDAT    ;LOAD EXPECTED
1620  014054  017737  165702  001126          MOV     @SREG0,$BDDAT   ;READ STATUS
1621  014062  032737  000040  001126          BIT     #40,$BDDAT      ;TEST BIT 5
1622  014070  001002                          BNE     2$
1623  014072  104020                          ERROR   20              ;EDGE INDICATOR FAILED TO SET
1624  014074  000414                          BR      TST46           ;;
1625
```

```
1626  014076                          2$:
 (1)  014076  004737  024524                JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
1627  014102  005037  001124                CLR     $GDDAT          ;CLEAR EXPECTED
1628  014106  017737  165650  001126        MOV     @SREG0,$BDDAT   ;READ STATUS
1629  014114  032737  000040  001126        BIT     #40,$BDDAT      ;TEST BIT 5
1630  014122  001401                        BEQ     TST46           ;;BR IF EQUAL
1631  014124  104020                        ERROR   20              ;EDGE INDICATOR FAILED TO CLEAR
1632
1633                                  ;;*****************************************************************
 (3)                                  ;*TEST 46       TEST THAT EXCEEDING +Y AXIS SETS EDGE INDICATOR
 (3)                                  ;;*****************************************************************
 (2)  014126  000004                  TST46:  SCOPE
 (2)  014130  012737  000046  001202          MOV     #46,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1634  014136  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
1635  014144  013700  001712                MOV     DBUF,R0
1636  014150  012720  116000                MOV     #116000,(0)+    ;LOAD POINT
1637  014154  005020                        CLR     (0)+            ;LOAD X
1638  014156  012720  001777                MOV     #1777,(0)+      ;LOAD MAX Y
1639  014162  012720  110000                MOV     #110000,(0)+    ;LOAD LONG VECTOR
1640  014166  005020                        CLR     (0)+            ;LOAD DETA X
1641  014170  012720  000001                MOV     #1,(0)+         ;LOAD DELTA Y
1642  014174  012720  172000                MOV     #172000,(R0)+   ;LOAD STOP
1643  014200  004737  024524                JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
1644
1645  014204  012737  000040  001124        MOV     #BIT5,$GDDAT    ;LOAD EXPECTED
1646  014212  017737  165544  001126        MOV     @SREG0,$BDDAT   ;READ STATUS
1647  014220  033737  001124  001126        BIT     $GDDAT,$BDDAT   ;TEST BIT 5
1648  014226  001001                        BNE     TST47           ;;BR IF SET
1649  014230  104020                        ERROR   20              ;EDGE FLAG FAILED TO SET
1650
1651
1652                                  ;;*****************************************************************
 (3)                                  ;*TEST 47       TEST THAT EXCEEDING -Y AXIS SETS EDGE INDICATOR
 (3)                                  ;;*****************************************************************
 (2)  014232  000004                  TST47:  SCOPE
 (1)  014234  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
 (2)  014242  012737  000047  001202          MOV     #47,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1653  014250  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
1654  014256  013700  001712                MOV     DBUF,R0
1655  014262  012720  116000                MOV     #116000,(0)+    ;LOAD POINT
1656  014266  005020                        CLR     (0)+            ;LOAD X
1657  014270  005020                        CLR     (0)+            ;LOAD Y
1658  014272  012720  110000                MOV     #110000,(0)+    ;LOAD LONG VECTOR
1659  014276  005020                        CLR     (0)+            ;LOAD DETA X
1660  014300  012720  020001                MOV     #20001,(0)+     ;LOAD DELTA Y
1661  014304  012720  172000                MOV     #172000,(0)+    ;LOAD STOP
1662  014310  004737  024524                JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
1663
1664  014314  012737  000040  001124        MOV     #BIT5,$GDDAT    ;LOAD EXPECTED
1665  014322  017737  165434  001126        MOV     @SREG0,$BDDAT   ;READ STATUS
1666  014330  032737  000040  001126        BIT     #40,$BDDAT      ;TEST BIT 5
1667  014336  001001                        BNE     1$              ;BR IF SET
1668  014340  104020                        ERROR   20              ;EDGE FLAG FAILED TO SET
1669  014342  000005                  1$:   RESET                   ;CLR EDGE INDICATOR
1670  014344  005037  001124                CLR     $GDDAT          ;EXPECT ZERO
1671  014350  017737  165406  001126        MOV     @SREG0,$BDDAT   ;READ REG 02
```

```
                                                BIC     #177737,$BDDAT  ;CLR JUNK
                                                BEQ     TST50           ;;NEXT TEST IF CLRED
                                                ERROR   32              ;RESET FAILED TO CLEAR EDGE INDICATOR


                                        ;;**********************************************************************
                                        ;*TEST 50       TEST EDGE INDICATOR & EDGE INTR GOING-ON TO OFF, OFF TO OFF & OFF TO ON
                                        ;;**********************************************************************
                                        TST50:  SCOPE
                                                MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                                                MOV     #50,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                                ;DO ABOVE TEST ON ALL FOUR SCREEN EDGES USING LONG VECTOR
                                                ;SEQUENCE AS FOLLOWS - VECTOR OFF SCREEN DELTA 1377 FROM
                                                ;POINT 1000/1000, VECTOR THRU CORNER OF SCREEN DELTA 1377
                                                ;TO EQUIVILENT POINT IN ADJACENT AXIS, THEN VECTOR ON SCREEN
                                                ;DELTA 1377 TO PIONT OF ORIGIN 1000/1000
                                                MOV     #340,PSW        ;HIGHEST PRIORITY
                                                MOV     #2$,@LPVCT      ;INTR RETURN ADRS
                                                MOV     #340,@LPVCT1    ;HIGHEST PRIORITY ON INTR
                                                MOV     #BUFFER,R0      ;STARTING ADRS OF INSTRS
                                                MOV     #176060,(R0)+   ;LD STATUS B' INSTR (ENAB EDGE FLAG INTR)
                                                MOV     #116000,(R0)+   ;POINT INSTR
                                                MOV     #1000,(R0)+     ;X=1000
                                                MOV     #1000,(R0)+     ;Y=1000
                                                MOV     #112000,(R0)+   ;LONG VECTOR INSTR
                                                MOV     #1377,(R0)+     ;X
                                                CLR     (R0)+   ;Y
                                                MOV     #21377,(R0)+    ;-X
                                                MOV     #1377,(R0)+     ;Y
                                                CLR     (R0)+   ;X
                                                MOV     #21377,(R0)+    ;-Y
                                                CLR     (R0)+   ;X
                                                MOV     #1377,(R0)+     ;Y
                                                MOV     #21377,(R0)+    ;-X
                                                MOV     #21377,(R0)+    ;-Y
                                                MOV     #1377,(R0)+     ;X
                                                CLR     (R0)+   ;Y
                                                MOV     #21377,(R0)+    ;-X
                                                CLR     (R0)+   ;Y
                                                MOV     #1377,(R0)+     ;X
                                                MOV     #21377,(R0)+    ;-Y
                                                CLR     (R0)+   ;X
                                                MOV     #1377,(R0)+     ;Y
                                                CLR     (R0)+   ;X
                                                MOV     #21377,(R0)+    ;-Y
                                                MOV     #1377,(R0)+     ;X
                                                MOV     #1377,(R0)+     ;Y
                                                MOV     #21377,(R0)+    ;-X
                                                CLR     (R0)+   ;Y
                                                MOV     #172000,(R0)    ;STOP
                                                MOV     #TBL4R,R0       ;ADRS OF EXPECTED X & Y POS VALUES
                                                CLR     R2              ;CLR SOFTWARE TIMER
                                                MOV     #163636,R4      ;THE EXPECTED EDGE INDICATOR
                                                MOV     #171,R3         ;CONTROLLED BY R3 & R4
                                                MOV     #BUFFER,@DPC    ;START
                                                CLR     PSW             ;ALLOW INT TO OCCUR
```

```
1723  014634  005302                    1$:     DEC     R2              ;THIS COUNTER SHOULD NEVER UNDERFLOW
1724  014636  001376                            BNE     1$              ;WAIT FOR INTR
1725  014640  013737  001762  001122            MOV     SREG0,$BDADR    ;SET UP REG 02 ADRS
1726  014646  012737  000004  001124            MOV     #4,$GDDAT       ;EXPECT EDGE FLAG
1727  014654  017737  165102  001126            MOV     @SREG0,$BDDAT   ;READ REG 02
1728  014662  042737  177773  001126            BIC     #177773,$BDDAT  ;SAVE ONLY EDGE FLAG
1729  014670  104033                            ERROR   33              ;EDGE FLAG FAILED TO INTERRUPT
1730  014672  000545                            BR      11$             ;LOOK FOR LOOP ON TEST
1731  014674  022626                    2$:     CMP     (SP)+,(SP)+     ;RETURN HERE ON INTR - FIX STACK SINCE NO RTI
1732  014676  013737  001762  001122            MOV     SREG0,$BDADR    ;SET REG ADRS 02
1733  014704  012737  000004  001124            MOV     #4,$GDDAT       ;EXPECT EDGE FLAG
1734  014712  000241                            CLC                     ;ZERO CARRY
1735  014714  006003                            ROR     R3              ;SHIFT HI WORD
1736  014716  006004                            ROR     R4              ;SHIFT LO WORD
1737  014720  103003                            BCC     3$              ;BR IF NOT EXPECTING EDGE INDICATOR
1738  014722  052737  000040  001124            BIS     #40,$GDDAT      ;EXPECT EDGE INDICATOR ALSO
1739  014730  017737  165026  001126    3$:     MOV     @SREG0,$BDDAT   ;READ REG 02
1740  014736  042737  177733  001126            BIC     #177733,$BDDAT  ;SAVE EDGE BITS ONLY
1741  014744  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
1742  014752  001402                            BEQ     4$              ;BR IF SO
1743  014754  104033                            ERROR   33              ;EDGE FLAG OR INDICATOR FAILED TO SET ON INTERRUPT
1744  014756  000513                            BR      11$             ;LOOK FOR LOOP ON TEST
1745  014760  012037  001124            4$:     MOV     (R0)+,$GDDAT    ;LD EXPECTED X POS
1746  014764  013737  001764  001122            MOV     XPOS,$BDADR     ;SET UP REG 04 ADRS
1747  014772  017737  164766  001126            MOV     @XPOS,$BDDAT    ;READ X POS
1748  015000  017705  164770                    MOV     @XDOFF,R5       ;GET HI X POS
1749  015004  004737  025136                    JSR     PC,POCONV       ;MAKE SURE NOT GREATER THAN 10 BITS
1750  015010  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
1751  015016  001402                            BEQ     5$              ;BR IF SO
1752  015020  104033                            ERROR   33              ;X POS INCORRECT ON INTR
1753  015022  000471                            BR      11$             ;LOOK FOR LOOP ON TEST
1754  015024  012037  001124            5$:     MOV     (R0)+,$GDDAT    ;LD EXPECTED Y POS
1755  015030  013737  001766  001122            MOV     YPOS,$BDADR     ;SET UP REG 06 ADRS
1756  015036  017737  164724  001126            MOV     @YPOS,$BDDAT    ;READ Y POS
1757  015044  017705  164726                    MOV     @YDOFF,R5       ;GET HI Y POS
1758  015050  004737  025136                    JSR     PC,POCONV       ;MAKE SURE NOT GREATER THAN 02 BITS
1759  015054  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
1760  015062  001402                            BEQ     6$              ;BR IF SO
1761  015064  104033                            ERROR   33              ;Y POS INCORRECT ON INTR
1762  015066  000447                            BR      11$             ;LOOK FOR LOOP ON TEST
1763  015070  005710                    6$:     TST     (R0)            ;HAVE 24 INTRS BEEN DONE? (18 OFF-6 ON SCREEN)
1764  015072  100405                            BMI     7$              ;BR IF SO
1765  015074  005037  177776                    CLR     PSW             ;ALLOW INTR ON RESUME
1766  015100  005277  164654                    INC     @DPC            ;RESUME
1767  015104  000653                            BR      1$              ;RETURN TO COUNTER
1768  015106  042737  000020  032530    7$:     BIC     #20,BUFFER      ;SET UP FOR NO EDGE FLAG INTR ENAB
1769  015114  012777  015150  164702            MOV     #9$,@LPVCT      ;SET UP INTR RETURN ADRS - NONE EXPECTED
1770  015122  013737  001762  001122            MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
1771  015130  012777  032530  164622            MOV     #BUFFER,@DPC    ;START
1772  015136  012702  001000                    MOV     #1000,R2        ;SET COUNTER
1773  015142  005302                    8$:     DEC     R2              ;COUNT
1774  015144  001376                            BNE     8$              ;ALLOW TIME TO COMPLETE
1775  015146  000401                            BR      10$             ;SKIP OVER INTR RETURN
1776  015150  022626                    9$:     CMP     (SP)+,(SP)+     ;FIX STACK IF UNEXPECTED INTR OCCURED
1777  015152  012737  100000  001124    10$:    MOV     #100000,$GDDAT  ;EXPECT STOP ONLY
1778  015160  017737  164576  001126            MOV     @SREG0,$BDDAT   ;READ REG 02
```

```
1779  015166  042737  077773  001126          BIC     #77773,$BDDAT   ;SAVE STOP + EDGE FLAG ONLY
1780  015174  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1781  015202  001401                          BEQ     11$             ;BR IF SO
1782  015204  104045                          ERROR   45              ;UNEXPECTED EDGE INTR OR EDGE FAILED TO CLR
1783  015206  004737  024634          11$:    JSR     PC,RSTVEC       ;RESTORE VECTOR
1784  015212  000461                          BR      TST51           ;;TO NEXT TEST
1785
1786  015214  001000          TBL4R:  1000    ;TABLE CONTAINS EXPCTED X & Y POS RESULTS ON EA. EDGE INTR
1787  015216  001000                  1000
1788  015220  001777                  1777
1789  015222  001000                  1000
1790  015224  001777                  1777
1791  015226  001400                  1400
1792  015230  001400                  1400
1793  015232  001777                  1777
1794  015234  001000                  1000
1795  015236  001777                  1777
1796  015240  001000                  1000
1797  015242  001000                  1000
1798  015244  001000                  1000
1799  015246  001000                  1000
1800  015250  001000                  1000
1801  015252  001777                  1777
1802  015254  000400                  400
1803  015256  001777                  1777
1804  015260  000000                  0
1805  015262  001377                  1377
1806  015264  000000                  0
1807  015266  001000                  1000
1808  015270  001000                  1000
1809  015272  001000                  1000
1810  015274  001000                  1000
1811  015276  001000                  1000
1812  015300  000000                  0
1813  015302  001000                  1000
1814  015304  000000                  0
1815  015306  000401                  401
1816  015310  000401                  401
1817  015312  000000                  0
1818  015314  001000                  1000
1819  015316  000000                  0
1820  015320  001000                  1000
1821  015322  001000                  1000
1822  015324  001000                  1000
1823  015326  001000                  1000
1824  015330  001000                  1000
1825  015332  000000                  0
1826  015334  001377                  1377
1827  015336  000000                  0
1828  015340  001777                  1777
1829  015342  000400                  400
1830  015344  001777                  1777
1831  015346  001000                  1000
1832  015350  001000                  1000
1833  015352  001000                  1000
1834  015354  100000                  100000          ;TERMINATOR
```

```
1835
1836                                    ;;*****************************************************************
 (3)                                    ;*TEST 51       LOAD CHARACTER REGISTER WITH #0 THRU #77
 (3)                                    ;;*****************************************************************
 (2) 015356 000004                      TST51:  SCOPE
 (1) 015360 012737 000100 001166                MOV     #100,$TIMES     ;;DO 100 ITERATIONS
 (2) 015366 012737 000051 001202                MOV     #51,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1837 015374 013737 001766 001122                MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
1838 015402 012737 015434 001110                MOV     #1$,$LPERR      ;LOAD RETURN
1839 015410 012777 100000 164274                MOV     #100000,@DBUF   ;LOAD ''CHARACTER MODE''
1840 015416 005037 001124                       CLR     $GDDAT          ;LOAD ''NULL'' CHARACTER
1841 015422 005037 001726                       CLR     DSAVE           ;CLEAR EXPECT CHARACTER
1842 015426 012737 172000 032534                MOV     #172000,BUFFER+4        ;LOAD DISPLAY STOP
1843 015434 013777 001726 164252 1$:            MOV     DSAVE,@DBUF1    ;LOAD BUFFER
1844 015442 004737 024524                       JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
1845
1846 015446 017737 164314 001126                MOV     @YPOS,$BDDAT    ;READ CHARACTER REG.
1847 015454 042737 001777 001126                BIC     #1777,$BDDAT    ;MASK TO BITS 10-15
1848 015462 023737 001124 001126                CMP     $GDDAT,$BDDAT           ;COMPARE EXPCT TO RCVD
1849 015470 001402                              BEQ     2$              ;;BR IF EQUAL
1850 015472 104021                              ERROR   21              ;ERROR, CHARACTER REGISTER LOADED IN ERROR
1851 015474 000415                              BR      TST52           ;;BR UPON ERROR
1852
1853 015476 062737 002000 001124 2$:            ADD     #2000,$GDDAT    ;UPDATE EXPECTED VALUE
1854 015504 005237 001726                       INC     DSAVE           ;UPDATE CHARACTER VALUE
1855 015510 022737 000016 001726                CMP     #16,DSAVE       ;TEST FOR ''SHIFT-OUT''
1856 015516 001767                              BEQ     2$              ;BR BACK AND GET ANOTHER CHAR
1857 015520 022737 000100 001726                CMP     #100,DSAVE      ;TEST FOR LAST VALID CHAR <6 BITS>
1858 015526 001342                              BNE     1$              ;BR BACK IF NOT DONE
1859
1860                                    ;;*****************************************************************
 (3)                                    ;*TEST 52       TEST THAT 'NULL'' DOES NOT CHANGE X OR Y AXIS
 (3)                                    ;;*****************************************************************
 (2) 015530 000004                      TST52:  SCOPE
 (2) 015532 012737 000052 001202                MOV     #52,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1861 015540 012777 116000 164144                MOV     #116000,@DBUF   ;POINT MODE
1862 015546 012777 001000 164140                MOV     #1000,@DBUF1    ;LOAD BUFFER
1863 015554 012777 001000 164134                MOV     #1000,@DBUF2    ;1000,1000
1864 015562 012777 100000 164130                MOV     #100000,@DBUF3  ;LOAD ''CHARACTER MODE''
1865 015570 005077 164126                       CLR     @DBUF4          ;NULL CHARACTER
1866 015574 012777 172000 164122                MOV     #172000,@DBUF5  ;LOAD DISPLAY STOP
1867 015602 004737 024524                       JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
1868
1869 015606 017737 164152 001126 1$:            MOV     @XPOS,$BDDAT    ;READ X AXIS
1870 015614 012737 001000 001124                MOV     #1000,$GDDAT    ;LOAD EXPECTED
1871 015622 023737 001124 001126                CMP     $GDDAT,$BDDAT           ;COMPARE EXPCT TO RCVD
1872 015630 001405                              BEQ     2$              ;YES
1873 015632 013737 001764 001122                MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
1874 015640 104022                              ERROR   22              ;''NULL'' CHARACTER CHANGED X AXIS
1875 015642 000416                              BR      TST53           ;;
1876
1877 015644 017737 164116 001126 2$:            MOV     @YPOS,$BDDAT    ;READ Y AXIS
1878 015652 042737 176000 001126                BIC     #176000,$BDDAT  ;MASK TO BITS 0-9
1879 015660 023737 001124 001126                CMP     $GDDAT,$BDDAT           ;COMPARE EXPCT TO RCVD
1880 015666 001404                              BEQ     TST53           ;;BR IF EQUAL
1881 015670 013737 001766 001122                MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
```

```
                        ERROR   22              ;'NULL' CHARACTER CHANGED Y AXIS

                ;;*****************************************************************
                ;*TEST 53        CHAR SCALE 1/2 - NOT ROTATED - TEST DELTA X
                ;;*****************************************************************
                TST53:  SCOPE
                        MOV     #53,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                        MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
                        MOV     #155200,(R0)+           ;LOAD 'STATUS C NO ROT. & CHAR SCALE 0'
                        MOV     #116000,(R0)+           ;LOAD 'POINT MODE'
                        MOV     #1000,(R0)+             ;X = 1000
                        MOV     #1000,(R0)+             ;Y = 1000
                        MOV     #100000,(R0)+           ;CHAR MODE
                        MOV     #40,(R0)+               ; 'SPACE' & 'NULL'
                        MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
                        JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG

                        MOV     @XPOS,$BDDAT            ;READ X POSITION
                        BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
                        MOV     #1000,$GDDAT            ;LOAD BASE
                        ADD     CHSZ0,$GDDAT            ;LOAD EXPECTED DATA
                        CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
                        BEQ     1$              ;;BR IF DELTA X IS CORRECT
                        MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
                        ERROR   23              ;INCORRECT DELTA X WITH CHAR SCALE OF 1/2
                1$:     MOV     @YPOS,$BDDAT            ;READ Y AXIS
                        BIC     #176000,$BDDAT          ;MASK TOP BITS
                        MOV     #1000,$GDDAT            ;LOAD EXPECTED
                        CMP     $GDDAT,$BDDAT           ;TEST IF EQUAL
                        BEQ     TST54           ;;BR IF SAME
                        MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
                        ERROR   22              ;Y AXIS CHANGED IN ERROR

                ;;*****************************************************************
                ;*TEST 54        CHAR SCALE 1 - NOT ROTATED - TEST DELTA X
                ;;*****************************************************************
                TST54:  SCOPE
                        MOV     #54,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                        MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
                        MOV     #155240,(R0)+           ;LOAD 'STATUS C NO ROT. & CHAR SCALE 1'
                        MOV     #116000,(R0)+           ;LOAD 'POINT MODE'
                        MOV     #1000,(R0)+             ;X = 1000
                        MOV     #1000,(R0)+             ;Y = 1000
                        MOV     #100000,(R0)+           ;CHAR MODE
                        MOV     #40,(R0)+               ; 'SPACE' & 'NULL'
                        MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
                        JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG

                        MOV     @XPOS,$BDDAT            ;READ X POSITION
                        BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
                        MOV     #1000,$GDDAT            ;LOAD BASE
                        ADD     CHSZ1,$GDDAT            ;LOAD EXPECTED DATA
                        CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
                        BEQ     TST55           ;;BR IF DELTA X IS CORRECT
                        MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
                        ERROR   23              ;INCORRECT DELTA X WITH CHAR SCALE OF 1
```

