VS60

INSTRUCTION TEST PART 3

MD-11-DZVSC-B

EP-DZVSC-B-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN USA

```
                                        .REM    %

                                                          IDENTIFICATION
                                                          --------------

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

                                                     PRODUCT NAME:    VS-60 INSTRUCTION TEST PART III

                                                     DATE:            AUG. 21, 1976

                                                     MAINTAINER:      DIAGNOSTIC ENGINEERING
```

0.0 TABLE OF CONTENTS

```
1.0     ABSTRACT
        --------

        THIS PROGRAM IS ONE OF A SERIES USED TO TEST STRICTLY LOGIC FUNCTIONS
        OF THE VS60 DISPLAY PROCESSOR.  IT IS THE PRIMARY TEST OF THE INTERNAL
        EIGHT LEVEL STACK AND FOUR LEVEL SILO.  THERE ARE ADDITIONAL TESTS
        WHICH CHECK THE DELTA LENGTH AND TANGENT LOGIC ALONG WITH A MEMORY
        TEST CONCERNED WITH OPERATING THE VS60 ABOVE 28K OF MEMORY IF AVAILABLE.

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
        D.  KT11 FOR MEMORY TEST ONLY

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, INDICATE
        THE MEMORY SIZE IN THE SYSTEM, INITIALIZE THE SYSTEM, AND BEGIN
        TESTING.

4.2     RESTART ADDRES 204

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

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          BELL ON ERROR
```

```
141                                                 SW09=1          LOOP ON ERROR
142                                                 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 OR FROM THE ERROR TABLE ($ERRTB:) TOWARD THE FRONT
        OF THE LISTING.

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
        STK SEL    INDICATES STACK LEVEL WHERE FAILURE OCCURED
        GDPC       INDICATES LOW ORDER 16 BIT ADDRESS EXPECTED (MEM TST ONLY)
        GDPC-HI    INDICATES HIGH ORDER 2 BIT ADDRESS EXPECTED (MEM TST ONLY)
        BDPC       RECEIVED LOW ORDER 16 BIT ADDRESS (MEM TST ONLY)
        BDPC-HI    RECEIVED HIGH ORDER 2 BIT ADDRESS (MEM TST ONLY)
        VECTOR     INDICATES VECTOR VALUE FROM WHICH DELTA LENGTH
                   AND TANGENT ARE CALCULATED

       *NOTE:  THESE ARE TYPICALLY TYPED.

7.0     MISCELLANEOUS
        -------------

7.1     VS60 BUS/VECTOR ADDRESS MODIFICATION

        MODIFY LOCATION '$VECT1' IF BASE VECTOR ADDRESS IS NOT 320.
        MODIFY LOCATION '$BASE' IF BASE BUS 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.

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 2 MINUTES WITH ITERATIONS ENABLED. EXECUTION TIME WILL INCREASE AS MEMORY IS INCREASED ABOVE 8K. AN 'END PASS' MESSAGE IS TYPED EACH PASS THRU THE DIAGNOSTIC.

9.0 PROGRAM TEST DESCRIPTIONS

9.1 STACK POINTER TESTS

STACK POINTER TESTS INCLUDE:
DECREMENT (JSR), INCREMENT (POP), STACK OVERFLOW (GREATER THAN 8 JSR'S WITHOUT A POP), STACK UNDERFLOW (GREATER THAN 8 POP'S WITHOUT A JSR), STACK OVERFLOW INTERRUPT AND STACK UNDERFLOW INTERRUPT.

9.2 STACK TESTS - JSR (PUSH)

THE STACK CONSISTS OF 8 LEVELS OF 64 BITS. EACH WRITEABLE BIT IS FORCED TO EACH STATE STARTING FROM THE 'TOP OF STACK' POSITION THRU 8 LEVELS TO THE BOTTOM OF THE STACK. FOR EXAMPLE THE TYPICAL FORMAT MIGHT BE AS FOLLOWS: 'TOP OF STACK' IS SET; AN INSTRUCTION IS SET UP AND EXECUTED WHICH SETS UP THE CONDITION UNDER TEST - FOR EXAMPLE LOAD THE VECTOR SCALE TO SOME VALUE; THEN A JSR (PUSH) INSTRUCTION IS EXECUTED - THIS WILL PUSH OR WRITE ON THE STACK THE VECTOR SCALE AT THE BITS ASSIGNED; THEN THE STACK POINTER IS SET (REG 32) TO POINT AT THE STACK LEVEL AND WORD WHERE THE VECTOR SCALE WAS STORED AND CAN BE READ FROM (REG 26); THEN IT IS CHECKED AGAINST WHAT SHOULD HAVE BEEN STORED AND IF OK THE TEST IS REPEATED AT THE NEXT STACK LEVEL.

9.3 STACK TESTS - POP (RESTORE)

THESE STACK TESTS INCLUDE EXACTLY THE SAME PROCEDURE AS 9.2 BUT INSTEAD OF THE CONDITION UNDER TEST BEING VERIFIED ON THE STACK, IT IS RESTORED VIA A POP INSTRUCTION AND VERIFIED FROM THE BUS. NOTE THAT BEFORE THE POP IS PERFORMED THE COMPLEMENT OF WHAT WAS WRITTEN ON THE STACK IS SENT TO THE FLOP/S (CONDITION) UNDER TEST. THERE ARE SEVERAL CONDITIONS (I.E. STOP INTERRUPT ENABLE, L.P. HIT DISABLE, ETC.) THAT CANNOT BE VERIFIED FROM THE BUS, IN THIS CASE AN ADDITIONAL JSR IS EXECUTED AND THE CONDITION IS CHECKED ON THE STACK.

9.4 STACK TESTS ON DPC BETWEEN 8K & 28K

THESE TWO DPC STACK TESTS ARE IDENTICAL TO THE BASIC JSR & POP DPC TESTS EXCEPT THAT THEY ARE PERFORMED AT THE HIGHEST MEMORY BOUNDRY BELOW 28K. THE INTENTION IS TO STACK PREVIOUSLY UNUSED HIGH ORDER DPC BITS. AT THE START OF THE PROGRAM A SIZER PROGRAM IS USED TO DETERMINE HIGHEST AVAILABLE MEMORY TO 28K FOR THESE TESTS. ON 8K SYSTEMS THESE TESTS ARE ABORTED.