```
                                ;;*****************************************************************
                                ;*TEST 55        CHAR SCALE 1 1/2 - NOT ROTATED - TEST DELTA X
                                ;;*****************************************************************
                                TST55:  SCOPE
016212  000004                  
016214  012737  000055  001202          MOV     #55,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
016222  012700  032530                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
016226  012720  155300                  MOV     #155300,(R0)+           ;LOAD 'STATUS C NO ROT. & CHAR SCALE 1 1/2
016232  012720  116000                  MOV     #116000,(R0)+           ;LOAD 'POINT MODE'
016236  012720  001000                  MOV     #1000.(R0)+             ;X = 1000
016242  012720  001000                  MOV     #1000.(R0)+             ;Y = 1000
016246  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
016252  012720  000040                  MOV     #40,(R0)+               ; 'SPACE' & 'NULL'
016256  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
016262  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG

016266  017737  163472  001126          MOV     @XPOS,$BDDAT            ;READ X POSITION
016274  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
016302  012737  001000  001124          MOV     #1000,$GDDAT    ;LOAD BASE
016310  063737  001744  001124          ADD     CHSZ2,$GDDAT            ;LOAD EXPECTED DATA
016316  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
016324  001404                          BEQ     TST56           ;;BR IF DELTA X IS CORRECT
016326  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
016334  104023                          ERROR   23                      ;INCORRECT DELTA X WITH CHAR SCALE OF 1 1/2

                                ;;*****************************************************************
                                ;*TEST 56        CHAR SCALE 2 - NOT ROTATED - TEST DELTA X
                                ;;*****************************************************************
                                TST56:  SCOPE
016336  000004                  
016340  012737  000056  001202          MOV     #56,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
016346  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
016354  012700  032530                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
016360  012720  155340                  MOV     #155340,(R0)+           ;LOAD 'STATUS C NO ROT. & CHAR SCALE 2'
016364  012720  116000                  MOV     #116000,(R0)+           ;LOAD 'POINT MODE'
016370  012720  001000                  MOV     #1000.(R0)+             ;X = 1000
016374  012720  001000                  MOV     #1000.(R0)+             ;Y = 1000
016400  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
016404  012720  000040                  MOV     #40,(R0)+               ; 'SPACE' & 'NULL'
016410  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
016414  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG

016420  017737  163340  001126          MOV     @XPOS,$BDDAT            ;READ X POSITION
016426  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
016434  012737  001000  001124          MOV     #1000,$GDDAT            ;LOAD BASE
016442  063737  001746  001124          ADD     CHSZ3,$GDDAT            ;LOAD EXPECTED DATA
016450  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
016456  001401                          BEQ     TST57           ;;BR IF DELTA X IS CORRECT
016460  104023                          ERROR   23                      ;INCORRECT DELTA X WITH CHAR SCALE OF 2

                                ;;*****************************************************************
                                ;*TEST 57        CHAR SCALE 1/2 - NOT ROTATED - TEST DELTA Y
                                ;;*****************************************************************
                                TST57:  SCOPE
016462  000004                  
016464  012737  000057  001202          MOV     #57,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
016472  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
016500  012700  032530                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
```

```
 1974  016504  012720  155200                  MOV     #155200,(R0)+           ;LOAD 'STATUS C NO ROT. & CHAR SCALE 1/2'
 1975  016510  012720  116000                  MOV     #116000,(R0)+           ;LOAD 'POINT MODE'
 1976  016514  012720  001000                  MOV     #1000.,(R0)+            ;X = 1000
 1977  016520  012720  001000                  MOV     #1000.,(R0)+            ;Y = 1000
 1978  016524  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
 1979  016530  012720  000012                  MOV     #12,(R0)+               ; 'LF' & 'NULL'
 1980  016534  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
 1981  016540  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 1982
 1983  016544  017737  163216  001126          MOV     @YPOS,$BDDAT            ;READ Y POSITION
 1984  016552  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
 1985  016560  012737  001000  001124          MOV     #1000.,$GDDAT           ;LOAD BASE
 1986  016566  163737  001750  001124          SUB     LFSZ0,$GDDAT            ;LOAD EXPECTED DATA
 1987  016574  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
 1988  016602  001401                          BEQ     TST60           ;;BR IF DELTA Y IS CORRECT
 1989  016604  104024                          ERROR   24                      ;INCORRECT DELTA Y WITH CHAR SCALE OF 1/2
 1990
 1991                                  ;;**************************************************************
  (3)                                  ;*TEST 60        CHAR SCALE 1 - NOT ROTATED - TEST DELTA Y
  (3)                                  ;;**************************************************************
  (2)  016606  000004                  TST60:  SCOPE
  (2)  016610  012737  000060  001202          MOV     #60.,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 1992  016616  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 1993  016624  012700  032530                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
 1994  016630  012720  155240                  MOV     #155240,(R0)+           ;LOAD 'STATUS C NO ROT. & CHAR SCALE 1'
 1995  016634  012720  116000                  MOV     #116000,(R0)+           ;LOAD 'POINT MODE'
 1996  016640  012720  001000                  MOV     #1000.,(R0)+            ;X = 1000
 1997  016644  012720  001000                  MOV     #1000.,(R0)+            ;Y = 1000
 1998  016650  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
 1999  016654  012720  000012                  MOV     #12,(R0)+               ; 'LF' & 'NULL'
 2000  016660  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
 2001  016664  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 2002
 2003  016670  017737  163072  001126          MOV     @YPOS,$BDDAT            ;READ Y POSITION
 2004  016676  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
 2005  016704  012737  001000  001124          MOV     #1000.,$GDDAT           ;LOAD BASE
 2006  016712  163737  001752  001124          SUB     LFSZ1,$GDDAT            ;LOAD EXPECTED DATA
 2007  016720  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
 2008  016726  001401                          BEQ     TST61           ;;BR IF DELTA Y IS CORRECT
 2009  016730  104024                          ERROR   24                      ;INCORRECT DELTA Y WITH CHAR SCALE OF 1
 2010
 2011                                  ;;**************************************************************
  (3)                                  ;*TEST 61        CHAR SCALE 1 1/2 - NOT ROTATED - TEST DELTA Y
  (3)                                  ;;**************************************************************
  (2)  016732  000004                  TST61:  SCOPE
  (2)  016734  012737  000061  001202          MOV     #61.,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 2012  016742  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 2013  016750  012700  032530                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
 2014  016754  012720  155300                  MOV     #155300,(R0)+           ;LOAD 'STATUS C NO ROT. & CHAR SCALE 1 1/2'
 2015  016760  012720  116000                  MOV     #116000,(R0)+           ;LOAD 'POINT MODE'
 2016  016764  012720  001000                  MOV     #1000.,(R0)+            ;X = 1000
 2017  016770  012720  001000                  MOV     #1000.,(R0)+            ;Y = 1000
 2018  016774  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
 2019  017000  012720  000012                  MOV     #12,(R0)+               ; 'LF' & 'NULL'
 2020  017004  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
 2021  017010  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
```

```
2022
2023  017014  017737  162746  001126          MOV     @YPOS,$BDDAT            ;READ Y POSITION
2024  017022  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
2025  017030  012737  001000  001124          MOV     #1000,$GDDAT            ;LOAD EXPECTED
2026  017036  163737  001754  001124          SUB     LFSZ2,$GDDAT            ;LOAD EXPECTED DATA
2027  017044  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
2028  017052  001401                          BEQ     TST62           ;;BR IF DELTA Y IS CORRECT
2029  017054  104024                          ERROR   24                      ;INCORRECT DELTA Y WITH CHAR SCALE OF 1 1//2
2030
2031                                  ;;*****************************************************************
 (3)                                  ;*TEST 62       CHAR SCALE 2 - NOT ROTATED - TEST DELTA Y
 (3)                                  ;;*****************************************************************
 (2)  017056  000004                  TST62:  SCOPE
 (2)  017060  012737  000062  001202          MOV     #62,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2032  017066  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
2033  017074  012700  032530                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
2034  017100  012720  155340                  MOV     #155340,(R0)+           ;LOAD 'STATUS C NO ROT. & CHAR SCALE 2''
2035  017104  012720  116000                  MOV     #116000,(R0)+           ;LOAD 'POINT MODE''
2036  017110  012720  001000                  MOV     #1000,(R0)+             ;X = 1000
2037  017114  012720  001000                  MOV     #1000,(R0)+             ;Y = 1000
2038  017120  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
2039  017124  012720  000012                  MOV     #12,(R0)+               ; 'LF' & 'NULL''
2040  017130  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
2041  017134  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
2042
2043  017140  017737  162622  001126          MOV     @YPOS,$BDDAT            ;READ Y POSITION
2044  017146  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
2045  017154  012737  001000  001124          MOV     #1000,$GDDAT            ;LOAD EXPECTED
2046  017162  163737  001756  001124          SUB     LFSZ3,$GDDAT            ;LOAD EXPECTED DATA
2047  017170  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
2048  017176  001401                          BEQ     TST63           ;;BR IF DELTA Y IS CORRECT
2049  017200  104024                          ERROR   24                      ;INCORRECT DELTA Y WITH CHAR SCALE OF 2
2050
2051                                  ;;*****************************************************************
 (3)                                  ;*TEST 63       CHAR SCALE 1/2 - ROTATED - TEST DELTA Y
 (3)                                  ;;*****************************************************************
 (2)  017202  000004                  TST63:  SCOPE
 (2)  017204  012737  000063  001202          MOV     #63,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2052  017212  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
2053  017220  012700  032530                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
2054  017224  012720  155600                  MOV     #155600,(R0)+           ;LOAD 'STATUS C ROT. & CHAR SCALE 1/2''
2055  017230  012720  116000                  MOV     #116000,(R0)+           ;LOAD 'POINT MODE''
2056  017234  012720  001000                  MOV     #1000,(R0)+             ;X = 1000
2057  017240  012720  001000                  MOV     #1000,(R0)+             ;Y = 1000
2058  017244  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
2059  017250  012720  000040                  MOV     #40,(R0)+               ; 'SPACE' & 'NULL''
2060  017254  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
2061  017260  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
2062
2063  017264  017737  162476  001126          MOV     @YPOS,$BDDAT            ;READ Y POSITION
2064  017272  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
2065  017300  012737  001000  001124          MOV     #1000,$GDDAT            ;LOAD BASE
2066  017306  063737  001740  001124          ADD     CHSZ0,$GDDAT            ;LOAD EXPECTED DATA
2067  017314  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
2068  017322  001401                          BEQ     1$              ;;BR IF DELTA Y IS CORRECT
2069  017324  104025                          ERROR   25                      ;INCORRECT DELTA Y WITH CHAR SCALE OF 1/2
```

```
 2070  017326  017737  162432  001126  1$:     MOV     @XPOS,$BDDAT            ;READ X AXIS
 2071  017334  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK TOP BITS
 2072  017342  012737  001000  001124          MOV     #1000,$GDDAT            ;LOAD EXPECTED
 2073  017350  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST IF EQUAL
 2074  017356  001401                          BEQ     TST64           ;;BR IF SAME
 2075  017360  104022                          ERROR   22                      ;X AXIS CHANGED IN ERROR
 2076
 2077                                  ;;**********************************************************************
   (3)                                 ;*TEST 64       CHAR SCALE 1 - ROTATED - TEST DELTA Y
   (3)                                 ;;**********************************************************************
   (2)  017362  000004                  TST64:  SCOPE
   (2)  017364  012737  000064  001202          MOV     #64,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2078  017372  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 2079  017400  012700  032530                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
 2080  017404  012720  155640                  MOV     #155640,(R0)+           ;LOAD ''STATUS C ROT. & CHAR SCALE 1''
 2081  017410  012720  116000                  MOV     #116000,(R0)+           ;LOAD 'POINT MODE''
 2082  017414  012720  001000                  MOV     #1000,(R0)+             ;X = 1000
 2083  017420  012720  001000                  MOV     #1000,(R0)+             ;Y = 1000
 2084  017424  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
 2085  017430  012720  000040                  MOV     #40,(R0)+               ; ''SPACE'' & 'NULL''
 2086  017434  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
 2087  017440  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 2088
 2089  017444  017737  162316  001126          MOV     @YPOS,$BDDAT            ;READ Y POSITION
 2090  017452  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
 2091  017460  012737  001000  001124          MOV     #1000,$GDDAT            ;LOAD BASE
 2092  017466  063737  001742  001124          ADD     CHSZ1,$GDDAT            ;LOAD EXPECTED DATA
 2093  017474  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
 2094  017502  001401                          BEQ     TST65           ;;BR IF DELTA Y IS CORRECT
 2095  017504  104025                          ERROR   25                      ;INCORRECT DELTA Y WITH CHAR SCALE OF 1
 2096
 2097                                  ;;**********************************************************************
   (3)                                 ;*TEST 65       CHAR SCALE 1 1/2 - ROTATED - TEST DELTA Y
   (3)                                 ;;**********************************************************************
   (2)  017506  000004                  TST65:  SCOPE
   (2)  017510  012737  000065  001202          MOV     #65,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2098  017516  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 2099  017524  012700  032530                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
 2100  017530  012720  155700                  MOV     #155700,(R0)+           ;LOAD ''STATUS C ROT. & CHAR SCALE 1 1/2
 2101  017534  012720  116000                  MOV     #116000,(R0)+           ;LOAD 'POINT MODE''
 2102  017540  012720  001000                  MOV     #1000,(R0)+             ;X = 1000
 2103  017544  012720  001000                  MOV     #1000,(R0)+             ;Y = 1000
 2104  017550  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
 2105  017554  012720  000040                  MOV     #40,(R0)+               ; ''SPACE'' & 'NULL''
 2106  017560  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
 2107  017564  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 2108
 2109  017570  017737  162172  001126          MOV     @YPOS,$BDDAT            ;READ Y POSITION
 2110  017576  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
 2111  017604  012737  001000  001124          MOV     #1000,$GDDAT            ;LOAD BASE DATA
 2112  017612  063737  001744  001124          ADD     CHSZ2,$GDDAT            ;LOAD EXPECTED DATA
 2113  017620  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
 2114  017626  001401                          BEQ     TST66           ;;BR IF DELTA Y IS CORRECT
 2115  017630  104025                          ERROR   25                      ;INCORRECT DELTA Y WITH CHAR SCALE OF 1 1/2
 2116
 2117                                  ;;**********************************************************************
```

```
  (3)                                   ;*TEST 66       CHAR SCALE 2 - ROTATED - TEST DELTA Y
  (3)                                   ;;*****************************************************************
  (2)  017632  000004                   TST66:  SCOPE
  (2)  017634  012737  000066  001202            MOV     #66,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2118   017642  013737  001766  001122            MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
2119   017650  012700  032530                    MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
2120   017654  012720  155740                    MOV     #155740,(R0)+           ;LOAD ''STATUS C ROT. & CHAR SCALE 2''
2121   017660  012720  116000                    MOV     #116000,(R0)+           ;LOAD 'POINT MODE''
2122   017664  012720  001000                    MOV     #1000,(R0)+             ;X = 1000
2123   017670  012720  001000                    MOV     #1000,(R0)+             ;Y = 1000
2124   017674  012720  100000                    MOV     #100000,(R0)+           ;CHAR MODE
2125   017700  012720  000040                    MOV     #40,(R0)+               ; ''SPACE'' & 'NULL''
2126   017704  012720  172000                    MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
2127   017710  004737  024524                    JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
2128
2129   017714  017737  162046  001126            MOV     @YPOS,$BDDAT            ;READ Y POSITION
2130   017722  042737  176000  001126            BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
2131   017730  012737  001000  001124            MOV     #1000,$GDDAT            ;LOAD BASE
2132   017736  063737  001746  001124            ADD     CHSZ3,$GDDAT            ;LOAD EXPECTED DATA
2133   017744  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
2134   017752  001401                            BEQ     TST67           ;;BR IF DELTA Y IS CORRECT
2135   017754  104025                            ERROR   25              ;INCORRECT DELTA Y WITH CHAR SCALE OF 2
2136
2137                                   ;;*****************************************************************
  (3)                                   ;*TEST 67       CHAR SCALE 1/2 - ROTATED - TEST DELTA X
  (3)                                   ;;*****************************************************************
  (2)  017756  000004                   TST67:  SCOPE
  (2)  017760  012737  000067  001202            MOV     #67,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2138   017766  013737  001764  001122            MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
2139   017774  012700  032530                    MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
2140   020000  012720  155600                    MOV     #155600,(R0)+           ;LOAD ''STATUS C ROT. & CHAR SCALE 1/2''
2141   020004  012720  116000                    MOV     #116000,(R0)+           ;LOAD 'POINT MODE''
2142   020010  012720  001000                    MOV     #1000,(R0)+             ;X = 1000
2143   020014  012720  001000                    MOV     #1000,(R0)+             ;Y = 1000
2144   020020  012720  100000                    MOV     #100000,(R0)+           ;CHAR MODE
2145   020024  012720  000012                    MOV     #12,(R0)+               ; 'LF'' & 'NULL''
2146   020030  012720  172000                    MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
2147   020034  004737  024524                    JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
2148
2149   020040  017737  161720  001126            MOV     @XPOS,$BDDAT            ;READ X POSITION
2150   020046  042737  176000  001126            BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
2151   020054  012737  001000  001124            MOV     #1000,$GDDAT            ;LOAD BASE
2152   020062  063737  001750  001124            ADD     LFSZ0,$GDDAT            ;LOAD EXPECTED DATA
2153   020070  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
2154   020076  001401                            BEQ     TST70           ;;BR IF DELTA X IS CORRECT
2155   020100  104026                            ERROR   26              ;INCORRECT DELTA X WITH CHAR SCALE OF 1/2
2156
2157                                   ;;*****************************************************************
  (3)                                   ;*TEST 70       CHAR SCALE 1 - ROTATED - TEST DELTA X
  (3)                                   ;;*****************************************************************
  (2)  020102  000004                   TST70:  SCOPE
  (2)  020104  012737  000070  001202            MOV     #70,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2158   020112  013737  001764  001122            MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
2159   020120  012700  032530                    MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
2160   020124  012720  155640                    MOV     #155640,(R0)+           ;LOAD ''STATUS C ROT. & CHAR SCALE 1''
2161   020130  012720  116000                    MOV     #116000,(R0)+           ;LOAD 'POINT MODE''
```

```
 2162  020134  012720  001000                  MOV     #1000,(R0)+             ;X = 1000
 2163  020140  012720  001000                  MOV     #1000,(R0)+             ;Y = 1000
 2164  020144  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
 2165  020150  012720  000012                  MOV     #12,(R0)+               ; 'LF' & 'NULL'
 2166  020154  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
 2167  020160  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 2168
 2169  020164  017737  161574  001126          MOV     @XPOS,$BDDAT            ;READ X POSITION
 2170  020172  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
 2171  020200  012737  001000  001124          MOV     #1000,$GDDAT            ;LOAD BASE
 2172  020206  063737  001752  001124          ADD     LFSZ1,$GDDAT            ;LOAD EXPECTED DATA
 2173  020214  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
 2174  020222  001401                          BEQ     TST71           ;;BR IF DELTA X IS CORRECT
 2175  020224  104026                          ERROR   26                      ;INCORRECT DELTA X WITH CHAR SCALE OF 1
 2176
 2177                                  ;;**********************************************************************
   (3)                                 ;*TEST 71       CHAR SCALE 1 1/2 - ROTATED - TEST DELTA X
   (3)                                 ;;**********************************************************************
   (2)  020226  000004                  TST71:  SCOPE
   (2)  020230  012737  000071  001202          MOV     #71,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2178  020236  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
 2179  020244  012700  032530                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
 2180  020250  012720  155700                  MOV     #155700,(R0)+           ;LOAD 'STATUS C ROT. & CHAR SCALE 1 1/2'
 2181  020254  012720  116000                  MOV     #116000,(R0)+           ;LOAD 'POINT MODE'
 2182  020260  012720  001000                  MOV     #1000,(R0)+             ;X = 1000
 2183  020264  012720  001000                  MOV     #1000,(R0)+             ;Y = 1000
 2184  020270  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
 2185  020274  012720  000012                  MOV     #12,(R0)+               ; 'LF' & 'NULL'
 2186  020300  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
 2187  020304  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 2188
 2189  020310  017737  161450  001126          MOV     @XPOS,$BDDAT            ;READ X POSITION
 2190  020316  042737  176000  001126          BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
 2191  020324  012737  001000  001124          MOV     #1000,$GDDAT            ;LOAD BASE
 2192  020332  063737  001754  001124          ADD     LFSZ2,$GDDAT            ;LOAD EXPECTED DATA
 2193  020340  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
 2194  020346  001401                          BEQ     TST72           ;;BR IF DELTA X IS CORRECT
 2195  020350  104026                          ERROR   26                      ;INCORRECT DELTA X WITH CHAR SCALE OF 1 1//2
 2196
 2197                                  ;;**********************************************************************
   (3)                                 ;*TEST 72       CHAR SCALE 2 - ROTATED - TEST DELTA X
   (3)                                 ;;**********************************************************************
   (2)  020352  000004                  TST72:  SCOPE
   (2)  020354  012737  000072  001202          MOV     #72,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2198  020362  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
 2199  020370  012700  032530                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
 2200  020374  012720  155740                  MOV     #155740,(R0)+           ;LOAD 'STATUS C ROT. & CHAR SCALE 2'
 2201  020400  012720  116000                  MOV     #116000,(R0)+           ;LOAD 'POINT MODE'
 2202  020404  012720  001000                  MOV     #1000,(R0)+             ;X = 1000
 2203  020410  012720  001000                  MOV     #1000,(R0)+             ;Y = 1000
 2204  020414  012720  100000                  MOV     #100000,(R0)+           ;CHAR MODE
 2205  020420  012720  000012                  MOV     #12,(R0)+               ; 'LF' & 'NULL'
 2206  020424  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
 2207  020430  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 2208
 2209  020434  017737  161324  001126          MOV     @XPOS,$BDDAT            ;READ X POSITION
```

```
                                        BIC     #176000,$BDDAT          ;MASK  GRAPH INC.
                                        MOV     #1000,$GDDAT            ;LOAD BASE
                                        ADD     LFSZ3,$GDDAT            ;LOAD EXPECTED DATA
                                        CMP     $GDDAT,$BDDAT           ;COMPARE RESULT
                                        BEQ     TST73           ;;BR IF DELTA X IS CORRECT
                                        ERROR   26              ;INCORRECT DELTA X WITH CHAR SCALE OF 2

                                ;;**********************************************************************
                                ;*TEST 73       TEST THAT SUPERSCRIPT ON & OFF CHANGES CHAR SCALE & DELTA X & Y(REPEAT R
                                ;;**********************************************************************
                                TST73:  SCOPE
                                        MOV     #73,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                        MOV     #BUFFER,R0      ;LOAD ADRS POINTER
                                        MOV     #155340,(R0)+   ;CHAR SCALE AND ROTATE TO BE UPDATED
                                        MOV     #116000,(R0)+   ;POINT MODE
                                        MOV     #1000,(R0)+     ;X=1000
                                        MOV     #1000,(R0)+     ;Y=1000
                                        MOV     #102000,(R0)+   ;CHAR MODE
                                        MOV     #21,(R0)+       ;SUPERSCRIPT ON - LATER SUPERSCRIPT ESCAPE
                                        MOV     #172000,(R0)+   ;STOP
                                        MOV     #CSCL,R0        ;R0 POINTS TO 'LD STATUS C' INSTRS FOR SCALE
                                        MOV     #XPOSR,R3       ;R3 POINTS TO EXPECTED X DELTA
                                        MOV     #YPOSR,R4       ;R4 POINTS TO EXPECTED Y DELTA
                                        MOV     #CSCLR,R5       ;R5 POINTS TO EXPECTED CHAR SCALE
                                        MOV     #3,R2           ;R2 WHEN ZERO SAYS TO SET UP SUPERSCRIPT ESCAPE
                                        MOV     #1,R1           ;R1=0 SAYS ROTATE, R1=-1 SAYS NEXT TEST
                                        MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
                                1$:
                                        JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
                                        MOV     SREG1,$BDADR    ;SET UP REG 12 ADRS
                                        MOV     (R5),$GDDAT     ;LD EXPECTED CHAR SCALE
                                        MOV     @SREG1,$BDDAT   ;READ CHAR SCALE
                                        BIC     #176377,$BDDAT  ;SAVE ONLY CHAR SCALE
                                        CMP     $GDDAT,$BDDAT   ;CORRECT?
                                        BEQ     2$              ;BR IF SO
                                        ERROR   43              ;SUPERSCRIPT FAILED TO CHANGE CHAR SCALE
                                2$:     MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
                                        MOV     (R3),$GDDAT     ;LD EXPECTED DELTA X
                                        MOV     @XPOS,$BDDAT    ;READ X DELTA
                                        BIC     #176000,$BDDAT  ;SAVE X ONLY
                                        CMP     $GDDAT,$BDDAT   ;CORRECT?
                                        BEQ     3$              ;BR IF SO
                                        ERROR   43              ;SUPERSCRIPT DELTA X ER AT CHAR SCALE IN REG 12
                                3$:     MOV     YPOS,$BDADR     ;SET UP REG ADS 06
                                        MOV     (R4),$GDDAT     ;LD EXPECTED Y DELTA
                                        MOV     @YPOS,$BDDAT    ;READ DELTA Y
                                        BIC     #176000,$BDDAT  ;SAVE ONLY Y
                                        CMP     $GDDAT,$BDDAT   ;CORRECT?
                                        BEQ     4$              ;BR IF SO
                                        ERROR   43              ;SUPERSCRIPT DELTA Y ER AT CHAR SCALE IN REG 12
                                4$:     DEC     R2              ;TIME TO TURN ON SUPER ESCAPE?
                                        BNE     5$              ;BR IF NOT OR ON ALREADY
                                        MOV     #23,BUFFER+12   ;NOW SET UP SUPER ESCAPE
                                5$:     ADD     #2,R3           ;UPDATE X DELTA PTR
                                        ADD     #2,R4           ;UPDATE Y DELTA PTR
                                        ADD     #2,R5           ;UPDATE SCALE PTR
```

```
2261  020776  012037  032530            MOV     (R0)+,BUFFER    ;SET UP NEXT CHAR SCALE AT INSTR LOC
2262  021002  005701                    TST     R1              ;THIS PASS ROTATED?
2263  021004  001003                    BNE     6$              ;BR IF NOT
2264  021006  052737  000400  032530    BIS     #400,BUFFER     ;YES - SET CHAR ROTATE BIT AT INSTR LOC
2265  021014  005715              6$:   TST     (R5)            ;HAVE WE DONE SUPER AND SUPER ESCAPE?
2266  021016  100272                    BPL     1$              ;BR IF NOT
2267  021020  005301                    DEC     R1              ;WILL REPEAT TEST ROTATED NOW
2268  021022  100464                    BMI     TST74           ;;NEXT TEST IF ROTATED SUPERSCRIPT ALREADY DONE
2269  021024  012705  021066            MOV     #CSCLR,R5       ;RESET EXPECTED CHAR SCALE PTR
2270  021030  012702  000003            MOV     #3,R2           ;RESET R3 - TELLS WHEN TO SWITCH TO SUPER ESCAPE
2271  021034  012737  000021  032542    MOV     #21,BUFFER+12   ;RESET SUPER IN FILE
2272  021042  012703  021144            MOV     #XPOSRR,R3      ;R3 POINTS TO EXPECTED X DELTA WHEN ROTATED
2273  021046  012704  021160            MOV     #YPOSRR,R4      ;R4 POINTS TO EXPECTED Y DELTA WHEN ROTATED
2274  021052  012700  021102            MOV     #CSCL,R0        ;RESET PTR TO 'LD STATUS C' INSTRS (SCALE)
2275  021056  012737  155740  032530    MOV     #155740,BUFFER  ;SET UP CHAR SCALE INSTR THIS TIME ROTATED
2276  021064  000647                    BR      1$              ;REPEAT TEST ROTATED NOW
2277
2278  021066  001000              CSCLR:  1000    ;TABLE OF EXPECTED CHAR SCALE RESULTS
2279  021070  000400                      400
2280  021072  000000                      0
2281  021074  000400                      400
2282  021076  001000                      1000
2283  021100  001400                      1400
2284
2285  021102  155300              CSCL:   155300  ;TABLE OF CHAR SCALE INSTRS
2286  021104  155240                      155240
2287  021106  155200                      155200
2288  021110  155240                      155240
2289  021112  155300                      155300
2290
2291  021114  000775              XPOSR:  775     ;EXPECTED SUPERSCIPT DELTA X RESULTS
2292  021116  000774                      774
2293  021120  000776                      776
2294  021122  001002                      1002
2295  021124  001000                      1000
2296  021126  001000                      1000
2297
2298
2299  021130  001030              YPOSR:  1030    ;EXPECTED SUPERSCRIPT DELTA Y RESULTS
2300  021132  001022                      1022
2301  021134  001011                      1011
2302  021136  000767                      767
2303  021140  000756                      756
2304  021142  000750                      750
2305
2306  021144  000750              XPOSRR: 750     ;EXPECTED SUPERSCRIPT DELTA X ROTATED RESULTS
2307  021146  000756                      756
2308  021150  000767                      767
2309  021152  001011                      1011
2310  021154  001022                      1022
2311  021156  001030                      1030
2312
2313  021160  001000              YPOSRR: 1000    ;EXPECTED SUPERSCRIPT DELTA Y ROTATED RESULTS
2314  021162  000774                      774
2315  021164  000776                      776
2316  021166  001002                      1002
```

```
 2317  021170  001000                                  1000
 2318  021172  001000                                  1000
 2319
 2320                                          ;;*****************************************************************
  (3)                                          ;*TEST 74       TEST THAT SUBSCRIPT ON & OFF CHANGES CHAR SCALE & DELTA X & Y(REPEAT ROT
  (3)                                          ;;*****************************************************************
  (2)  021174  000004                          TST74:  SCOPE
  (2)  021176  012737  000074  001202                  MOV     #74,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2321  021204  012700  032530                         MOV     #BUFFER,R0      ;LOAD POINTER ADRS
 2322  021210  012720  155340                         MOV     #155340,(R0)+   ;CHAR SCALE AND ROTATE TO BE UPDATAED
 2323  021214  012720  116000                         MOV     #116000,(R0)+   ;POINT MODE
 2324  021220  012720  001000                         MOV     #1000,(R0)+     ;X=1000
 2325  021224  012720  001000                         MOV     #1000,(R0)+     ;Y=1000
 2326  021230  012720  102000                         MOV     #102000,(R0)+   ;CHAR MODE
 2327  021234  012720  000022                         MOV     #22,(R0)+       ;SUBSCRIPT ON - LATER SUBSCRIPT ESCAPE
 2328  021240  012720  172000                         MOV     #172000,(R0)+   ;STOP
 2329  021244  012700  021102                         MOV     #CSCL,R0        ;R0 POINTS TO 'LD STATUS C' INSTRS FOR SCALE
 2330  021250  012703  021564                         MOV     #XPOSR1,R3      ;R3 POINTS TO EXPECTED X DELTA RESULTS
 2331  021254  012704  021600                         MOV     #YPOSR1,R4      ;R4 POINTS TO EXPECTED Y DELTA RESULTS
 2332  021260  012705  021066                         MOV     #CSCLR,R5       ;R5 POINTS TO EXPECTED CHAR SCALE
 2333  021264  012702  000003                         MOV     #3,R2           ;R2 WHEN 0 SAYS TO SET UP SUB ESCAPE
 2334  021270  012701  000001                         MOV     #1,R1           ;R1=0 SAYS ROTATE, R1=-1 SAYS NEXT TEST
 2335  021274  012737  021302  001110                 MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
 2336  021302                                1$:
  (1)  021302  004737  024524                         JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 2337  021306  013737  001772  001122                 MOV     SREG1,$BDADR    ;SET UP REG ADRS 12
 2338  021314  011537  001124                         MOV     (R5),$GDDAT     ;LD EXPECTED CHAR SCALE
 2339  021320  017737  160446  001126                 MOV     @SREG1,$BDDAT   ;READ CHAR SCALE
 2340  021326  042737  176377  001126                 BIC     #176377,$BDDAT  ;SAVE ONLY SCALE
 2341  021334  023737  001124  001126                 CMP     $GDDAT,$BDDAT   ;CORRECT?
 2342  021342  001401                                 BEQ     2$              ;BR IF SO
 2343  021344  104043                                 ERROR   43              ;SUB SCRIPT FAILED TO CHANGE CHAR SCALE
 2344  021346  013737  001764  001122  2$:            MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
 2345  021354  011337  001124                         MOV     (R3),$GDDAT     ;LD EXPECTED X DELTA
 2346  021360  017737  160400  001126                 MOV     @XPOS,$BDDAT    ;READ DELTA X
 2347  021366  042737  176000  001126                 BIC     #176000,$BDDAT  ;SAVE ONLY DELTA X
 2348  021374  023737  001124  001126                 CMP     $GDDAT,$BDDAT   ;CORRECT?
 2349  021402  001401                                 BEQ     3$              ;BR IF SO
 2350  021404  104043                                 ERROR   43              ;SUBSCRIPT DELTA X ER AT CHAR SCALE IN REG 12
 2351  021406  013737  001766  001122  3$:            MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 2352  021414  011437  001124                         MOV     (R4),$GDDAT     ;LD EXPECTED DELTA Y
 2353  021420  017737  160342  001126                 MOV     @YPOS,$BDDAT    ;READ DELTA Y
 2354  021426  042737  176000  001126                 BIC     #176000,$BDDAT  ;SAVE ONLY DELTA Y
 2355  021434  023737  001124  001126                 CMP     $GDDAT,$BDDAT   ;CORRECT?
 2356  021442  001401                                 BEQ     4$              ;BR IF SO
 2357  021444  104043                                 ERROR   43              ;SUBSCRIPT DELTA Y ER AT CHAR SCALE AT REG 12
 2358  021446  005302                          4$:    DEC     R2              ;TIME TO SWITCH TO SUB ESCAPE?
 2359  021450  001003                                 BNE     5$              ;BR IF NOT OR ON ALREADY
 2360  021452  012737  000024  032542                 MOV     #24,BUFFER+12   ;NOE SET UP FOR SUB ESCAPE
 2361  021460  062703  000002                  5$:    ADD     #2,R3           ;UPDATE X DELTA RESULT PTR
 2362  021464  062704  000002                         ADD     #2,R4           ;UPDATE Y DELTA RESULT PTR
 2363  021470  062705  000002                         ADD     #2,R5           ;UPDATE CHAR SCALE RESULT PTR
 2364  021474  012037  032530                         MOV     (R0)+,BUFFER    ;SET UP NEXT CHAR SCALE AT INSTR LOC
 2365  021500  005701                                 TST     R1              ;THIS PASS ROTATED?
 2366  021502  001003                                 BNE     6$              ;BR IF NOT
 2367  021504  052737  000400  032530                 BIS     #400,BUFFER     ;YES - SET CHAR ROTATE BIT AT INSTR LOC
```

```
2368  021512  005715                    6$:     TST     (R5)            ;HAVE WE DONE SUB AND SUB ESCAPE?
2369  021514  100272                            BPL     1$              ;BR IF NOT
2370  021516  005301                            DEC     R1              ;WILL REPEAT TEST NOW ROTATED
2371  021520  100451                            BMI     TST75           ;;NEXT TEST IF ROTATED SUBSCRIPT ALREADY DONE
2372  021522  012705  021066                    MOV     #CSCLR,R5       ;RESET EXPECTED CHAR SCALE PTR
2373  021526  012702  000003                    MOV     #3,R2           ;RESET R2 - TELLS WHEN TO SWITCH TO SUB ESCAPE
2374  021532  012737  000022  032542            MOV     #22,BUFFER+12   ;RESET SUB IN FILE
2375  021540  012703  021614                    MOV     #XPORR1,R3      ;R3 POINTS TO EXPECTED X DELTA ROTATED
2376  021544  012704  021630                    MOV     #YPORR1,R4      ;R4 POINTS TO EXPECTED Y DELTA ROTATED
2377  021550  012700  021102                    MOV     #CSCL,R0        ;RESET PTR TO 'LD STATUS C' INSTRS (SCALE)
2378  021554  012737  155740  032530            MOV     #155740,BUFFER  ;SET CHAR SCALE INSTR THIS TIME ROTATED
2379  021562  000647                            BR      1$              ;REPEAT TEST CAHR ROTATED
2380
2381  021564  000775                    XPOSR1: 775     ;EXPECTED SUBSCRIPT DELTA X RESULTS
2382  021566  000774                            774
2383  021570  000776                            776
2384  021572  001002                            1002
2385  021574  001000                            1000
2386  021576  001000                            1000
2387
2388  021600  000771                    YPOSR1: 771     ;EXPECTED SUBSCRIPT DELTA Y RESULTS
2389  021602  000772                            772
2390  021604  000775                            775
2391  021606  001003                            1003
2392  021610  001006                            1006
2393  021612  001010                            1010
2394
2395  021614  001007                    XPORR1: 1007    ;EXPECTED SUBSCRIPT DELTA X ROTATED RESULTS
2396  021616  001006                            1006
2397  021620  001003                            1003
2398  021622  000775                            775
2399  021624  000772                            772
2400  021626  000770                            770
2401
2402  021630  001000                    YPORR1: 1000    ;EXPECTED SUBSCRIPT DELTA Y ROTATED RESULTS
2403  021632  000774                            774
2404  021634  000776                            776
2405  021636  001002                            1002
2406  021640  001000                            1000
2407  021642  001000                            1000
2408
2409                                    ;;*****************************************************************
  (3)                                  ;*TEST 75        TEST THAT "CR" DOES CHANGE X AND DOES NOT CHANGE Y AXIS
  (3)                                  ;;*****************************************************************
  (2)   021644  000004                  TST75:  SCOPE
  (2)   021646  012737  000075  001202          MOV     #75,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2410  021654  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
2411  021662  012700  032530                  MOV     #BUFFER,R0      ;LOAD BUFFER POINTER
2412  021666  012720  155240                  MOV     #155240,(R0)+   ;LOAD CHAR SIZE 1
2413  021672  012720  116000                  MOV     #116000,(R0)+   ;POINT MODE
2414  021676  012720  001000                  MOV     #1000,(R0)+     ;LOAD BUFFER
2415  021702  012720  001000                  MOV     #1000,(R0)+     ;
2416  021706  012720  100000                  MOV     #100000,(R0)+   ;LOAD "CHARACTER MODE"
2417  021712  012720  000015                  MOV     #15,(R0)+       ;LOAD "CR"
2418  021716  012720  172000                  MOV     #172000,(R0)+   ;LOAD DISPLAY STOP
2419  021722  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
```

```
2420
2421  021726  017737  160032  001126  1$:     MOV     @XPOS,$BDDAT    ;READ X AXIS
2422  021734  005037  001124                  CLR     $GDDAT  ;LOAD EXPECTED
2423  021740  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE EXPCT TO RCVD
2424  021746  001402                          BEQ     2$              ;YES
2425  021750  104022                          ERROR   22              ;''CR'' CHARACTER FAILED TO CHANGED X AXIS CORRECTLY
2426  021752  000421                          BR      TST76           ;;
2427
2428  021754  017737  160006  001126  2$:     MOV     @YPOS,$BDDAT    ;READ Y AXIS
2429  021762  042737  176000  001126          BIC     #176000,$BDDAT  ;MASK TO BITS 0-9
2430  021770  012737  001000  001124          MOV     #1000,$GDDAT    ;LOAD EXPECTED
2431  021776  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE EXPCT TO RCVD
2432  022004  001404                          BEQ     TST76           ;;BR IF EQUAL
2433  022006  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
2434  022014  104022                          ERROR   22              ;''CR'' CHARACTER CHANGED Y AXIS
2435
2436                                  ;;**********************************************************************
  (3)                               ;*TEST 76       TEST THAT 'BS'' (NOT ROTATED)  DOES CHANGE X BUT NOT Y AXIS
  (3)                               ;;**********************************************************************
  (2)   022016  000004              TST76:  SCOPE
  (2)   022020  012737  000076  001202          MOV     #76,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2437  022026  012700  032530                    MOV     #BUFFER,R0
2438  022032  012720  155240                    MOV     #155240,(R0)+   ;LOAD NO ROT CHAR SIZE = 1
2439  022036  012720  116000                    MOV     #116000,(R0)+   ;POINT MODE
2440  022042  012720  001000                    MOV     #1000,(R0)+     ;LOAD BUFFER
2441  022046  012720  001000                    MOV     #1000,(R0)+     ;1000,1000
2442  022052  012720  100000                    MOV     #100000,(R0)+   ;LOAD ''CHARACTER MODE''
2443  022056  012720  000010                    MOV     #10,(R0)+
2444  022062  012720  172000                    MOV     #172000,(R0)+   ;LOAD DISPLAY STOP
2445  022066  004737  024524                    JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
2446  022072  017737  157666  001126  1$:     MOV     @XPOS,$BDDAT    ;READ X AXIS
2447  022100  012737  001000  001124          MOV     #1000,$GDDAT ;LOAD EXPECTED
2448  022106  163737  001742  001124          SUB     CHSZ1,$GDDAT    ;ADJUST EXPECTED
2449  022114  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE
2450  022122  001405                          BEQ     2$              ;YES
2451  022124  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
2452  022132  104022                          ERROR   22              ;'BS'' CHARACTER FAILED TO CHANGED X AXIS CORRECTLY
2453  022134  000435                          BR      TST77           ;;
2454
2455  022136  017737  157624  001126  2$:     MOV     @YPOS,$BDDAT    ;READ Y AXIS
2456  022144  042737  176000  001126          BIC     #176000,$BDDAT  ;MASK TO BITS 0-9
2457  022152  012737  001000  001124          MOV     #1000,$GDDAT    ;LOAD EXPECTED
2458  022160  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE EXPCT TO RCVD
2459  022166  001405                          BEQ     3$              ;YES
2460  022170  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
2461  022176  104022                          ERROR   22              ;'BS'' CHARACTER CHANGED Y AXIS
2462  022200  000413                          BR      TST77           ;;
2463
2464                                  ;TEST THAT ''SHIFT-OUT'' STATUS BIT IS NOT SET
2465
2466  022202  017737  157554  001126  3$:     MOV     @SREG0,$BDDAT           ;READ STATUS
2467  022210  032737  000100  001126          BIT     #100,$BDDAT
2468  022216  001404                          BEQ     TST77           ;;BR IF EQUAL
2469  022220  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
2470  022226  104035                          ERROR   35              ;SHIFT OUT STATUS BIT IS SET
2471
```

```
                                ;;*****************************************************************
                                ;*TEST 77       TEST THAT 'BS'' (ROTATED)  DOES CHANGE Y BUT NOT X AXIS
                                ;;*****************************************************************
                        TST77:  SCOPE
                                MOV     #77,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                MOV     #BUFFER,R0
                                MOV     #155640,(R0)+   ;LOAD ROTATE CHSIZE = 1
                                MOV     #116000,(R0)+   ;POINT MODE
                                MOV     #1000,(R0)+     ;LOAD BUFFER
                                MOV     #1000,(R0)+     ;1000,1000
                                MOV     #100000,(R0)+   ;LOAD ''CHARACTER MODE''
                                MOV     #10,(R0)+
                                MOV     #172000,(R0)+   ;LOAD DISPLAY STOP
                                JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
                        1$:     MOV     @YPOS,$BDDAT    ;READ Y AXIS
                                BIC     #176000,$BDDAT
                                MOV     #1000,$GDDAT ;LOAD EXPECTED
                                SUB     CHSZ1,$GDDAT    ;ADJUST EXPECTED
                                CMP     $GDDAT,$BDDAT   ;COMPARE
                                BEQ     2$              ;YES
                                MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
                                ERROR   22              ;'BS'' (ROTATED)  CHARACTER FAILED TO CHANGED Y AXIS CORR
                                BR      TST100          ;;

                        2$:     MOV     @XPOS,$BDDAT    ;READ X AXIS
                                BIC     #176000,$BDDAT  ;MASK TO BITS 0-9
                                MOV     #1000,$GDDAT    ;LOAD EXPECTED
                                CMP     $GDDAT,$BDDAT           ;COMPARE EXPCT TO RCVD
                                BEQ     3$              ;YES
                                MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
                                ERROR   22              ;'BS'' (ROTATED)  CHARACTER CHANGED Y AXIS
                                BR      TST100          ;;

                        ;TEST THAT ''SHIFT-OUT'' STATUS BIT IS NOT SET

                        3$:     MOV     @SREG0,$BDDAT           ;READ STATUS
                                BIT     #100,$BDDAT
                                BEQ     TST100          ;;BR IF EQUAL
                                MOV     SREG0,$BDADR    ;SET REG ADRS 02
                                ERROR   35              ;SHIFT OUT STATUS BIT IS SET

                                ;;*****************************************************************
                                ;*TEST 100      TEST THAT ''SHIFT-OUT'' GENERATES A STATUS BIT
                                ;;*****************************************************************
                        TST100: SCOPE
                                MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                                MOV     #100,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
                        ;SHIFT-OUT <LOW BYTE>, FOLLOWED BY CODE 77 <HIGH BYTE>
                                MOV     #340,PSW                ;RAISE PRIORITY
                                MOV     #BUFFER,R0      ;LOAD BUFFER POINTER
                                MOV     #172000,(R0)+   ;LOAD ''ST''
                                MOV     #116000,(R0)+   ;POINT MODE
                                MOV     #1000,(R0)+
                                MOV     #1000,(R0)+     ;1000,1000
                                MOV     #100000,(R0)+   ;LOAD ''CHARACTER MODE''
                                MOV     #37416,(R0)+    ;''SHIFT-OUT'' IN LOW BYTE #77 IN HIGH BYTE
```

```
2519  022530  012720  000017                  MOV     #17,(R0)+               ;LOAD "SHIFT-IN"
2520  022534  012720  172000                  MOV     #172000,(R0)+           ;LOAD DISPLAY STOP
2521  022540  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
2522  022544  012737  000001  024632          MOV     #1,NOFLAG               ;ENABLE NO FLAG EXIT
2523  022552  004737  024552                  JSR     PC,CONT                 ;RESUME DISPLAY AND WAIT FOR STOP FLAG
2524  022556  012737  176000  001124          MOV     #176000,$GDDAT
2525  022564  017737  157176  001126          MOV     @YPOS,$BDDAT    ;READ CHARACTER REG
2526  022572  042737  001777  001126          BIC     #1777,$BDDAT    ;MASK TO BITS 10-15
2527  022600  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE EXPCT TO RCVD
2528  022606  001405                          BEQ     1$
2529  022610  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
2530  022616  104021                          ERROR   21              ;CHARACTER REGISTER IN ERROR
2531  022620  000477                          BR      TST101          ;;
2532
2533  022622  017737  157134  001126  1$:     MOV     @SREG0,$BDDAT           ;READ STATUS REGISTER
2534  022630  012737  000100  001124          MOV     #BIT6,$GDDAT    ;LOAD EXPECTED
2535  022636  032737  000100  001126          BIT     #100,$BDDAT
2536  022644  001005                          BNE     2$
2537  022646  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
2538  022654  104036                          ERROR   36              ;SHIFT OUT STATUS BIT FAILED TO SET
2539  022656  000460                          BR      TST101          ;;
2540
2541  022660  017737  157100  001126  2$:     MOV     @XPOS,$BDDAT    ;READ X POS
2542  022666  012737  001000  001124          MOV     #1000,$GDDAT    ;LOAD EXPECTED
2543  022674  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE EXPCT TO RCVD
2544  022702  001405                          BEQ     3$
2545  022704  013737  001764  001122          MOV     XPOS,$BDADR     ;SET UP REG ADRS 04
2546  022712  104022                          ERROR   22              ;SHIFT-OUT CHARACTER CHANGED X AXIS
2547  022714  000441                          BR      TST101          ;;
2548
2549  022716  017737  157044  001126  3$:     MOV     @YPOS,$BDDAT    ;READ Y POS
2550  022724  042737  176000  001126          BIC     #176000,$BDDAT  ;MASK
2551  022732  012737  001000  001124          MOV     #1000,$GDDAT    ;LOAD EXPECTED
2552  022740  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE EXPCT TO RCVD
2553  022746  001404                          BEQ     4$              ;;BR IF EQUAL
2554  022750  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
2555  022756  104022                          ERROR   22              ;SHIFT-OUT CHARACTER CHANGED Y AXIS
2556
2557  022760                          4$:
 (1)  022760  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
2558  022764  017737  156772  001126          MOV     @SREG0,$BDDAT   ;READ STATUS
2559  022772  005037  001124                  CLR     $GDDAT          ;CLEAR EXPECTED
2560  022776  032737  000100  001126          BIT     #BIT6,$BDDAT            ;TEST IF SHIFT-OUT SET
2561  023004  001404                          BEQ     5$              ;;BR IF ZERO
2562  023006  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
2563  023014  104035                          ERROR   35              ;SHIFT-OUT STATUS BIT FAILED TO CLEAR
2564  023016  000005                  5$:     RESET
2565
2566                                  ;;**********************************************************************
 (3)                                  ;*TEST 101       TEST THAT "SHIFT-OUT" DOES NOT GENERATE A STATUS BIT ON CODE 0-37
 (3)                                  ;;**********************************************************************
 (2)  023020  000004                  TST101: SCOPE
 (1)  023022  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
 (2)  023030  012737  000101  001202          MOV     #101,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
2567                                  ;("SHIFT-OUT" FOLLOWED BY CODE 0 THRU 37 EXCEPT #17)
2568
```

```
 2569  023036  012737  000016  001726          MOV     #16,DSAVE
 2570  023044  012700  032530                  MOV     #BUFFER,R0      ;LOAD POINTER
 2571  023050  012720  100000                  MOV     #100000,(R0)+   ;SET 'CHAR' MODE
 2572  023054  012720  000016                  MOV     #16,(R0)+       ;LOAD ''SHIFT-OUT'' INTO THE LOW BYTE
 2573  023060  012720  172000                  MOV     #172000,(R0)+   ;LOAD STOP
 2574  023064  013777  001726  156622  1$:     MOV     DSAVE,@DBUF1    ;LOAD DISPLAY BUFFER
 2575  023072  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 2576
 2577  023076  005037  001124                  CLR     $GDDAT
 2578  023102  017737  156654  001126          MOV     @SREG0,$BDDAT   ;READ STATUS
 2579  023110  032737  000100  001126          BIT     #100,$BDDAT     ;TEST FOR SHIFT BIT
 2580  023116  001405                          BEQ     2$
 2581  023120  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
 2582  023126  104035                          ERROR   35              ;SHIFT STATUS BIT SET IN ERROR
 2583  023130  000412                          BR      TST102          ;;
 2584
 2585  023132  105237  001727          2$:     INCB    DSAVE+1
 2586  023136  122737  000017  001727          CMPB    #17,DSAVE+1     ;TEST FOR #17 (SHIFT-IN)
 2587  023144  001772                          BEQ     2$              ;;BR AND TRY AGAIN
 2588  023146  122737  000040  001727          CMPB    #40,DSAVE+1
 2589  023154  001343                          BNE     1$              ;;BR BACK IF NOT #40
 2590
 2591                          ;;*******************************************************************
  (3)                          ;*TEST 102       TEST THAT 'SHIFT-OUT' IN THE HIGH BYTE FOLLOWED BY A CHARACTER
  (3)                          ;;*******************************************************************
  (2)  023156  000004          TST102: SCOPE
  (2)  023160  012737  000102  001202          MOV     #102,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 2592                          ; IN THE NEXT LOW BYTE GENERATES A STATUS BIT
 2593
 2594  023166  012737  000340  177776          MOV     #340,PSW                ;
 2595  023174  012777  100000  156510          MOV     #100000,@DBUF   ;LOAD SET 'CHAR' MODE
 2596  023202  012777  007000  156504          MOV     #7000,@DBUF1    ;LOAD 'SHIFT-OUT' INTO THE HIGH BYTE
 2597  023210  012777  000040  156500          MOV     #40,@DBUF2      ;LOAD A SHIFT-OUT CHARACTER IN THE NEXT
 2598                                                                  ;WORD <LOW BYTE>
 2599  023216  012777  172000  156474          MOV     #172000,@DBUF3  ;LOAD STOP
 2600  023224  012737  000001  024632          MOV     #1,NOFLAG               ;SET NO FLAG EXIT
 2601  023232  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 2602
 2603  023236  012737  000100  001124          MOV     #BIT6,$GDDAT    ;LOAD EXPECTED
 2604  023244  017737  156512  001126          MOV     @SREG0,$BDDAT   ;READ STATUS
 2605  023252  032737  000100  001126          BIT     #100,$BDDAT     ;TEST THE STATUS REGISTER
 2606  023260  001005                          BNE     1$
 2607  023262  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
 2608  023270  104036                          ERROR   36              ;SHIFT-OUT IN THE HIGH BYTE FAILED TO SET STATUS BIT
 2609  023272  000421                          BR      TST103          ;;
 2610
 2611  023274  017737  156466  001126  1$:     MOV     @YPOS,$BDDAT    ;READ Y POS
 2612  023302  042737  001777  001126          BIC     #1777,$BDDAT    ;MASK TO BITS 15-10
 2613  023310  012737  100000  001124          MOV     #BIT15,$GDDAT   ;LOAD EXPECTED
 2614  023316  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE EXPCT TO RCVD
 2615  023324  001404                          BEQ     TST103          ;;BR IF EQUAL
 2616  023326  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 2617  023334  104021                          ERROR   21              ;CHARACTER REGISTER IN ERROR AFTER A
 2618                                                                  ;''SHIFT-OUT'' <HIGH BYTE> FOLLOWED BY
 2619                                                                  ; #40 <LOW BYTE NEXT WORD>
 2620
```

```
                                        ;;*****************************************************************
                                        ;*TEST 103      TEST THAT RESET WILL CLEAR 'SHIFT-OUT' STATUS BIT
                                        ;;*****************************************************************
                                        TST103: SCOPE
                                                MOV     #40,$TIMES      ;;DO 40 ITERATIONS
                                                MOV     #103,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
2623    023354  012737  000340  177776          MOV     #340,PSW        ;RAISE PRIORITY
2624    023362  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG 02 ADRS
2625    023370  012737  100000  032530          MOV     #100000,BUFFER  ;LD CHAR INSTR
2626    023376  012737  007000  032532          MOV     #7000,BUFFER+2  ;SET UP FOR SHIFT OUT
2627    023404  012737  000040  032534          MOV     #40,BUFFER+4    ;SET UP SHIFT OUT CHAR
2628    023412  012737  172000  032536          MOV     #172000,BUFFER+6  ;LD STOP INSTR
2629    023420  012737  000001  024632          MOV     #1,NOFLAG       ;SET NO FLAG EXIT
2630    023426  004737  024524                  JSR     PC,EXECUTE      ;START DISPLAY
2631    023432  000005                          RESET                   ;CLR SHIFT OUT
2632    023434  005037  001124                  CLR     $GDDAT          ;EXPECT 0
2633    023440  017737  156316  001126          MOV     @SREG0,$BDDAT   ;READ REG 02
2634    023446  042737  177677  001126          BIC     #177677,$BDDAT  ;SAVE ONLY SHIFT-OUT
2635    023454  001404                          BEQ     TST104          ;;NEXT TEST IF CLEARED
2636    023456  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
2637    023464  104032                          ERROR   32              ;RESET FAILED TO CLR SHIFT-OUT
2638
2639                                    ;;*****************************************************************
                                        ;*TEST 104      INTERRUPT PRIORITY TEST (DEVICE LEVEL -1)
                                        ;;*****************************************************************
                                        TST104: SCOPE
                                                MOV     #104,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
2640    023476  012737  000340  177776          MOV     #340,PSW        ; RAISE CPU STAT.
2641    023504  012777  023574  156306          MOV     #1$,@DDONE      ; SET INTERRUPT ADDRESS.
2642    023512  012737  173400  032530          MOV     #173400,BUFFER  ; SET ''STOP AND INTERRUPT'' INSTR
2643    023520  013777  001712  156232          MOV     DBUF,@DPC       ;LOAD THE DISPLAY P.C.
2644    023526  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
2645    023534  012737  172000  001124          MOV     #172000,$GDDAT  ;EXPECT STOP, MODE & INTENSITY BITS
2646    023542  013700  001736                  MOV     BRLEV1,R0       ; GET BUS PRIORITY.
2647    023546  162700  000040                  SUB     #40,R0          ;MAKE ONE LOWER
2648    023552  010037  177776                  MOV     R0,PSW          ;LOWER PRIORITY TO DEVICE LEVEL -1
2649    023556  000240                          NOP                     ; *** HERE IF NO INTERRUPT ***
2650    023560  000240                          NOP
2651    023562  017737  156174  001126          MOV     @SREG0,$BDDAT   ;GET REG 02
2652    023570  104034                          ERROR   34              ;NO STOP INTERRUPT ON BR LEVEL INDICATED -1
2653    023572  000406                          BR      TST105          ;;BR TO NEXT TEST
2654
2655    023574  012716  023602          1$:     MOV     #2$,(SP)        ; OK, FIX STACKED PC...
2656    023600  000002                          RTI                     ;...AND DISMISS INTERRUPT.
2657    023602  013777  002022  156210  2$:     MOV     DDONE1,@DDONE   ;RESET VECTOR
2658
2659                                    ;;*****************************************************************
                                        ;*TEST 105      INTERRUPT PRIORITY TEST (DEVICE LEVEL)
                                        ;;*****************************************************************
                                        TST105: SCOPE
                                                MOV     #105,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
2660    023620  013737  001762  001122          MOV     SREG0,$BDADR    ;SET UP REG ADRS 02
2661    023626  012737  172000  001124          MOV     #172000,$GDDAT  ;EXPECT STOP, MODE & INTENSITY BITS
2662    023634  012737  000340  177776          MOV     #340,PSW        ; RAISE CPU STAT.
2663    023642  012777  023704  156150          MOV     #1$,@DDONE      ; SET INTERRUPT ADDRESS.
```

```
 2664  023650  012737  173400  032530          MOV     #173400,BUFFER  ; SET ''STOP AND INTERRUPT'' INSTR
 2665  023656  013777  001712  156074          MOV     DBUF,@DPC
 2666  023664  000240                          NOP
 2667  023666  000240                          NOP
 2668  023670  013737  001736  177776          MOV     BRLEV1,PSW      ;LOWER PRIORITY TO DEVICE LEVEL
 2669  023676  000240                          NOP                     ; *** SHOULD NOT INTERRUPT ***
 2670  023700  000240                          NOP
 2671  023702  000410                          BR      3$              ; IT DIDN'T, CONTINUE.
 2672
 2673  023704  012716  023712          1$:     MOV     #2$,(SP)        ; IT DID, FIX PC...
 2674  023710  000002                          RTI                     ;...DISMISS...
 2675  023712  017737  156044  001126  2$:     MOV     @SREG0,$BDDAT   ;GIVE THEM REG 02
 2676  023720  104034                          ERROR   34              ;VS60 INTERRUPTED ON THE WRONG BR LEVEL
 2677  023722  000425                          BR      TST106          ;;NEXT TEST
 2678
 2679  023724  012777  023754  156066  3$:     MOV     #4$,@DDONE      ; CHANGE INTERRUPT ADDRESS.
 2680  023732  005037  177776                  CLR     PSW             ;LOWER PRIORITY LEVEL
 2681  023736  000240                          NOP                     ; *** SHOULD INTERRUPT ***
 2682  023740  000240                          NOP
 2683  023742  017737  156014  001126          MOV     @SREG0,$BDDAT   ; DIDN'T, GIVE THEM REG 02
 2684  023750  104034                          ERROR   34              ;VS60 STOP FAILED TO INTERRUPT AFTER LOWERING PRIORITY
 2685  023752  000411                          BR      TST106          ;;NEXT TEST
 2686
 2687  023754  012716  023762          4$:     MOV     #5$,(SP)        ; OK, FIX PC...
 2688  023760  000002                          RTI                     ;...DISMISS...
 2689  023762  012737  000340  177776  5$:     MOV     #340,PSW        ;RAISE TO HIGHEST PRIORITY BEFORE ADVANCING
 2690  023770  013777  002022  156022          MOV     DDONE1,@DDONE   ; AND RESET DONE VECTOR.
 2691
 2692                                  ;;**************************************************************
  (3)                                  ;*TEST 106       RESET CLEARS CHARACTER REGISTER
  (3)                                  ;;**************************************************************
  (2)  023776  000004                  TST106: SCOPE
  (1)  024000  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
  (2)  024006  012737  000106  001202          MOV     #106,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 2693  024014  013737  001766  001122          MOV     YPOS,$BDADR     ;SET UP REG ADRS 06
 2694  024022  005077  155764                  CLR     @STKPT          ;LOWER MAINT SWITCHES
 2695  024026  012737  100000  032530          MOV     #100000,BUFFER
 2696  024034  012737  077577  032532          MOV     #77577,BUFFER+2 ;LOAD A ''177'' CHARACTER
 2697  024042  012737  172000  032534          MOV     #172000,BUFFER+4
 2698  024050  004737  024524                  JSR     PC,EXECUTE              ;START DISPLAY AND WAIT FOR STOP FLAG
 2699
 2700  024054  000005                          RESET
 2701  024056  005037  001124                  CLR     $GDDAT          ;CLEAR EXPECTED
 2702  024062  017737  155700  001126          MOV     @YPOS,$BDDAT    ;READ CHAR REG
 2703  024070  042737  001777  001126          BIC     #1777,$BDDAT
 2704  024076  001401                          BEQ     TST107          ;;BR IF CLEARED
 2705  024100  104032                          ERROR   32                      ;RESET ''INIT'' FAILED TO CLEAR CHARACTER REGISTER
 2706
 2707                                  ;;**************************************************************
  (3)                                  ;*TEST 107       TEST THE 'FRAMES PER SECOND' SYNC FEATURE - START 210 FOR EXT SYNC TSTIN
  (3)                                  ;;**************************************************************
  (2)  024102  000004                  TST107: SCOPE
  (1)  024104  012737  000001  001166          MOV     #1,$TIMES       ;;DO 1 ITERATION
  (2)  024112  012737  000107  001202          MOV     #107,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 2708  024120  000005                          RESET                   ;INITAIALIZE SYSTEM
 2709  024122  005037  001122                  CLR     $BDADR          ;TYPEOUT DATA NOT APPLICABLE
```

```
2710  024126  005037  001124                    CLR     $GDDAT          ;N/A
2711  024132  005037  001126                    CLR     $BDDAT          ;N/A
2712  024136  012700  032530                    MOV     #BUFFER,R0      ;SET UP ADRS FOR INSTRS
2713  024142  012720  170000                    MOV     #170000,(R0)+   ;LD STATUS A (SYNC INSTR)
2714  024146  012720  172000                    MOV     #172000,(R0)+   ;LD STOP INSTR
2715  024152  012720  170000                    MOV     #170000,(R0)+   ;LD ANOTHER LD STATUS A INSTR
2716  024156  012720  172000                    MOV     #172000,(R0)+   ;STOP INSTR
2717  024162  012702  024346                    MOV     #TMSAV,R2       ;SET UP ADRS FOR SYNC COUNTS
2718  024166  005000                    1$:     CLR     R0              ;CLR SYNC TIMER
2719  024170  012777  032530  155562            MOV     #BUFFER,@DPC    ;START
2720  024176  005200                    2$:     INC     R0              ;COUNT SYNC TIMER
2721  024200  001413                            BEQ     4$              ;SHOULD NEVER OVFLO - NO SYNC TMG
2722  024202  005777  155554                    TST     @SREG0          ;DONE SET?
2723  024206  100373                            BPL     2$              ;BR IF NOT
2724  024210  005000                            CLR     R0              ;CLR TIMER
2725  024212  005277  155542                    INC     @DPC            ;RESUME - NOW NPR REQ IN SYNC WITH INTERNAL OSC
2726  024216  005200                    3$:     INC     R0              ;COUNT TIMER
2727  024220  001403                            BEQ     4$              ;SHOULD NEVER OVFLO - NO SYNC TIMING
2728  024222  005777  155534                    TST     @SREG0          ;HAS STOP BEEN PICKED UP YET?
2729  024226  100373                            BPL     3$              ;BR IF NOT
2730  024230  010022                    4$:     MOV     R0,(R2)+        ;SAVE TIMER RESULTS
2731  024232  062737  000004  032534            ADD     #4,BUFFER+4     ;SET UP NEXT SYNC
2732  024240  062737  000004  032530            ADD     #4,BUFFER       ;SET UP NEXT SYNC
2733  024246  022737  170020  032530            CMP     #170020,BUFFER  ;ALL SYNCS TESTED?
2734  024254  001344                            BNE     1$              ;BR IF NOT
2735  024256  013701  024352            5$:     MOV     TMSAV+4,R1      ;40 FPS CNT TO R1
2736  024262  163701  024346                    SUB     TMSAV,R1        ;SUB FULL SPEED CNT
2737  024266  162701  000020                    SUB     #20,R1          ;MAKE SURE DIFF IS GREATER THAN 20
2738  024272  100001                            BPL     6$              ;BR IF 40 FPS SLOWER THAN FULL SPEED
2739  024274  104040                            ERROR   40              ;40 FPS SYNC ER
2740  024276  013701  024350            6$:     MOV     TMSAV+2,R1      ;30 FPS CNT TO R1
2741  024302  163701  024352                    SUB     TMSAV+4,R1      ;SUB 40 FPS CNT
2742  024306  162701  000020                    SUB     #20,R1          ;MAKE SURE DIFF IS GREATER THAN 20
2743  024312  100001                            BPL     7$              ;BR IF 30 FPS SLOWER THAN 40 FPS
2744  024314  104040                            ERROR   40              ;30 FPS SYNC ERROR
2745  024316  005737  024356            7$:     TST     EXSYNC          ;START 210 SAYS LOOK FOR EXT SYNC TMG
2746  024322  001404                            BEQ     8$              ;BR IF NOT
2747  024324  005737  024354                    TST     TMSAV+6         ;LOOK FOR NON-ZERO TMG CNT
2748  024330  001013                            BNE     TST110          ;;NEXT TEST IF TIMING THERE
2749  024332  104040                            ERROR   40              ;EXT SYNC FAILED TO TRIGGER DISPLAY
2750  024334  005737  024354            8$:     TST     TMSAV+6         ;LOOK FOR ZERO TMG CNT
2751  024340  001407                            BEQ     TST110          ;;NEXT TEST IF TIMING NOT FOUND
2752  024342  104040                            ERROR   40              ;EXT SYNC TIMING FOUND WHEN NOT EXPECTED
2753  024344  000405                            BR      TST110          ;;LOOK FOR LOOP ON TEST SW
2754
2755  024346  000000            TMSAV:  0                               ;FULL SPEED CNT
2756  024350  000000                    0                               ;30 FPS CNT
2757  024352  000000                    0                               ;40 FPS CNT
2758  024354  000000                    0                               ;EXTERNAL CNT
2759
2760  024356  000000            EXSYNC: 0                               ;ZERO(DON'T EXPECT EXT TMG), NON-ZERO(EXPECT EXT TMG)
2761
2762
  (3)                               ;;*****************************************************************
  (3)                               ;*TEST 110      TEST FOR HALT AT END OF DIAGNOSTIC (SW12 SET)
  (3)                               ;;*****************************************************************
  (2)  024360  000004               TST110: SCOPE
```

```
  (2)  024362  012737  000110  001202          MOV     #110,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
2763   024370  032777  010000  154542          BIT     #SW12,@SWR      ; IS SW12 SET ?
2764   024376  001401                          BEQ     $EOP            ;BR IF NOT
2765   024400  000000                          HALT                    ;COMPLETED PASS - SW12 SAYS HALT
```

```
 2767                                   .SBTTL  END OF PASS ROUTINE
  (1)
  (2)                                   ;;**********************************************************************
  (1)                                   ;*INCREMENT THE PASS NUMBER ($PASS)
  (1)                                   ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
  (1)                                   ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
  (1)                                   ;*IF THERES A MONITOR GO TO IT
  (1)                                   ;*IF THERE ISN'T JUMP TO RESTRT
  (1)
  (1)  024402                           $EOP:
  (1)  024402  000004                           SCOPE
  (1)  024404  005037  001102                   CLR     $TSTNM          ;;ZERO THE TEST NUMBER
  (1)  024410  005037  001166                   CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
  (1)  024414  005237  001204                   INC     $PASS           ;;INCREMENT THE PASS NUMBER
  (1)  024420  042737  100000  001204           BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
  (1)  024426  005327                           DEC     (PC)+           ;;LOOP?
  (1)  024430  000001                   $EOPCT: .WORD   1
  (1)  024432  003022                           BGT     $DOAGN          ;;YES
  (1)  024434  012737                           MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
  (1)  024436  000001                   $ENDCT: .WORD   1
  (1)  024440  024430                           $EOPCT
  (1)  024442  104401  024507                   TYPE    ,$ENDMG         ;;TYPE 'END PASS #'
  (2)  024446  013746  001204                   MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
  (2)  024452  104405                           TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
  (1)  024454  104401  024504                   TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
  (1)  024460  013700  000042           $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
  (1)  024464  001405                           BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
  (1)  024466  000005                           RESET                   ;;CLEAR THE WORLD
  (1)  024470  004710                   $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
  (1)  024472  000240                           NOP                     ;;SAVE ROOM
  (1)  024474  000240                           NOP                     ;;FOR
  (1)  024476  000240                           NOP                     ;;ACT11
  (1)  024500                           $DOAGN:
  (1)  024500  000137                           JMP     @(PC)+          ;;RETURN
  (1)  024502  002460                   $RTNAD: .WORD   RESTRT
  (1)  024504     377     377     000   $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
  (1)  024507     015  042412  042116   $ENDMG: .ASCIZ  <15><12>/END PASS #/
```

```
                                        ;;*****************************************************************
                                        ;SUBROUTINE TO START DISPLAY AND WAIT FOR STOP FLAG
                                        ;;*****************************************************************
024524                                  EXECUTE:
024524  032737  000001  001204                BIT     #BIT0,$PASS     ;ODD # PASS?
024532  001403                                BEQ     EXEC            ;BR IF NOT
024534  052777  100000  155250                BIS     #BIT15,@STKPT   ;RUN IN SERIAL MODE ON ODD # PASSES
024542  012777  032530  155210  EXEC:         MOV     #BUFFER,@DPC            ;START DISPLAY
024550  000403                                BR EXECRS

                                        ;;*****************************************************************
                                        ;SUBROUTINE TO CONTINUE DISPLAY AND WAIT FOR STOP FLAG
                                        ;;*****************************************************************

024552  012777  000001  155200  CONT:    MOV     #1,@DPC                 ;RESUME DISPLAY
024560  010046                  EXECRS:   MOV     R0,-(SP)                ;SAVE R0
024562  012700  000200                    MOV     #BIT7,R0                ;LOAD DELAY
024566  005777  155170          1$:       TST     @SREG0                  ;WAIT FOR STOP FLAG
024572  100410                            BMI     2$                      ;BR IF SET
024574  005300                            DEC     R0                      ;DELAY
024576  001373                            BNE     1$                      ;BR IF NOT DONE
024600  005737  024632                    TST     NOFLAG                  ;VALID NO STOP FLAG CONDITION ?
024604  001003                            BNE     2$                      ;BR IF YES
024606  012600                            MOV     (SP)+,R0                ;RESTORE R0 IN CASE OF LOOP ON ER SW
024610  104037                            ERROR   37                      ;STOP FLAG FAILED TO SET
024612  000401                            BR      3$                      ;R0 ALREADY RESTORED
024614  012600                  2$:       MOV     (SP)+,R0                ;RESTORE R0
024616  005037  024632          3$:       CLR     NOFLAG                  ;ENSURE RESET FLAG
024622  042777  100000  155162            BIC     #BIT15,@STKPT           ;RETURN WITH MAINT SW 4 OFF
024630  000207                            RTS     PC                      ;EXIT

024632  000000                  NOFLAG: 0

                                        ;;*****************************************************************
                                        ;THIS CODE RESETS ALL VS60 VECTORS WITH HALTS
                                        ;;*****************************************************************
024634  012737  000340  177776  RSTVEC:  MOV     #340,PSW                ;RESET PRIORITY TO HIGHEST LEVEL
024642  012701  000004                    MOV     #4,R1                   ;SET UP VECTOR LOCATION COUNT
024646  013700  001246                    MOV     $VECT1,R0               ;GET 1ST VS60 VECTOR ADRS
024652  042700  160000                    BIC     #160000,R0              ; STRIP PSW BITS.
024656  010010                  1$:       MOV     R0,(R0)                 ;SET UP ADRS TO HALT
024660  062720  000002                    ADD     #2,(R0)+                ;POINT IT TO HALT
024664  005020                            CLR     (R0)+                   ;SET UP HALT
024666  005301                            DEC     R1                      ;COUNT THIS VECTOR RESTORE
024670  001372                            BNE     1$                      ;BR IF MORE TO RESTORE
024672  000207                            RTS     PC                      ;RETURN IF ALL DONE

                                        ;;*****************************************************************
                                        ; ROUTINE TO MULTIPLY AN X OR Y POINT BY A SCALE FACTOR.
                                        ; SAVE SCALED POINT VALUE IN $GDDAT.
                                        ; *** TWO VERSION SUPPLIED                              ***
                                        ; *** FOR BEFORE AND AFTER SCALING ECO.                 ***
                                        ; *** SWR BIT 10 = 1 IF ECO IS INSTALLED                        ***
                                        ; ***                       G.P. MAY '9                 ***
                                        ;;*****************************************************************
024674  010146                  MULSCL:   MOV     R1,-(SP)                ;SAVE SCALE ON STACK
```

```
2825  024676  010046                        MOV     R0,-(SP)        ;SAVE POINT ON STACK
2826  024700  001001                        BNE     1$
2827  024702  005001                        CLR     R1              ; SET EXP POINT = 0...
2828  024704  005701                1$:     TST     R1
2829  024706  001461                        BEQ     11$             ;...AND EXIT IF EITHER IS 0.
2830
2831  024710  032777  002000  154222        BIT     #SW10,@SWR
2832  024716  001411                        BEQ     2$              ; BR IF SCALING ECO NOT IN.
2833                                ;
2834                                ; USE THE FOLLOWING AFTER SCALING ECO (#???) IS INSTALLED.
2835                                ; SCALE VALUE IS X 2^2  (I.E. 3 = 3/4, 4 = 1, ETC).
2836                                ;
2837  024720  010046                        MOV     R0,-(SP)        ; POINT...
2838  024722  010146                        MOV     R1,-(SP)        ;...TIMES SCALE [2^2]...
2839  024724  004737  027504                JSR     PC,$MULT
2840  024730  012601                        MOV     (SP)+,R1        ;... TO R1 [2^2].
2841  024732  005726                        TST     (SP)+           ; DISCARD HI HALF OF PRODUCT.
2842  024734  006201                        ASR     R1              ; RESCALE RESULT.
2843  024736  006201                         ASR    R1
2844  024740  000444                        BR      11$             ; AND EXIT.
2845                                ;
2846                                ; USE THE FOLLOWING BEFORE SCALING ECO.
2847                                ;
2848  024742  005003                2$:     CLR     R3              ;CLR UNITY MULTIPLIER
2849  024744  010002                        MOV     R0,R2           ;PUT POINT IN R2 ALSO
2850  024746  005004                        CLR     R4              ;CLR LSB ROUNDING INDICATOR
2851  024750  162701  000004        3$:     SUB     #4,R1           ;SUBTRACT ALL UNITY TIMES
2852  024754  002405                        BLT     4$              ;BR IF NOW TO FRACTIONAL PART
2853  024756  005203                        INC     R3              ;RECORD IN UNITY MULTIPLIER
2854  024760  005701                        TST     R1              ;DOES SCALE HAVE FRACTIONAL PART?
2855  024762  001372                        BNE     3$              ;BR IF FRACTION EXISTS
2856  024764  005000                        CLR     R0              ;NO FRACTIONAL PART EXISTS
2857  024766  000421                        BR      7$              ;GO MUL POINT BY SCALE
2858  024770  006200                4$:     ASR     R0              ;DIVIDE BY TWO
2859  024772  103001                        BCC     5$              ;BR IF NO CARRY
2860  024774  005204                        INC     R4              ;SAVE LSB SHIFT OUT FOR ROUNDING AT 1/4 OR 3/4
2861  024776  062701  000002        5$:     ADD     #2,R1           ;COUNT DIVISION BY TWO
2862  025002  001413                        BEQ     7$              ;GO MUL POINT BY SCALE IF FRACT PART IS 1/2
2863  025004  003006                        BGT     6$              ;BR IF FRACT PART IS 3/4
2864  025006  006200                        ASR     R0              ;ADJUST FRACTIONAL PART TO 1/4
2865  025010  103010                        BCC     7$              ;GO MUL POINT BY SCALE IF NO LSB CARRY OUT
2866  025012  005704                        TST     R4              ;WAS THER A PREVIOUS CARRY OUT?
2867  025014  001406                        BEQ     7$              ;BR IF NOT - GO MUL POINT BY SCALE
2868  025016  005200                        INC     R0              ;IF SO - TWO LSB CARRIES ROUNDS UP TO 1
2869  025020  000404                        BR      7$              ;GO MUL POINT BY SCALE
2870  025022  060400                6$:     ADD     R4,R0           ;ALWAYS ROUND UP IF LSB CARRY ON 1ST SHIFT ON 3/4 FRAC
2871  025024  006202                        ASR     R2              ;MAKE UP 1/4 FRACTION IN R2
2872  025026  006202                        ASR     R2              ;
2873  025030  060200                        ADD     R2,R0           ;ADD 1/4 TO 1/2 FOR 3/4'S
2874  025032  011601                7$:     MOV     (SP),R1         ;GET POINT VALUE TO R1
2875  025034  005303                8$:     DEC     R3              ;NOW COUNT UNITY TIMES
2876  025036  001404                        BEQ     10$             ;BR IF ALL WHOLE UNITS ADDED IN
2877  025040  100402                        BMI     9$              ;BR IF ONLY FRACTIONAL PART EXISTS
2878  025042  061601                        ADD     (SP),R1         ;ADD IN ANOTHER UNITY
2879  025044  000773                        BR      8$              ;GO LOOK FOR ANOTHER TIMES
2880  025046  005001                9$:     CLR     R1              ;THERE IS ONLY A FRACTIONAL PART
```

```
2881  025050  060001                  10$:    ADD     R0,R1           ;ADD IN FRACTIONAL PART
2882                                  ;
2883                                  ; COMMON EXIT FROM EITHER SCALING FUNCTION ABOVE.
2884                                  ;
2885  025052  010137  001124          11$:    MOV     R1,$GDDAT       ;SAVE COMPUTED POINT.
2886  025056  012600                          MOV     (SP)+,R0        ;RESTORE R0
2887  025060  012601                          MOV     (SP)+,R1        ;RESTORE R1
2888  025062  000207                          RTS     PC              ;EXIT
2889
2890                                  ;;*************************************************************************
2891                                          ;THIS ROUTINE ADDS AN X OR Y POINT TO THE RESPECTIVE X OR
2892                                          ;Y DYNAMIC OFFSET AND LEAVES THE 14 BIT RESULT IN $GDDAT.
2893                                          ;R0 CONTAINS THE POINT DATA & R1 CONTAINS THE OFFSET DATA
2894                                          ;WHEN ENTERING THIS ROUTINE.
2895                                  ;;*************************************************************************
2896
2897  025064  032700  020000          CALPT:  BIT     #BIT13,R0       ;POINT NEG?
2898  025070  001403                          BEQ     POS1            ;BR IF NOT
2899  025072  042700  020000                  BIC     #BIT13,R0       ;CLR SIGN BIT
2900  025076  005400                          NEG     R0              ;MAKE NEG
2901  025100  032701  020000          POS1:   BIT     #BIT13,R1       ;OFFSET NEG?
2902  025104  001404                          BEQ     POS2            ;BR IF NOT
2903  025106  042701  030000                  BIC     #BIT13!BIT12,R1 ;CLR SIGN & OFFSET BITS
2904  025112  005401                          NEG     R1              ;MAKE NEG
2905  025114  000402                          BR      OVER1           ;SKIP OVER  NEXT INSTR
2906  025116  042701  010000          POS2:   BIC     #BIT12,R1       ;DONT WANT OFFSET BIT
2907  025122  060100                  OVER1:  ADD     R1,R0           ;MAKE UP CALULATED POINT VALUE
2908  025124  042700  140000                  BIC     #140000,R0      ;LIMIT TO 14 BITS
2909  025130  010037  001124                  MOV     R0,$GDDAT       ;SAVE POINT RESULT IN $GDDAT
2910  025134  000207                          RTS     PC              ;RETURN
2911
2912                                  ;;*************************************************************************
2913                                          ;THIS ROUTINE MAKES UP A 14 BIT X OR Y POINT LENGTH WHICH IS LEFT IN $BDDAT
2914                                  ;;*************************************************************************
2915  025136  042737  176000  001126  POCONV: BIC     #176000,$BDDAT  ;SAVE ONLY X OR Y POSITION BITS
2916  025144  042705  007777                  BIC     #7777,R5        ;SAVE ONLY X OR Y HI ORDER POSITION BITS
2917  025150  000241                          CLC                     ;CLR CARRY BEFORE ROTATE
2918  025152  006005                          ROR     R5              ;ROTATE RIGHT 2 PLACES
2919  025154  006005                          ROR     R5              ;
2920  025156  060537  001126                  ADD     R5,$BDDAT       ;MAKE UP 14 BIT X OR Y POINT LENGTH
2921  025162  000207                          RTS     PC              ;EXIT
```

```
 2923                                   .SBTTL  SCOPE HANDLER ROUTINE
  (1)
  (2)                                   ;;**********************************************************************
  (1)                                   ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
  (1)                                   ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
  (1)                                   ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
  (1)                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
  (1)                                   ;*SW14=1        LOOP ON TEST
  (1)                                   ;*SW11=1        INHIBIT ITERATIONS
  (1)                                   ;*SW09=1        LOOP ON ERROR
  (1)                                   ;*SW08=1        LOOP ON TEST IN SWR<7:0>
  (1)                                   ;*CALL
  (1)                                   ;*      SCOPE           ;;SCOPE=IOT
  (1)
  (1)  025164                           $SCOPE:
  (1)  025164  032777  040000  153746   1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
  (1)  025172  001114                           BNE     $OVER           ;;YES IF SW14=1
  (1)                                   ;#####START OF CODE FOR THE XOR TESTER#####
  (1)  025174  000416                   $XTSTR: BR      6$              ;;IF RUNNING ON THE ''XOR'' TESTER CHANGE
  (1)                                                                   ;;THIS INSTRUCTION TO A ''NOP'' (NOP=240)
  (1)  025176  013746  000004                   MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
  (1)  025202  012737  025222  000004           MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
  (1)  025210  005737  177060                   TST     @#177060        ;;TIME OUT ON XOR?
  (1)  025214  012637  000004                   MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
  (1)  025220  000463                           BR      $SVLAD          ;;GO TO THE NEXT TEST
  (1)  025222  022626                   5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
  (1)  025224  012637  000004                   MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
  (1)  025230  000423                           BR      7$              ;;LOOP ON THE PRESENT TEST
  (1)  025232                           6$:;#####END OF CODE FOR THE XOR TESTER#####
  (1)  025232  032777  000400  153700           BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
  (1)  025240  001404                           BEQ     2$              ;;BR IF NO
  (1)  025242  127737  153672  001102           CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
  (1)  025250  001465                           BEQ     $OVER           ;;BR IF YES
  (1)  025252  105737  001103           2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
  (1)  025256  001421                           BEQ     3$              ;;BR IF NO
  (1)  025260  123737  001115  001103           CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
  (1)  025266  101015                           BHI     3$              ;;BR IF NO
  (1)  025270  032777  001000  153642           BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
  (1)  025276  001404                           BEQ     4$              ;;BR IF NO
  (1)  025300  013737  001110  001106   7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
  (1)  025306  000446                           BR      $OVER
  (1)  025310  105037  001103           4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
  (1)  025314  005037  001166                   CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
  (1)  025320  000415                           BR      1$              ;;ESCAPE TO THE NEXT TEST
  (1)  025322  032777  004000  153610   3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
  (1)  025330  001011                           BNE     1$              ;;BR IF YES
  (1)  025332  005737  001204                   TST     $PASS           ;;IF FIRST PASS OF PROGRAM
  (1)  025336  001406                           BEQ     1$              ;;      INHIBIT ITERATIONS
  (1)  025340  005237  001104                   INC     $ICNT           ;;INCREMENT ITERATION COUNT
  (1)  025344  023737  001166  001104           CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
  (1)  025352  002024                           BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
  (1)  025354  012737  000001  001104   1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
  (1)  025362  013737  025440  001166           MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
  (1)  025370  105237  001102           $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
  (1)  025374  113737  001102  001202           MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
  (1)  025402  011637  001106                   MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
```

```
(1)  025406  011637  001110                 MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
(1)  025412  005037  001170                 CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)  025416  112737  000001  001115         MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1)  025424  013777  001102  153510 $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(1)  025432  013716  001106                 MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  025436  000002                         RTI                     ;;FIXES PS
(1)  025440  003720                 $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
2924                                .SBTTL  APT COMMUNICATIONS ROUTINE
(1)
(2)                                 ;;**********************************************************************
(1)  025442  112737  000001  025706 $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
(1)  025450  112737  000001  025704 $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
(1)  025456  000403                         BR      $ATYC
(1)  025460  112737  000001  025706 $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
(1)  025466                         $ATYC:
(3)  025466  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  025470  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(1)  025472  105737  025704                 TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
(1)  025476  001450                         BEQ     5$              ;;IF NOT:  BR
(1)  025500  122737  000001  001216         CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
(1)  025506  001031                         BNE     3$              ;;IF NOT:  BR
(1)  025510  132737  000100  001217         BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(1)  025516  001425                         BEQ     3$              ;;IF NOT:  BR
(1)  025520  017600  000004                 MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
(1)  025524  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(1)  025532  005737  001176         1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
(1)  025536  001375                         BNE     1$              ;;IF NOT:  WAIT
(1)  025540  010037  001212                 MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
(1)  025544  105720                 2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
(1)  025546  001376                         BNE     2$
(1)  025550  163700  001212                 SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
(1)  025554  006200                         ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
(1)  025556  010037  001214                 MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
(1)  025562  012737  000004  001176         MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
(1)  025570  000413                         BR      5$
(1)  025572  017637  000004  025616 3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
(1)  025600  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
(3)  025606  013746  177776                 MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
(1)  025612  004737  027222                 JSR     PC,$TYPE        ;;CALL TYPE MACRO
(1)  025616  000000                 4$:     .WORD   0
(1)  025620                         5$:
(1)  025620  105737  025706         10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
(1)  025624  001416                         BEQ     12$             ;;IF NOT:  BR
(1)  025626  005737  001216                 TST     $ENV            ;;RUNNING UNDER APT?
(1)  025632  001413                         BEQ     12$             ;;IF NOT:  BR
(1)  025634  005737  001176         11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
(1)  025640  001375                         BNE     11$             ;;IF NOT:  WAIT
(1)  025642  017637  000004  001200         MOV     @4(SP),$FATAL   ;;GET ERROR #
(1)  025650  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(1)  025656  005237  001176                 INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
(1)  025662  105037  025706         12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
(1)  025666  105037  025705                 CLRB    $LFLG           ;;CLEAR LOG FLAG
(1)  025672  105037  025704                 CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
(3)  025676  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  025700  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  025702  000207                         RTS     PC              ;;RETURN
```

```
(1)  025704  000                  $MFLG:  .BYTE   0               ;;MESSG. FLAG
(1)  025705  000                  $LFLG:  .BYTE   0               ;;LOG FLAG
(1)  025706  000                  $FFLG:  .BYTE   0               ;;FATAL FLAG
(1)          025710                       .EVEN
(1)          000200               APTSIZE=200
(1)          000001               APTENV=001
(1)          000100               APTSPOOL=100
(1)          000040               APTCSUP=040
2925                              .SBTTL  ERROR HANDLER ROUTINE
(1)
(2)                               ;;**********************************************************************
(1)                               ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
(1)                               ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(1)                               ;*AND GO TO $ERRTYP ON ERROR
(1)                               ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                               ;*SW15=1        HALT ON ERROR
(1)                               ;*SW13=1        INHIBIT ERROR TYPEOUTS
(1)                               ;*SW09=1        LOOP ON ERROR
(1)                               ;*CALL
(1)                               ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
(1)
(1)  025710                       $ERROR:
(2)  025710  113737  001102  001734          MOVB    $TSTNM,TSTNUM
(1)  025716  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
(1)  025722  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(1)  025724  013777  001102  153210          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  025732  005237  001112                  INC     $ERTTL          ;;INC THE ERROR COUNT
(1)  025736  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  025742  162737  000002  001116          SUB     #2,$ERRPC
(1)  025750  117737  153142  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  025756  032777  020000  153154          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
(1)  025764  001004                          BNE     20$             ;;SKIP TYPEOUTS
(1)  025766  004737  026076                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
(1)  025772  104401  001173                  TYPE    ,$CRLF
(1)  025776                       20$:
(1)  025776  122737  000001  001216          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  026004  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
(1)  026006  113737  001114  026020          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
(1)  026014  004737  025460                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
(1)  026020    000           21$:    .BYTE   0
(1)  026021    000                   .BYTE   0
(1)  026022  000777           22$:    BR      22$             ;;APT ERROR LOOP
(1)  026024  005777  153110    2$:    TST     @SWR            ;;HALT ON ERROR
(1)  026030  100001                   BPL     3$              ;;SKIP IF CONTINUE
(1)  026032  000000                   HALT                    ;;HALT ON ERROR!
(1)  026034  032777  001000  153076    3$:    BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
(1)  026042  001402                   BEQ     4$              ;;BR IF NO
(1)  026044  013716  001110           MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
(1)  026050  005737  001170    4$:    TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
(1)  026054  001402                   BEQ     5$              ;;BR IF NONE
(1)  026056  013716  001170           MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  026062                       5$:
(1)  026062  022737  024470  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
(1)  026070  001001                          BNE     6$              ;;BRANCH IF NO
(1)  026072  000000                          HALT                    ;;YES
(1)  026074                       6$:
```

```
  (1)  026074  000002                    RTI                    ;;RETURN
2926
```

```
2928                                    .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
 (1)
 (2)                                    ;;**********************************************************************
 (1)                                    ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 (1)                                    ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
 (1)                                    ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
 (1)
 (1)  026076                            $ERRTYP:
 (1)  026076  104401  001173                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  026102  010046                            MOV     R0,-(SP)        ;;SAVE R0
 (1)  026104  005000                            CLR     R0              ;;PICKUP THE ITEM INDEX
 (1)  026106  153700  001114                    BISB    @#$ITEMB,R0
 (1)  026112  001004                            BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
 (1)                                                                    ;;TYPE THE PC OF THE ERROR
 (2)  026114  013746  001116                    MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
 (2)                                                                    ;;ERROR ADDRESS
 (2)  026120  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 (1)  026122  000426                            BR      6$              ;;GET OUT
 (1)  026124  005300                    1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
 (1)  026126  006300                            ASL     R0              ;;       WORK FOR THE ERROR TABLE
 (1)  026130  006300                            ASL     R0
 (1)  026132  006300                            ASL     R0
 (1)  026134  062700  001262                    ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
 (1)  026140  012037  026150                    MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
 (1)  026144  001404                            BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
 (1)  026146  104401                            TYPE                    ;;TYPE THE "ERROR MESSAGE"
 (1)  026150  000000                    2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
 (1)  026152  104401  001173                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  026156  012037  026166            3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
 (1)  026162  001404                            BEQ     5$              ;;SKIP TYPEOUT IF 0
 (1)  026164  104401                            TYPE                    ;;TYPE THE "DATA HEADER"
 (1)  026166  000000                    4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
 (1)  026170  104401  001173                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  026174  011000                    5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
 (1)  026176  001004                            BNE     7$              ;;GO TYPE THE DATA
 (1)  026200  012600                    6$:     MOV     (SP)+,R0        ;;RESTORE R0
 (1)  026202  104401  001173                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  026206  000207                            RTS     PC              ;;RETURN
 (1)  026210                            7$:
 (2)  026210  013046                            MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
 (2)  026212  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 (1)  026214  005710                            TST     (R0)            ;;IS THERE ANOTHER NUMBER?
 (1)  026216  001770                            BEQ     6$              ;;BR IF NO
 (1)  026220  104401  026226                    TYPE    ,8$             ;;TYPE TWO(2) SPACES
 (1)  026224  000771                            BR      7$              ;;LOOP
 (1)  026226  020040      000          8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
 (1)          026232                            .EVEN
2929
```

```
 2931                                   .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
  (1)
  (2)                                   ;;*********************************************************************
  (1)                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
  (1)                                   ;*OCTAL (ASCII) NUMBER AND TYPE IT.
  (1)                                   ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
  (1)                                   ;*CALL:
  (1)                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                                   ;*      TYPOS                   ;;CALL FOR TYPEOUT
  (1)                                   ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
  (1)                                   ;*      .BYTE   M               ;;M=1 OR 0
  (1)                                   ;*                                      ;;1=TYPE LEADING ZEROS
  (1)                                   ;*                                      ;;0=SUPPRESS LEADING ZEROS
  (1)                                   ;*
  (1)                                   ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
  (1)                                   ;*$TYPOS OR $TYPOC
  (1)                                   ;*CALL:
  (1)                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                                   ;*      TYPON                   ;;CALL FOR TYPEOUT
  (1)                                   ;*
  (1)                                   ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
  (1)                                   ;*CALL:
  (1)                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                                   ;*      TYPOC                   ;;CALL FOR TYPEOUT
  (1)
  (1)  026232  017646  000000           $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
  (1)  026236  116637  000001  026455           MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
  (1)  026244  112637  026457                   MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
  (1)  026250  062716  000002                   ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
  (1)  026254  000406                           BR      $TYPON
  (1)  026256  112737  000001  026455   $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
  (1)  026264  112737  000006  026457           MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
  (1)  026272  112737  000005  026454   $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
  (1)  026300  010346                           MOV     R3,-(SP)        ;;SAVE R3
  (1)  026302  010446                           MOV     R4,-(SP)        ;;SAVE R4
  (1)  026304  010546                           MOV     R5,-(SP)        ;;SAVE R5
  (1)  026306  113704  026457                   MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
  (1)  026312  005404                           NEG     R4
  (1)  026314  062704  000006                   ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
  (1)  026320  110437  026456                   MOVB    R4,$OMODE       ;;SAVE IT FOR USE
  (1)  026324  113704  026455                   MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
  (1)  026330  016605  000012                   MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
  (1)  026334  005003                           CLR     R3              ;;CLEAR THE OUTPUT WORD
  (1)  026336  006105                   1$:     ROL     R5              ;;ROTATE MSB INTO "C"
  (1)  026340  000404                           BR      3$              ;;GO DO MSB
  (1)  026342  006105                   2$:     ROL     R5              ;;FORM THIS DIGIT
  (1)  026344  006105                           ROL     R5
  (1)  026346  006105                           ROL     R5
  (1)  026350  010503                           MOV     R5,R3
  (1)  026352  006103                   3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
  (1)  026354  105337  026456                   DECB    $OMODE          ;;TYPE THIS DIGIT?
  (1)  026360  100016                           BPL     7$              ;;BR IF NO
  (1)  026362  042703  177770                   BIC     #177770,R3      ;;GET RID OF JUNK
  (1)  026366  001002                           BNE     4$              ;;TEST FOR 0
  (1)  026370  005704                           TST     R4              ;;SUPPRESS THIS 0?
  (1)  026372  001403                           BEQ     5$              ;;BR IF YES
```

```
(1)  026374  005204                    4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  026376  052703  000060            5$:     BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  026402  052703  000040                    BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  026406  110337  026452                    MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  026412  104401  026452                    TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  026416  105337  026454            7$:     DECB    $OCNT           ;;COUNT BY 1
(1)  026422  003347                            BGT     2$              ;;BR IF MORE TO DO
(1)  026424  002402                            BLT     6$              ;;BR IF DONE
(1)  026426  005204                            INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  026430  000744                            BR      2$              ;;GO DO THE LAST DIGIT
(1)  026432  012605                    6$:     MOV     (SP)+,R5        ;;RESTORE R5
(1)  026434  012604                            MOV     (SP)+,R4        ;;RESTORE R4
(1)  026436  012603                            MOV     (SP)+,R3        ;;RESTORE R3
(1)  026440  016666  000002  000004            MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  026446  012616                            MOV     (SP)+,(SP)
(1)  026450  000002                            RTI                     ;;RETURN
(1)  026452     000            8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  026453     000                    .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  026454     000            $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
(1)  026455     000            $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
(1)  026456  000000            $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2932                                   .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
(1)
(2)                                    ;;**************************************************************
(1)                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
(1)                                    ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
(1)                                    ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
(1)                                    ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
(1)                                    ;*REPLACED WITH SPACES.
(1)                                    ;*CALL:
(1)                                    ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
(1)                                    ;*      TYPDS                   ;;GO TO THE ROUTINE
(1)
(1)  026460                            $TYPDS:
(3)  026460  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  026462  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  026464  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  026466  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  026470  010546                            MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  026472  012746  020200                    MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1)  026476  016605  000020                    MOV     20(SP),R5       ;;GET THE INPUT NUMBER
(1)  026502  100004                            BPL     1$              ;;BR IF INPUT IS POS.
(1)  026504  005405                            NEG     R5              ;;MAKE THE BINARY NUMBER POS.
(1)  026506  112766  000055  000001            MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)  026514  005000            1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)  026516  012703  026674                    MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)  026522  112723  000040                    MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)  026526  005002            2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
(1)  026530  016001  026664                    MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)  026534  160105            3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)  026536  002402                            BLT     4$              ;;BR IF DONE
(1)  026540  005202                            INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)  026542  000774                            BR      3$
(1)  026544  060105            4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)  026546  005702                            TST     R2              ;;CHECK IF BCD DIGIT=0
(1)  026550  001002                            BNE     5$              ;;FALL THROUGH IF 0
```

```
  (1)  026552  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
  (1)  026554  100407                          BMI     7$              ;;BR IF YES
  (1)  026556  106316                  5$:     ASLB    (SP)            ;;MSD?
  (1)  026560  103003                          BCC     6$              ;;BR IF NO
  (1)  026562  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
  (1)  026570  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
  (1)  026574  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
  (1)  026600  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
  (1)  026602  005720                          TST     (R0)+           ;;JUST INCREMENTING
  (1)  026604  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
  (1)  026610  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
  (1)  026612  003002                          BGT     8$              ;;GO TO EXIT
  (1)  026614  010502                          MOV     R5,R2           ;;GET THE LSD
  (1)  026616  000764                          BR      6$              ;;GO CHANGE TO ASCII
  (1)  026620  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
  (1)  026622  100003                          BPL     9$              ;;BR IF NO
  (1)  026624  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
  (1)  026632  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
  (3)  026634  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
  (3)  026636  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
  (3)  026640  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
  (3)  026642  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
  (3)  026644  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
  (1)  026646  104401 026674                   TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
  (1)  026652  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
  (1)  026660  012616                          MOV     (SP)+,(SP)
  (1)  026662  000002                          RTI                     ;;RETURN TO USER
  (1)  026664  023420                  $DTBL:  10000.
  (1)  026666  001750                          1000.
  (1)  026670  000144                          100.
  (1)  026672  000012                          10.
  (1)  026674  000004                  $DBLK:  .BLKW   4
2933
2934                                           .SBTTL  POWER DOWN AND UP ROUTINES
  (1)
  (2)                                          ;;**********************************************************************
  (1)                                          ;POWER DOWN ROUTINE
  (1)  026704  012737  027050  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
  (1)  026712  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
  (3)  026720  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)  026722  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (3)  026724  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (3)  026726  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (3)  026730  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (3)  026732  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
  (3)  026734  017746  152200                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
  (1)  026740  010637  027054                  MOV     SP,$SAVR6       ;;SAVE SP
  (1)  026744  012737  026756  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
  (1)  026752  000000                          HALT
  (1)  026754  000776                          BR      .-2             ;;HANG UP
  (1)
  (2)                                          ;;**********************************************************************
  (1)                                          ;POWER UP ROUTINE
  (1)  026756  012737  027050  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
  (1)  026764  013706  027054                  MOV     $SAVR6,SP       ;;GET SP
  (1)  026770  005037  027054                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
```

```
(1)  026774  005237  027054          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
(1)  027000  001375                          BNE     1$              ;;OF  WORD
(3)  027002  012677  152132                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
(3)  027006  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  027010  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
(3)  027012  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  027014  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  027016  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  027020  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  027022  012737  026704  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1)  027030  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
(1)  027036  104401                          TYPE                    ;;REPORT THE POWER FAILURE
(1)  027040  027056                  $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
(1)  027042  012716                          MOV     (PC)+,(SP)      ;;RESTART AT RESTRT
(1)  027044  002460                  $PWRAD: .WORD   RESTRT          ;;RESTART ADDRESS
(1)  027046  000002                          RTI
(1)  027050  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
(1)  027052  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
(1)  027054  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
2935 027056  005015  042522  052123  PWRMSG: .ASCIZ  <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12>
2936         027124                          .EVEN
2937                                 .SBTTL  ROUTINE TO SIZE MEMORY
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*CALL:
(1)                                  ;*      JSR     PC,$SIZE
(1)                                  ;*      RETURN
(1)                                  ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
(1)
(1)  027124  010046                  $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
(1)  027126  010146                          MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
(1)  027130  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
(1)  027134  013746  000006                  MOV     @#ERRVEC+2,-(SP)
(1)  027140  010600                          MOV     SP,R0           ;;SAVE THE STACK POINTER
(1)                                  ;;SET THE ERRVEC PS TO THE PRESENT PS
(2)  027142  104400                          TRAP                    ;;PUSH OLD PSW AND PC ON STACK
(2)  027144  012637  000006                  MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
(1)  027150  012737  027170  000004          MOV     #2$,@#ERRVEC    ;;SET FOR TIMEOUT
(1)  027156  012701  020000                  MOV     #20000,R1       ;;FIRST ADDRESS
(1)  027162  005711                  1$:     TST     (R1)            ;;TEST THIS ADDRESS
(1)  027164  005721                          TST     (R1)+           ;;STEP TO NEXT ADDRESS
(1)  027166  000775                          BR      1$              ;;TRY ANOTHER
(1)  027170  162701  000002          2$:     SUB     #2,R1           ;;DROP BACK
(1)  027174  010006                          MOV     R0,SP           ;;RESTORE THE STACK
(1)  027176  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
(1)  027202  012637  000004                  MOV     (SP)+,@#ERRVEC
(1)  027206  010137  027220                  MOV     R1,$LSTAD       ;;LAST ADDRESS
(1)  027212  012601                          MOV     (SP)+,R1        ;;RESTORE R1
(1)  027214  012600                          MOV     (SP)+,R0        ;;RESTORE R0
(1)  027216  000207                          RTS     PC
(1)  027220  000000                  $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
2938
2939                                 .SBTTL  TYPE ROUTINE
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
```

```
(1)                                     ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(1)                                     ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(1)                                     ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(1)                                     ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(1)                                     ;*
(1)                                     ;*CALL:
(1)                                     ;*1) USING A TRAP INSTRUCTION
(1)                                     ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(1)                                     ;*OR
(1)                                     ;*      TYPE
(1)                                     ;*      MESADR
(1)                                     ;*
(1)
(1)  027222  105737  001157             $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(1)  027226  100002                             BPL     1$              ;;BR IF YES
(1)  027230  000000                             HALT                    ;;HALT HERE IF NO TERMINAL
(1)  027232  000430                             BR      3$              ;;LEAVE
(1)  027234  010046                     1$:     MOV     R0,-(SP)        ;;SAVE R0
(1)  027236  017600  000002                     MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(1)  027242  122737  000001  001216             CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  027250  001011                             BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
(1)  027252  132737  000100  001217             BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
(1)  027260  001405                             BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(1)  027262  010037  027272                     MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(1)  027266  004737  025450                     JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(1)  027272  000000                     61$:    .WORD   0               ;;MESSAGE ADDRESS
(1)  027274  132737  000040  001217     62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(1)  027302  001003                             BNE     60$             ;;YES,SKIP TYPE OUT
(1)  027304  112046                     2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  027306  001005                             BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  027310  005726                             TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  027312  012600                     60$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  027314  062716  000002             3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  027320  000002                             RTI                     ;;RETURN
(1)  027322  122716  000011             4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  027326  001430                             BEQ     8$
(1)  027330  122716  000200                     CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)  027334  001006                             BNE     5$
(1)  027336  005726                             TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  027340  104401                             TYPE                    ;;TYPE A CR AND LF
(1)  027342  001173                             $CRLF
(1)  027344  105037  027500                     CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)  027350  000755                             BR      2$              ;;GET NEXT CHARACTER
(1)  027352  004737  027434             5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  027356  123726  001156             6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  027362  001350                             BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  027364  013746  001154                     MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                             ;;AND THE NULL CHAR.
(1)  027370  105366  000001             7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  027374  002770                             BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  027376  004737  027434                     JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  027402  105337  027500                     DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  027406  000770                             BR      7$              ;;LOOP
(1)
(1)                                     ;HORIZONTAL TAB PROCESSOR
(1)
```

```
(1)  027410  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  027414  004737  027434          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  027420  132737  000007  027500          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  027426  001372                          BNE     9$              ;;TAB STOP
(1)  027430  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
(1)  027432  000724                          BR      2$              ;;GET NEXT CHARACTER
(1)  027434  105777  151510          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  027440  100375                          BPL     $TYPEC
(1)  027442  116677  000002  151502          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  027450  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  027456  001003                          BNE     1$              ;;BRANCH IF NO
(1)  027460  105037  027500                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  027464  000406                          BR      $TYPEX          ;;EXIT
(1)  027466  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  027474  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
(1)  027476  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  027500  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(1)  027502  000207          $TYPEX: RTS     PC
(1)
2940
2941                                  .SBTTL  INTEGER MULTIPLY ROUTINE
(1)
(2)                                   ;;**********************************************************************
(1)                                   ;*CALL
(1)                                   ;*      MOV     MULTIPLER,-(SP)
(1)                                   ;*      MOV     MULTIPLICAND,-(SP)
(1)                                   ;*      JSR     PC,@#$MULT
(1)                                   ;*      RETURN  ;;PRODUCT IS ON THE STACK
(1)                                   ;*
(1)                                   ;*      STACK   PRODUCT
(1)                                   ;*      -----   -------
(1)                                   ;*      TOP     LSB'S
(1)                                   ;*      +2      MSB'S
(1)
(1)  027504                           $MULT:
(3)  027504  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  027506  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  027510  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(1)  027512  005046                          CLR     -(SP)           ;;CLEAR THE SIGN KEY
(1)  027514  016601  000012                  MOV     12(SP),R1       ;;GET THE MULTIPLICAND
(1)  027520  100002                          BPL     1$              ;;BR IF PLUS
(1)  027522  005216                          INC     (SP)            ;;SET THE SIGN KEY
(1)  027524  005401                          NEG     R1              ;;MAKE THE MULTIPLICAND POSTIVE
(1)  027526  016602  000014          1$:     MOV     14(SP),R2       ;;GET THE MULTIPLIER
(1)  027532  100002                          BPL     2$              ;;BR IF PLUS
(1)  027534  005316                          DEC     (SP)            ;;UPDATE THE SIGN KEY
(1)  027536  005402                          NEG     R2              ;;MAKE THE MULTIPLIER POSTIVE
(1)  027540  012746  000021          2$:     MOV     #17.,-(SP)      ;;SET THE LOOP COUNT
(1)  027544  005000                          CLR     R0              ;;SETUP FOR THE MULTIPLY LOOP
(1)  027546  103001          3$:     BCC     4$              ;;DON'T ADD IF MULTIPLICAND = 0
(1)  027550  060200                          ADD     R2,R0
(1)  027552  006000          4$:     ROR     R0              ;;POSITION THE PARITIAL PRODUCT AND
(1)  027554  006001                          ROR     R1              ;;THE MULTIPLICAND
(1)  027556  005316                          DEC     (SP)            ;;HAS ALL BITS OF THE MULTIPLICAND BEEN DONE?
(1)  027560  001372                          BNE     3$              ;;BR IF NO
(1)  027562  022616                          CMP     (SP)+,(SP)      ;;SHOULD PRODUCT BE NEGATIVE?
```

```
  (1)  027564  001403                    BEQ     5$              ;;GO TO EXIT IF NO
  (1)  027566  005400                    NEG     R0              ;;YES--SO MAKE IT SO
  (1)  027570  005401                    NEG     R1
  (1)  027572  005600                    SBC     R0
  (1)  027574  005726            5$:     TST     (SP)+           ;;CLEAR SIGN INFO. OFF OF STACK
  (1)  027576  010066  000012            MOV     R0,12(SP)       ;;PUT THE PRODUCT ON THE STACK (MSB'S)
  (1)  027602  010166  000010            MOV     R1,10(SP)       ;;LSB'S
  (3)  027606  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
  (3)  027610  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
  (3)  027612  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
  (1)  027614  000207                    RTS     PC
2942
2943                             .SBTTL  TRAP DECODER
  (1)
  (2)                            ;;**************************************************************************
  (1)                            ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE ''TRAP'' INSTRUCTION
  (1)                            ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
  (1)                            ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
  (1)                            ;*GO TO THAT ROUTINE.
  (1)
  (1)  027616  010046            $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
  (1)  027620  016600  000002            MOV     2(SP),R0        ;;GET TRAP ADDRESS
  (1)  027624  005740                    TST     -(R0)           ;;BACKUP BY 2
  (1)  027626  111000                    MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
  (1)  027630  006300                    ASL     R0              ;;POSITION FOR INDEXING
  (1)  027632  016000  027652            MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
  (1)  027636  000200                    RTS     R0              ;;GO TO ROUTINE
  (1)
  (1)
  (1)                            ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO
  (1)
  (1)  027640  011646            $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
  (1)  027642  016666  000004  000002    MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
  (1)  027650  000002                    RTI                     ;;RESTORE THE PSW
  (1)
  (3)                            .SBTTL  TRAP TABLE
  (3)
  (3)                            ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
  (3)                            ;*BY THE ''TRAP'' INSTRUCTION.
  (3)
  (3)                            ;       ROUTINE
  (3)                            ;       -------
  (3)  027652  027640            $TRPAD: .WORD   $TRAP2
  (3)  027654  027222                    $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
  (3)  027656  026256                    $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
  (3)  027660  026232                    $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
  (3)  027662  026272                    $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
  (3)  027664  026460                    $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
  (1)
  (1)
2944
```

```
                                                .SBTTL  ASCII MESSAGES
027666  005015  051412  040524  HEADER: .ASCIZ  <15><12><12>/START OF CZVSBB  VS60 INSTRUCTION TEST PART II/<15><12>
027752  005015  042523  020124  ECOMSG: .ASCIZ  <15><12>/SET SWITCH REGISTER BIT 10 (002000) IF ECO VT48-0012 IS INSTALL
030061     115  044501  052116  EM1:    .ASCIZ  /MAINT. SWITCH REG/
030103     114  040517  044504  EM2:    .ASCIZ  /LOADING D.P.C. REG <STATIC>/
030137     114  040517  044504  EM3:    .ASCIZ  /LOADING RELOCATE REG/
030164  042522  042101  047111  EM4:    .ASCIZ  /READING D.P.C. WITH RELOCATE REG SET/
030231     117  043106  042523  EM5:    .ASCIZ  /OFFSET INDICATOR FAILED TO SET/
030270  043117  051506  052105  EM6:    .ASCIZ  /OFFSET POLARITY BIT FAILED TO SET/
030332  043117  051506  052105  EM7:    .ASCIZ  /OFFSET INSTR FAILED TO LOAD DYNAMIC REG/
030402  043117  051506  052105  EM10:   .ASCIZ  /OFFSET INSTR FAILED TO LOAD EXTENDED POS BITS/
030460  041523  046101  020105  EM11:   .ASCIZ  /SCALE ER/
030471     101  051502  053040  EM12:   .ASCIZ  /ABS VECTOR FAILED TO SET UP X OR Y POS REG/
030544  051107  050101  050110  EM13:   .ASCIZ  /GRAPHPLOT INC REG ER/
030571     114  040517  044504  EM14:   .ASCIZ  /LOADING X POS REG/
030613     114  040517  044504  EM15:   .ASCIZ  /LOADING Y POS REG/
030635     111  041516  042522  EM16:   .ASCIZ  /INCREMENTING DELTA X OR DELTA Y REG/
030701     104  041505  042522  EM17:   .ASCIZ  /DECREMENTING DELTA X OR DELTA Y REG/
030745     105  043504  020105  EM20:   .ASCIZ  /EDGE INDICATOR UPON EXCEEDING AN EDGE/
031013     114  040517  044504  EM21:   .ASCIZ  /LOADING CHAR REG/
031034  047111  047503  051122  EM22:   .ASCIZ  /INCORRECT OR UNEXPECTED CHANGE IN X OR Y POSITION REGS/
031123     127  047522  043516  EM23:   .ASCIZ  /WRONG DELTA X - SCALED CHAR MODE/
031164  051127  047117  020107  EM24:   .ASCIZ  /WRONG DELTA Y - SCALED CHAR MODE/
031225     127  047522  043516  EM25:   .ASCIZ  /WRONG DELTA Y - SCALED CHAR MODE ROTATED/
031276  051127  047117  020107  EM26:   .ASCIZ  /WRONG DELTA X - SCALED CHAR MODE ROTATED/
031347     107  040522  044120  EM27:   .ASCIZ  /GRAPHPLOT INC FAILED TO SET UP X OR Y POS REG/
031425     102  051501  041511  EM30:   .ASCIZ  /BASIC VECTOR FAILED TO LOAD X OR Y POS REG/
031500  040502  044523  020103  EM31:   .ASCIZ  /BASIC SHORT VECTOR FAILED TO LOAD X OR Y POS REG/
031561     122  051505  052105  EM32:   .ASCIZ  /RESET FAILED TO CLEAR A REG/
031615     105  043504  020105  EM33:   .ASCIZ  /EDGE FLAG INTR ER/
031637     111  052116  020122  EM34:   .ASCIZ  /INTR PRIORITY FAILURE USING INTERNAL STOP/
031711     123  044510  052106  EM35:   .ASCIZ  /SHIFT-OUT BIT SET IN ERROR/
031744  044123  043111  026524  EM36:   .ASCIZ  /SHIFT-OUT BIT FAILED TO SET/
032000  047111  042524  047122  EM37:   .ASCIZ  /INTERNAL STOP FLAG FAILED TO SET/
032041     047  051106  046501  EM40:   .ASCIZ  /'FRAMES PER SECOND' SYNC ER/

032075     114  047117  020107  EM41:   .ASCIZ  /LONG VECTOR FAILED TO SET UP X OR Y POS REG/
032151     120  044517  052116  EM42:   .ASCIZ  /POINT OR ABS VEC WITH AN OFFSET FAILED TO SET UP X OR Y POS/
032245     123  050125  051105  EM43:   .ASCIZ  "SUPER/SUBSCRIPT CHAR SCALE OR DELTA X OR Y ER"

032323     105  051122  041520  DH1:    .ASCIZ  /ERRPC   TSTNUM  BUSADR  EXPCT   RCVD/
032370  051105  050122  020103  DH2:    .ASCIZ  /ERRPC   TSTPC/
032406  051105  050122  020103  DH11:   .ASCIZ  /ERRPC   TSTNUM  EXPCT   RCVD    SCALE   STARTVAL/
        032470                          .EVEN
032470  001116  001734  001122  DT1:    $ERRPC,TSTNUM,$BDADR,$GDDAT,$BDDAT,0
032504  001116  001106  000000  DT2:    $ERRPC,$LPADR,0
032512  001116  001734  001124  DT11:   $ERRPC,TSTNUM,$GDDAT,$BDDAT,$TMP0,$REG0,0

                                ;;**************************************************************
                                        ;THIS IS THE WORKING AREA FOR ALL VS60 NPR'S (INSTR'S AND DATA)
                                        ;FROM HERE TO THE END OF 8K
                                ;;**************************************************************
032530  000000                  BUFFER: 0

        000001                          .END
```

```
ABASE = 172000              21#     36
ACDW1 = 000000              36
ACDW2 = 000000              36
ACPUOP= 000000              36
ADDW0 = 000000              36
ADDW1 = 000000              36
ADDW10= 000000              36
ADDW11= 000000              36
ADDW12= 000000              36
ADDW13= 000000              36
ADDW14= 000000              36
ADDW15= 000000              36
ADDW2 = 000000              36
ADDW3 = 000000              36
ADDW4 = 000000              36
ADDW5 = 000000              36
ADDW6 = 000000              36
ADDW7 = 000000              36
ADDW8 = 000000              36
ADDW9 = 000000              36
ADEVCT= 000000              36
ADEVM = 000000              36
AENV  = 000000              36
AENVM = 000000              36
AFATAL= 000000              36
AMADR1= 000000              36
AMADR2= 000000              36
AMADR3= 000000              36
AMADR4= 000000              36
AMAMS1= 000000              36
AMAMS2= 000000              36
AMAMS3= 000000              36
AMAMS4= 000000              36
AMSGAD= 000000              36
AMSGLG= 000000              36
AMSGTY= 000000              36
AMTYP1= 000000              36
AMTYP2= 000000              36
AMTYP3= 000000              36
AMTYP4= 000000              36
ANAME   002000             334#
APASS = 000000              36
APRIOR= 000000              36
APTCSU= 000040            2924#   2939
APTENV= 000001            2924#   2925    2939
APTSIZ= 000200             361    2924#
APTSPO= 000100            2924#   2939
ASWREG= 000000              36
ATESTN= 000000              36
AUNIT = 000000              36
AUSWR = 000000              36
AVECT1= 100320              22#     36
AVECT2= 000000              36
BEGIN   002050              27     360#
BEGINA  002040              29     358#    377
BEGIN1  002054             359     361#
```

```
BIT0  = 000001              19#    760    2773
BIT00 = 000001              19#
BIT01 = 000002              19#
BIT02 = 000004              19#
BIT03 = 000010              19#
BIT04 = 000020              19#
BIT05 = 000040              19#
BIT06 = 000100              19#
BIT07 = 000200              19#
BIT08 = 000400              19#   2923
BIT09 = 001000              19#   2923   2925
BIT1  = 000002              19#    428    764
BIT10 = 002000              19#
BIT11 = 004000              19#   2923
BIT12 = 010000              19#    429    440    536    540   2903   2906
BIT13 = 020000              19#    759    762    763    766    767    770    771    774   2897   2899   2901   2903
                           2925
BIT14 = 040000              19#   2923
BIT15 = 100000              19#   2613   2775   2797
BIT2  = 000004              19#    768
BIT3  = 000010              19#    772
BIT4  = 000020              19#
BIT5  = 000040              19#   1591   1619   1645   1664
BIT6  = 000100              19#   2534   2560   2603
BIT7  = 000200              19#   2785
BIT8  = 000400              19#
BIT9  = 001000              19#
BPTVEC= 000014              19#
BRLEV1  001736             295#    365*  2646   2668
BUFFER  032530              284    285    286    287    288    289    460*   461*   462   463*   465*   487*   488*
                            489*   490*   491*   498*   508*   509*   510*   511*   512*   519*   528*   529*   532*
                            533*   534*   535*   536*   539    546    564*   565*   568*   569*   570*   571*   574
                            582    602    611*   612*   613*   614*   617    625    645    660*   661    668    683*
                            684    688    714*   715*   716*   717*   725    737    738    747    748    759*   762*
                            763*   766*   767*   770*   771*   774*   776    778*   783    866    898*   967    998*
                           1067   1098*  1104   1137*  1138*  1143*  1149   1181*  1240*  1241*  1245   1277*  1283
                           1315*  1322   1348*  1350*  1351   1353*  1375   1405*  1409*  1410   1412*  1416   1445*
                           1447*  1451*  1473   1503*  1516   1563*  1686   1721   1768*  1771   1842*  1885   1912
                           1932   1953   1973   1993   2013   2033   2053   2079   2099   2119   2139   2159   2179
                           2199   2218   2257*  2261*  2264*  2271*  2275*  2321   2360*  2364*  2367*  2374*  2378*
                           2411   2437   2473   2512   2570   2625*  2626*  2627*  2628*  2642*  2664*  2695*  2696*
                           2697*  2712   2719   2731*  2732*  2733   2776   2998#
CALPT   025064              739    749   2897#
CHSZ0   001740              297#  1898   2066
CHSZ1   001742              298#  1925   2092   2448   2485
CHSZ2   001744              299#  1945   2112
CHSZ3   001746              300#  1966   2132
CONS    002002              335#
CONT    024552              499    520   1550   1590   1617   2523   2783#
CR    = 000015               19#  2939
CRLF  = 000200               19#  2939
CSCL    021102             2226   2274   2285#  2329   2377
CSCLR   021066             2229   2269   2278#  2332   2372
DBUF    001712              284#   807    838    908    940   1008   1040   1191   1217   1570   1607   1635   1654
                           1839*  1861*  2595*  2643   2665
DBUF1   001714              285#  1843*  1862*  2574*  2596*
```

```
DBUF2   001716          286#   1863*  2597*
DBUF3   001720          287#   1864*  2599*
DBUF4   001722          288#   1865*
DBUF5   001724          289#   1866*
DDISP = 177570           19#     36    361
DDONE   002020          343#    370    372   2641*  2657*  2663*  2679*  2690*
DDONE1  002022          344#   2657   2690
DH1     032323            40     47     54     61     68     75     82     89    103    110    117    124    131
                         138    145    152    159    166    173    180    187    194    201    208    215    222
                         229    236    243    257    264    270    277   2986#
DH11    032406           96   2988#
DH2     032370          250   2987#
DISPLA  001142           36#    361*  2923*  2925*
DISPRE  000174           27#    361
DNAME   002004          336#
DPC     001760          326#    366    426    430*   431    441    442*   447    462*   464*   466*   472*  1721*
                        1766*  1771*  2643*  2665*  2719*  2725*  2776*  2783*
DSAVE   001726          290#   1190*  1196   1207*  1216*  1221   1234*  1841*  1843   1854*  1855   1857   2569*
                        2574   2585*  2586   2588
DSAVE1  001730          291#
DSWR  = 177570           19#     36    361
DT1     032470            41     48     55     62     69     76     83     90    104    111    118    125    132
                         139    146    153    160    167    174    181    188    195    202    209    216    223
                         230    237    244    258    265    271    278   2990#
DT11    032512           97   2992#
DT2     032504          251   2991#
ECOMSG  027752          388   2948#
EMTVEC= 000030           19#    361*
EM1     030061           39   2949#
EM10    030402           88   2956#
EM11    030460           95   2957#
EM12    030471          102   2958#
EM13    030544          109   2959#
EM14    030571          116   2960#
EM15    030613          123   2961#
EM16    030635          130   2962#
EM17    030701          137   2963#
EM2     030103           46   2950#
EM20    030745          144   2964#
EM21    031013          151   2965#
EM22    031034          158   2966#
EM23    031123          165   2967#
EM24    031164          172   2968#
EM25    031225          179   2969#
EM26    031276          186   2970#
EM27    031347          193   2971#
EM3     030137           53   2951#
EM30    031425          200   2972#
EM31    031500          207   2973#
EM32    031561          214   2974#
EM33    031615          221   2975#
EM34    031637          228   2976#
EM35    031711          235   2977#
EM36    031744          242   2978#
EM37    032000          249   2979#
EM4     030164           60   2952#
```

```
EM40    032041              256    2980#
EM41    032075              263    2982#
EM42    032151              269    2983#
EM43    032245              276    2984#
EM5     030231               67    2953#
EM6     030270               74    2954#
EM7     030332               81    2955#
ERRVEC= 000004               19#    361*   2923*  2937*
EXEC    024542             2774    2776#
EXECRS  024560             2777    2784#
EXECUT  024524              492     513    537    572    615    654    677    699    735    792    815    846    880
                            916     947    980   1016   1047   1080   1118   1163   1200   1225   1242   1258   1296
                           1332    1389   1427   1487   1539   1580   1598   1616   1626   1643   1662   1844   1867
                           1893    1920   1940   1961   1981   2001   2021   2041   2061   2087   2107   2127   2147
                           2167    2187   2207   2233   2336   2419   2445   2481   2521   2557   2575   2601   2630
                           2698    2772#
EXSYNC  024356              358*    360*   2745   2760#
GNS   = ****** U             27    2943
HEADER  027666              386    2947#
HT    = 000011               19#   2939
IOTVEC= 000020               19#    361*
KIPAR0= ****** U            2937
LF    = 000012               19#   2939
LFSZ0   001750              302#   1986   2152
LFSZ1   001752              303#   2006   2172
LFSZ2   001754              304#   2026   2192
LFSZ3   001756              305#   2046   2212
LPVCT   002024              346#   1684*  1769*
LPVCT1  002026              347#   1685*
MULSCL  024674             1115    2824#
NAMEVT  002034              352#
NAMEV1  002036              353#
NOFLAG  024632             2522*   2600*  2629*  2790   2796*  2800#
OVER1   025122             2905    2907#
PIRQ  = 177772               19#
PIRQVE= 000240               19#
POCONV  025136               577     585    620    628    742    752   1122   1129   1749   1758   2915#
POS1    025100             2898    2901#
POS2    025116             2902    2906#
PR0   = 000000               19#
PR1   = 000040               19#
PR2   = 000100               19#
PR3   = 000140               19#
PR4   = 000200               19#
PR5   = 000240               19#
PR6   = 000300               19#
PR7   = 000340               19#
PS    = 177776               19#
PSW   = 177776               19#    389*   1683*  1722*  1765*  2511*  2594*  2623*  2640*  2648*  2662*  2668*  2680*
                           2689*   2805*
PWRMSG  027056             2934    2935#
PWRVEC= 000024               19#    361*   2934*
RESTRT  002460               28     384     389#   2767   2934
RESVEC= 000010               19#
RLO     001770              330#    401     403*   404    410*   445*   455*
RSTVEC  024634              379    1783    2805#
```

```
SETUP1  002324            362#
SETUP2  002352            368#    371
SETUP3  002402            375#    378
SIZE    001732            292#    381*    382*
SREG0   001762            327#   1569    1583    1592    1600    1606    1620    1628    1634    1646    1653    1665    1671
                         1725    1727    1732    1739    1770    1778    2466    2469    2503    2506    2533    2537    2558
                         2562    2578    2581    2604    2607    2624    2633    2636    2644    2651    2660    2675    2683
                         2722    2728    2786
SREG1   001772            331#   2234    2236    2337    2339
STACK = 001100             19#    361     390
STKLMT= 177774             19#
STKPT   002012            339#    413     416*    417     429*    440*    458*    467     468     474     484*    2694*   2775*
                         2797*
STKVAL  002006            337#
SWR     001140             36#    361*   1513    2763    2831    2923    2925    2934*
SWREG   000176             27#    361
SW0   = 000001             19#
SW00  = 000001             19#
SW01  = 000002             19#
SW02  = 000004             19#
SW03  = 000010             19#
SW04  = 000020             19#
SW05  = 000040             19#
SW06  = 000100             19#
SW07  = 000200             19#
SW08  = 000400             19#
SW09  = 001000             19#
SW1   = 000002             19#
SW10  = 002000             19#    1513    2831
SW11  = 004000             19#
SW12  = 010000             19#    2763
SW13  = 020000             19#
SW14  = 040000             19#
SW15  = 100000             19#
SW2   = 000004             19#
SW3   = 000010             19#
SW4   = 000020             19#
SW5   = 000040             19#
SW6   = 000100             19#
SW7   = 000200             19#
SW8   = 000400             19#
SW9   = 001000             19#
TBITVE= 000014             19#
TBL2R   012206           1331    1354    1357#
TBL3R   012764           1425    1448    1455#
TBL4R   015214           1717    1786#
TERMCH  002010            338#
TIMEVT  002030            349#
TKVEC = 000060             19#
TMEVT1  002032            350#
TMSAV   024346           2717    2735    2736    2740    2741    2747    2750    2755#
TPVEC = 000064             19#
TRAPVE= 000034             19#     361*
TRTVEC= 000014             19#
TSTNUM  001734            293#   2925*   2990    2992
TST1    002500            399#
```

```
TST10   003562            522     526#
TST100  022450           2490    2499    2505    2509#
TST101  023020           2531    2539    2547    2566#
TST102  023156           2583    2591#
TST103  023336           2609    2615    2622#
TST104  023466           2635    2639#
TST105  023610           2653    2659#
TST106  023776           2677    2685    2692#
TST107  024102           2704    2707#
TST11   004004            554     560#
TST110  024360           2748    2751    2753    2762#
TST12   004242            592     598#
TST13   004512            635     641#
TST14   004654            664#
TST15   005016            687#
TST16   005252            722#
TST17   005626            777     782#
TST2    002572            412#
TST20   005756            799     803#
TST21   006140            833#
TST22   006324            865#
TST23   006562            897     903#
TST24   006752            935#
TST25   007136            966#
TST26   007372            997    1003#
TST27   007562           1035#
TST3    002670            425#
TST30   007746           1066#
TST31   010202           1097    1103#
TST32   010456           1148#
TST33   010714           1180    1186#
TST34   011060           1206    1212#
TST35   011246           1232    1244#
TST36   011506           1276    1282#
TST37   011746           1314    1320#
TST4    002764            439#
TST40   012246           1352    1374#
TST41   012524           1411    1415#
TST42   013024           1452    1472#
TST43   013254           1502    1508#
TST44   013562           1561    1568#
TST45   013762           1587    1596    1602    1605#
TST46   014126           1624    1630    1633#
TST47   014232           1648    1652#
TST5    003106            457#
TST50   014370           1673    1677#
TST51   015356           1784    1836#
TST52   015530           1851    1860#
TST53   015700           1875    1880    1884#
TST54   016066           1907    1911#
TST55   016212           1927    1931#
TST56   016336           1947    1951#
TST57   016462           1968    1971#
TST6    003254            475     483#
TST60   016606           1988    1991#
TST61   016732           2008    2011#
```

```
TST62    017056            2028    2031#
TST63    017202            2048    2051#
TST64    017362            2074    2077#
TST65    017506            2094    2097#
TST66    017632            2114    2117#
TST67    017756            2134    2137#
TST7     003420             501     505#
TST70    020102            2154    2157#
TST71    020226            2174    2177#
TST72    020352            2194    2197#
TST73    020476            2214    2217#
TST74    021174            2268    2320#
TST75    021644            2371    2409#
TST76    022016            2426    2432    2436#
TST77    022230            2453    2462    2468    2472#
TYPDS  = 104405            2767    2943#
TYPE   = 104401             385     387    2767    2925    2928    2931    2932    2934    2939    2943#
TYPOC  = 104402            2928    2943#
TYPON  = 104404            2943#
TYPOS  = 104403            2943#
XDOFF    001774             332#    538     541     561*    576     599*    619     643     653*    655     700     702     741
                            794     796    1116*   1121    1748
XPORR1   021614            2375    2395#
XPOS     001764             328#    573     575     616     618     736     740     817     820     848     851     881     884
                            918     921     949     952     981     984    1018    1021    1049    1052    1081    1084    1119
                           1120    1164    1167    1213    1227    1259    1262    1304    1307    1333    1335    1390    1392
                           1428    1430    1488    1490    1540    1551    1746    1747    1869    1873    1895    1901    1922
                           1928    1942    1952    1963    2070    2138    2149    2158    2169    2178    2189    2198    2209
                           2241    2243    2344    2346    2410    2421    2446    2460    2492    2497    2541    2545
XPOSR    021114            2227    2291#
XPOSRR   021144            2272    2306#
XPOSR1   021564            2330    2381#
YDOFF    001776             333#    545     547     562*    584     600*    627     666     676*    678     707     709     751
                            798     800    1117*   1128    1757
YPORR1   021630            2376    2402#
YPOS     001766             329#    581     583     624     626     746     750     823     826     854     857     888     891
                            924     925     955     958     988     991    1024    1027    1055    1058    1088    1091    1126
                           1127    1171    1174    1187    1202    1267    1270    1297    1300    1340    1342    1396    1398
                           1435    1437    1494    1496    1544    1555    1755    1756    1837    1846    1877    1881    1903
                           1908    1948    1972    1983    1992    2003    2012    2023    2032    2043    2052    2063    2078
                           2089    2098    2109    2118    2129    2248    2250    2351    2353    2428    2433    2451    2455
                           2482    2488    2525    2529    2549    2554    2611    2616    2693    2702
YPOSR    021130            2228    2299#
YPOSRR   021160            2273    2313#
YPOSR1   021600            2331    2388#
ZDOFF    002016             341#    485     493     500     506     514     521
ZPOS     002014             340#
$APTHD   001000              34#
$ASTAT=  ****** U          2924
$ATYC    025466            2924#
$ATY1    025442            2924#
$ATY3    025450            2924#   2939
$ATY4    025460            2924#   2925
$AUTOB   001134              36#
$BASE    001252              36#    367
$BDADR   001122              36#    401*    413*    426*    441*    467*    485*    506*    538*    545*    573*    581*    616*
```

```
                    624*  643*  666*  700*  707*  736*  746*  796*  800*  820*  826*  851*  857*
                    881*  888*  921*  924*  952*  958*  981*  988*  1021* 1027* 1052* 1058* 1081*
                   1088* 1119* 1126* 1164* 1171* 1187* 1213* 1259* 1267* 1297* 1304* 1333* 1340*
                   1390* 1396* 1428* 1435* 1488* 1494* 1569* 1606* 1634* 1653* 1725* 1732* 1746*
                   1755* 1770* 1837* 1873* 1881* 1901* 1908* 1928* 1948* 1952* 1972* 1992* 2012*
                   2032* 2052* 2078* 2098* 2118* 2138* 2158* 2178* 2198* 2234* 2241* 2248* 2337*
                   2344* 2351* 2410* 2433* 2451* 2460* 2469* 2488* 2497* 2506* 2529* 2537* 2545*
                   2554* 2562* 2581* 2607* 2616* 2624* 2636* 2644* 2660* 2693* 2709* 2990
$BDDAT  001126      36#   404*  405   417*  418*  419   431*  432   447*  448   470*  476*  495*
                    502*  516*  523*  541*  542   547*  548   575*  578   583*  586   618*  621
                    626*  629   655*  656   678*  679   702*  703*  704   709*  710*  711   740*
                    743   750*  753   794*  798*  817*  818   823*  824   848*  849   854*  855
                    884*  885   891*  892   918*  919   925*  926   949*  950   955*  956   984*
                    985   991*  992   1018* 1019  1024* 1025  1049* 1050  1055* 1056  1084* 1085
                   1091*  1092  1120* 1123  1127* 1130  1167* 1168  1174* 1175  1202* 1203  1227*
                   1228*  1229  1262* 1263* 1264  1270* 1271  1300* 1301  1307* 1308* 1309  1335*
                   1336*  1337  1342* 1343* 1344  1392* 1393  1398* 1399  1430* 1431* 1432  1437*
                   1438*  1439  1490* 1491  1496* 1497  1540* 1541  1544* 1545  1551* 1552  1555*
                   1556   1583* 1584  1592* 1593  1600* 1601  1620* 1621  1628* 1629  1646* 1647
                   1665*  1666  1671* 1672* 1727* 1728* 1739* 1740* 1741  1747* 1750  1756* 1759
                   1778*  1779* 1780  1846* 1847* 1848  1869* 1871  1877* 1878* 1879  1895* 1896*
                   1899   1903* 1904* 1906  1922* 1923* 1926  1942* 1943* 1946  1963* 1964* 1967
                   1983*  1984* 1987  2003* 2004* 2007  2023* 2024* 2027  2043* 2044* 2047  2063*
                   2064*  2067  2070* 2071* 2073  2089* 2090* 2093  2109* 2110* 2113  2129* 2130*
                   2133   2149* 2150* 2153  2169* 2170* 2173  2189* 2190* 2193  2209* 2210* 2213
                   2236*  2237* 2238  2243* 2244* 2245  2250* 2251* 2252  2339* 2340* 2341  2346*
                   2347*  2348  2353* 2354* 2355  2421* 2423  2428* 2429* 2431  2446* 2449  2455*
                   2456*  2458  2466* 2467  2482* 2483* 2486  2492* 2493* 2495  2503* 2504  2525*
                   2526*  2527  2533* 2535  2541* 2543  2549* 2550* 2552  2558* 2560  2578* 2579
                   2604*  2605  2611* 2612* 2614  2633* 2634* 2651* 2675* 2683* 2702* 2703* 2711*
                   2915*  2920* 2990  2992
$CDW1   001256      36#
$CDW2   001260      36#
$CHARC  027500     2939#*
$CKSWR= ****** U   2943
$CMTAG  001100      36#   361
$CM1  = 000001      36#
$CM2  = 000002      36#
$CM3  = 000001      36#
$CM4  = 000001      36#
$CPUOP  001224      36#
$CRLF   001173      36#   2925  2928  2939
$DBLK   026674     2932#
$DEVCT  001206      36#
$DEVM   001254      36#
$DOAGN  024500     2767#
$DTBL   026664     2932#
$ENDAD  024470      31    2767#  2925
$ENDCT  024436     361    2767#
$ENDMG  024507     2767#
$ENULL  024504     2767#
$ENV    001216      36#   2924  2925  2939
$ENVM   001217      36#    361  2924  2939
$EOP    024402     2764   2767#
$EOPCT  024430     361*   2767#
$ERFLG  0011C3      36#   2923* 2925*
```

```
$ERMAX  001115            36#   361*  2923*
$ERROR  025710           361   2925#
$ERRPC  001116            36#   2925*  2928   2990   2991   2992
$ERRTB  001262            36#   2928
$ERRTY  026076          2925    2928#
$ERTTL  001112            36#   2925*
$ESCAP  001170            36#   361*  2923*  2925
$ETABL  001216            36#
$ETEND  001262            34     36#
$FATAL  001200            36#   2924*
$FFLG   025706          2924#*
$FILLC  001156            36#   2939
$FILLS  001155            36#   2939
$GDADR  001120            36#
$GDDAT  001124            36#   402*   403    405    408*   415*   416    419    422*   428*   430    432    436*
                         443*   448    451*   459*   473*   486*   497*   507*   518*   539*   540*   542    546*
                         548    574*   578    582*   586    617*   621    625*   629    644*   656    659*   667*
                         679    682*   701*   704    708*   711    743    753    791*   806*   812    813    818
                         824    829*   836*   849    855    861*   882*   885    889*   892    906*   919    926
                         931*   938*   950    956    962*   982*   985    989*   992   1006*  1019   1025   1031*
                        1038*  1050   1056   1062*  1082*  1085   1089*  1092   1123   1130   1165*  1168   1172*
                        1175   1189*  1203   1208*  1215*  1229   1235*  1260*  1264   1268*  1271   1298*  1301
                        1305*  1309   1334*  1337   1341*  1344   1391*  1393   1397*  1399   1429*  1432   1436*
                        1439   1489*  1491   1495*  1497   1538*  1541   1545   1549*  1552   1556   1582*  1591*
                        1599*  1619*  1627*  1645*  1647   1664*  1670*  1726*  1733*  1738*  1741   1745*  1750
                        1754*  1759   1777*  1780   1840*  1848   1853*  1870*  1871   1879   1897*  1898*  1899
                        1905*  1906   1924*  1925*  1926   1944*  1945*  1946   1965*  1966*  1967   1985*  1986*
                        1987   2005*  2006*  2007   2025*  2026*  2027   2045*  2046*  2047   2065*  2066*  2067
                        2072*  2073   2091*  2092*  2093   2111*  2112*  2113   2131*  2132*  2133   2151*  2152*
                        2153   2171*  2172*  2173   2191*  2192*  2193   2211*  2212*  2213   2235*  2238   2242*
                        2245   2249*  2252   2338*  2341   2345*  2348   2352*  2355   2422*  2423   2430*  2431
                        2447*  2448*  2449   2457*  2458   2484*  2485*  2486   2494*  2495   2524*  2527   2534*
                        2542*  2543   2551*  2552   2559*  2577*  2603*  2613*  2614   2632*  2645*  2661*  2701*
                        2710*  2885*  2909*  2990   2992
$GET42  024460          2767#
$GTSWR= ****** U        2943
$HD   = 000000            13
$HIBTS  001000            34#
$ICNT   001104            36#   2923*
$ILLUP  027050          2934#
$INTAG  001135            36#
$ITEMB  001114            36#   2925*  2928
$LF     001174            36#   2925   2939
$LFLG   025705          2924#*
$LPADR  001106            36#   361*   2923*  2991
$LPERR  001110            36#   361*   400*   414*   427*   446*   527*   563*   601*   642*   665*   698*   733*
                         804*   834*   879*   904*   936*   979*  1004*  1036*  1079*  1114*  1162*  1257*  1295*
                        1321*  1387*  1426*  1486*  1534*  1838*  2232*  2335*  2923*  2925
$LSTAD  027220           381   2937#*
$MADR1  001230            36#
$MADR2  001234            36#
$MADR3  001240            36#
$MADR4  001244            36#
$MAIL   001176            34     36#    361    399    412    425    439    457    483    505    526    560    598
                         641    664    687    722    782    803    833    865    903    935    966   1003   1035
                        1066   1103   1148   1186   1212   1244   1282   1320   1374   1415   1472   1508   1568
```

```
                        1605    1633    1652    1677    1836    1860    1884    1911    1931    1951    1971    1991    2011
                        2031    2051    2077    2097    2117    2137    2157    2177    2197    2217    2320    2409    2436
                        2472    2509    2566    2591    2622    2639    2659    2692    2707    2762    2923    2925    2939
$MAMS1  001226            36#
$MAMS2  001232            36#
$MAMS3  001236            36#
$MAMS4  001242            36#
$MBADR  001002            34#
$MFLG   025704          2924#*
$MSGAD  001212            36#   2924*
$MSGLG  001214            36#   2924*
$MSGTY  001176            36#   2924*
$MTYP1  001227            36#
$MTYP2  001233            36#
$MTYP3  001237            36#
$MTYP4  001243            36#
$MULT   027504          2839    2941#
$MXCNT  025440          2923#
$NULL   001154            36#   2939
$NWTST= 000001           399#    412#    425#    439#    457#    483#    505#    526#    560#    598#    641#    664#    687#
                         722#    782#    803#    833#    865#    903#    935#    966#   1003#   1035#   1066#   1103#   1148#
                        1186#   1212#   1244#   1282#   1320#   1374#   1415#   1472#   1508#   1568#   1605#   1633#   1652#
                        1677#   1836#   1860#   1884#   1911#   1931#   1951#   1971#   1991#   2011#   2031#   2051#   2077#
                        2097#   2117#   2137#   2157#   2177#   2197#   2217#   2320#   2409#   2436#   2472#   2509#   2566#
                        2591#   2622#   2639#   2659#   2692#   2707#   2762#
$OCNT   026454          2931#*
$OMODE  026456          2931#*
$OVER   025424          2923#
$PASS   001204            36#    361*   2767*   2773    2923
$PASTM  001006            34#
$PWRAD  027044          2934#
$PWRDN  026704           361    2934#
$PWRMG  027040          2934#
$PWRUP  026756          2934#
$QUES   001172            36#   2925    2939
$RDCHR= ****** U        2943
$RDDEC= ****** U        2943
$RDLIN= ****** U        2943
$RDOCT= ****** U        2943
$REGAD  001160            36#
$REGO   001162            36#    883*    890*    983*    990*   1083*   1090*   1112*   1133*   1140*   1166*   1173*   1261*
                        1269*   1299*   1306*   1388*   1485*   1536*   2992
$RTNAD  024502          2767#
$R2A  = ****** U        2943
$SAVRE= ****** U        2943
$SAVR6  027054          2934#*
$SCOPE  025164           361    2923#
$SETUP= 000037           357#    361    2767    2923    2925
$SIZE   027124           380    2937#
$STUP = 177777           357#
$SVLAD  025370          2923#
$SVPC = 000214            31#
$SWR  = 165400             5#      13      23#     25      36     361     399     412     425     439     457     483     505
                         526     560     598     641     664     687     722     782     803     833     865     903     935
                         966    1003    1035    1066    1103    1148    1186    1212    1244    1282    1320    1374    1415
                        1472    1508    1568    1605    1633    1652    1677    1836    1860    1884    1911    1931    1951
```

```
                        1971    1991    2011    2031    2051    2077    2097    2117    2137    2157    2177    2197    2217
                        2320    2409    2436    2472    2509    2566    2591    2622    2639    2659    2692    2707    2762
                        2767    2923    2925    2934
$SWREG  001220           36#     361
$SWRMK= 000000           25     2923
$TESTN  001202           36#     399*    412*    425*    439*    457*    483*    505*    526*    560*    598*    641*    664*
                        687*    722*    782*    803*    833*    865*    903*    935*    966*   1003*   1035*   1066*   1103*
                       1148*   1186*   1212*   1244*   1282*   1320*   1374*   1415*   1472*   1508*   1568*   1605*   1633*
                       1652*   1677*   1836*   1860*   1884*   1911*   1931*   1951*   1971*   1991*   2011*   2031*   2051*
                       2077*   2097*   2117*   2137*   2157*   2177*   2197*   2217*   2320*   2409*   2436*   2472*   2509*
                       2566*   2591*   2622*   2639*   2659*   2692*   2707*   2762*   2923*
$TIMES  001166           36#     361*    439*    782*    803*    833*    903*    935*   1003*   1035*   1103*   1186*   1212*
                       1652*   1677*   1836*   2509*   2566*   2622*   2692*   2707*   2767*   2923*
$TKB    001146           36#
$TKS    001144           36#
$TMP0   001164           36#     876*    895*    896     978*    995*    996    1078*   1095*   1096    1110*   1142*   1159*
                       1178*   1179    1256*   1274*   1275    1294*   1312*   1313    1386*   1402*   1403    1484*   1500*
                       1501    1535*   1560    1562*   2992
$TN   = 000111            6#     13     399#    412#    425#    439#    457#    475     483#    501     505#    522     526#
                        554     560#    592     598#    635     641#    664#    687#    722#    777     782#    799     803#
                        833#    865#    897     903#    935#    966#    997    1003#   1035#   1066#   1097   1103#   1148#
                       1180    1186#   1206    1212#   1232    1244#   1276    1282#   1314    1320#   1352    1374#   1411
                       1415#   1452    1472#   1502    1508#   1561    1568#   1587    1596    1602    1605#   1624    1630
                       1633#   1648    1652#   1673    1677#   1784    1836#   1851    1860#   1875    1880    1884#   1907
                       1911#   1927    1931#   1947    1951#   1968    1971#   1988    1991#   2008    2011#   2028    2031#
                       2048    2051#   2074    2077#   2094    2097#   2114    2117#   2134    2137#   2154    2157#   2174
                       2177#   2194    2197#   2214    2217#   2268    2320#   2371    2409#   2426    2432    2436#   2453
                       2462    2468    2472#   2490    2499    2505    2509#   2531    2539    2547    2566#   2583    2591#
                       2609    2615    2622#   2635    2639#   2653    2659#   2677    2685    2692#   2704    2707#   2748
                       2751    2753    2762#
$TPB    001152           36#    2939*
$TPFLG  001157           36#    2939
$TPS    001150           36#    2939
$TRAP   027616          361     2943#
$TRAP2  027640         2943#
$TRP  = 000006         2943#
$TRPAD  027652         2943#
$TSTM   001004           34#
$TSTNM  001102           36#     392*    2767*   2923*   2925
$TYPBN= ****** U       2943
$TYPDS  026460         2932#    2943
$TYPE   027222         2924     2939#   2943
$TYPEC  027434         2939#
$TYPEX  027502         2939#
$TYPOC  026256         2931#    2943
$TYPON  026272         2931#    2943
$TYPOS  026232         2931#    2943
$UNIT   001210           36#
$UNITM  001010           34#
$USWR   001222           36#
$VECT1  001246           36#     362     373     2807
$VECT2  001250           36#
$XTSTR  025174         2923#
$$GET4= 000000         2767#
$OFILL  026455         2931#*
$40CAT= ****** U       2923     2925
```

```
.      = 032532              27#     31#     33#     34#     36#     361     1514    1542    1546    1553    1557    2767    2923
                            2924#   2925    2928#   2932#   2934    2936#   2939    2989#
.$ASTA= ****** U             2924
.$X    = 001000               34#
```

```
COMMEN     19#
CONT      316#    1590    1617    2523
ENDCOM     19#
ERROR      19#     407     421     434     450     471     477     496     503     517     524     544     550     580     588
          623     631     658     681     706     713     745     755     797     801     821     827     852     858     887
          894     922     928     953     959     987     994    1022    1028    1053    1059    1087    1094    1125    1132
         1170    1177    1205    1231    1266    1273    1303    1311    1339    1346    1395    1401    1434    1441    1493
         1499    1543    1547    1554    1558    1586    1595    1603    1623    1631    1649    1668    1674    1729    1743
         1752    1761    1782    1850    1874    1882    1902    1909    1929    1949    1969    1989    2009    2029    2049
         2069    2075    2095    2115    2135    2155    2175    2195    2215    2240    2247    2254    2343    2350    2357
         2425    2434    2452    2461    2470    2489    2498    2507    2530    2538    2546    2555    2563    2582    2608
         2617    2637    2652    2676    2684    2705    2739    2744    2749    2752    2793
ESCAPE     19#
EXECUT    313#     492     513     537     699     735     815     846     916     947    1016    1047    1200    1225    1242
         1580    1598    1616    1626    1643    1662    1844    1867    1893    1920    1940    1961    1981    2001    2021
         2041    2061    2087    2107    2127    2147    2167    2187    2207    2233    2336    2419    2445    2481    2521
         2557    2575    2601    2698
GETPRI     19#    2937
GETSWR     19#
MAINT1    307#     429
MAINT2    310#
MULT       19#
NEWTST     19#     399     412     425     439     457     483     505     526     560     598     641     664     687     722
          782     803     833     865     903     935     966    1003    1035    1066    1103    1148    1186    1212    1244
         1282    1320    1374    1415    1472    1508    1568    1605    1633    1652    1677    1836    1860    1884    1911
         1931    1951    1971    1991    2011    2031    2051    2077    2097    2117    2137    2157    2177    2197    2217
         2320    2409    2436    2472    2509    2566    2591    2622    2639    2659    2692    2707    2762
POP        19#    2924    2932    2934    2941
PUSH       19#    2924    2932    2934    2941
REPORT     19#
RESUME    319#
SCOPE      19#     399     412     425     439     457     483     505     526     560     598     641     664     687     722
          782     803     833     865     903     935     966    1003    1035    1066    1103    1148    1186    1212    1244
         1282    1320    1374    1415    1472    1508    1568    1605    1633    1652    1677    1836    1860    1884    1911
         1931    1951    1971    1991    2011    2031    2051    2077    2097    2117    2137    2157    2177    2197    2217
         2320    2409    2436    2472    2509    2566    2591    2622    2639    2659    2692    2707    2762    2767
SETPRI     19#
SETTRA    2943#
SETUP      19#    361
SKIP       19#     433     475     501     522     554     592     635     777     799     819     831     863     897     933
          964     997    1033    1064    1097    1180    1206    1210    1232    1237    1276    1314    1352    1411    1452
         1502    1561    1587    1596    1602    1624    1630    1648    1673    1784    1849    1851    1875    1880    1900
         1907    1927    1947    1968    1988    2008    2028    2048    2068    2074    2094    2114    2134    2154    2174
         2194    2214    2268    2371    2426    2432    2453    2462    2468    2490    2499    2505    2531    2539    2547
         2553    2561    2583    2587    2589    2609    2615    2635    2653    2677    2685    2704    2748    2751    2753
SLASH      19#
SPACE      19#
STARS      19#      31      34      36     399     412     425     439     457     483     505     526     560     598     641
          664     687     722     782     803     833     865     903     935     966    1003    1035    1066    1103    1148
         1186    1212    1244    1282    1320    1374    1415    1472    1508    1568    1605    1633    1652    1677    1836
         1860    1884    1911    1931    1951    1971    1991    2011    2031    2051    2077    2097    2117    2137    2157
         2177    2197    2217    2320    2409    2436    2472    2509    2566    2591    2622    2639    2659    2692    2707
         2762    2767    2769    2771    2779    2781    2802    2804    2816    2823    2890    2895    2912    2914    2923
         2924    2925    2928    2931    2932    2934    2937    2939    2941    2943    2994    2997
SWRSU      19#    361#
TRMTRP    2943#
```

```
TYPBIN      19#
TYPDEC      19#    2767
TYPNAM      19#
TYPNUM      19#
TYPOCS      19#
TYPOCT      19#    2928
TYPTXT      19#
$$CMRE      36#
$$CMTM      36#
$$ESCA      19#
$$NEWT      19#     399     412     425     439     457     483     505     526     560     598     641     664     687     722
             782     803     833     865     903     935     966    1003    1035    1066    1103    1148    1186    1212    1244
            1282    1320    1374    1415    1472    1508    1568    1605    1633    1652    1677    1836    1860    1884    1911
            1931    1951    1971    1991    2011    2031    2051    2077    2097    2117    2137    2157    2177    2197    2217
            2320    2409    2436    2472    2509    2566    2591    2622    2639    2659    2692    2707    2762
$$SET       2943#
$$SETM       361#
$$SKIP       19#     475     501     522     554     592     635     777     799     897     997    1097    1180    1206    1232
            1276    1314    1352    1411    1452    1502    1561    1587    1596    1602    1624    1630    1648    1673    1784
            1851    1875    1880    1907    1927    1947    1968    1988    2008    2028    2048    2074    2094    2114    2134
            2154    2174    2194    2214    2268    2371    2426    2432    2453    2462    2468    2490    2499    2505    2531
            2539    2547    2583    2609    2615    2635    2653    2677    2685    2704    2748    2751    2753
.EQUAT        7#      19
.HEADE        7#      13
.KT11         9#
.SETUP        8#     357
.SWRHI        9#      25
.SWRLO       25#
.$ACT1       10#      31
.$APTB       10#     36#
.$APTH       10#      34
.$APTY       10#    2924
.$CATC        7#      27
.$CMTA        7#      36
.$EOP         7#    2767
.$ERRO        7#    2925
.$ERRT        9#    2928
.$MULT        9#    2941
.$PARM        8#
.$POWE        8#    2934
.$READ        8#
.$SAVE        8#
.$SCOP        8#    2923
.$SIZE        9#    2937
.$SPAC        8#
.$SWDO        8#
.$TRAP        8#    2943
.$TYPD        9#    2932
.$TYPE        7#      8#    2939
.$TYPO        7#    2931

. ABS.  032532     000     CON   RO    ABS   GBL   I

 ERRORS DETECTED:  0
```

```
 CZVSBB,CZVSBB/CRF=CZVSBB
 RUN-TIME:  33 20 1 SECONDS
 RUN-TIME RATIO: 132/55=2.3
 CORE USED:  26K  (51 PAGES)
```