```
9.5     UPPER MEMORY ADDRESS TEST - ABOVE 28K

        THIS TEST REQUIRES THAT THE KT11 BE PRESENT OR THE TEST WILL
        BE ABORTED.  THIS TEST WAS INTENDED TO EXERCISE DPC BITS 16 & 17
        AT 4K MEMORY INTERVALS ABOVE 28K.  THE TEST SEQUENCE FROM EACH 4K
        BLOCK IS A JSR, POP & LOAD STATUS B INSTRUCTION.  THE GRAPH
        PLOT INCREMENT PORTION OF THE LOAD STATUS B INSTRUCTION CONTAINS
        A PAGE ADDRESS EQUAL TO ACTUAL 4K MEMORY ADDRESS ABOVE 28K BEING
        ADDRESSED.  REFER TO TEST FOR DETAILED EXPLAINATION.

9.6     DELTA LENGTH TESTS

        WITH MAINTENANCE MODE SWITCH 3 SET WHEN EXECUTING A LONG VECTOR
        INSTRUCTION THE X POSITION REGISTER WILL CONTAIN THE DELTA LENGTH.
        THE DELTA LENGTH IS THE AMOUNT OF CHANGE FROM THE POINT OF ORIGIN TO
        THE EDGE OF THE SCREEN NOT TO EXCEED 1777.  THESE TESTS WILL VERIFY
        THAT THE CORRECT DELTA LENGTH FROM THE POINT OF ORIGIN CAN BE
        CALCULATED WITH DIFFERENT VALUES OF X OR Y.

9.7     TANGENT TESTS

        WITH MAINTENANCE MODE SWITCH 3 SET WHEN EXECUTING A LONG VECTOR
        INSTRUCTION THE Y POSITION REGISTER WILL CONTAIN THE TANGENT.
        THE TANGENT IS THE OPPOSITE DIVIDED BY THE ADJACENT.
        THESE TESTS WILL VERIFY THE TANGENT VALUE FOR DIFFERENT
        VALUES OF X OR Y AT A VECTOR SCALE OF UNITY.

9.8     SILO TESTS

        THE SILO TESTS DETERMINE THAT EACH LEVEL OF THE SILO CAN RESTORE
        THE SAVED STATUS WHEN A LIGHT PEN FLAG INTERRUPT IS FORCED (USING
        MAINTENTANCE MODE 3).  THE ACTUAL TEST SEQUENCE MIGHT LOOK LIKE THIS;
        SET UP THE STATUS TO SOME UNIQUE VALUE, DO A GRAPHIC DISPLAY
        INSTRUCTION (THIS FORCES STATUS TO 1ST LEVEL OF SILO), THEN
        COMPLEMENT STATUS, THEN CAUSE A LIGHT PEN FLAG INTERRUPT (THIS
        WILL RESTORE THE STATUS FROM THE 1ST LEVEL OF THE SILO), THEN
        THE PROGRAM WILL VERIFY THAT THE STATUS WAS RESTORED PROPERLY.
        SINCE THE SILO ADDRESS POINTER IS ALWAYS RESET ON DISPLAY STARTS,
        THEN THE ABOVE SEQUENCE MUST BE REPEATED UP TO FOUR TIMES (FOUR
        LEVELS IN SILO) FOR ALL LEVELS TO BE TESTED.  THE ONLY DIFFERENCE
        WILL BE THAT THE LIGHT PEN INTERRUPT WILL BE FORCED AFTER THE 2ND,
        3RD, OR 4TH GRAPHIC INSTRUCTION DEPENDING ON THE SILO LEVEL
        UNDER TEST (EACH GRAPHIC INSTRUCTION LOADS THE SILO AND BUMPS
        THE SILO ADDRESS POINTER).  AN ADDITIONAL STEP IS TAKEN IN SILO TEST #2
        IN THE CASE WHERE THE SILO RESTORED STATUS CANNOT BE READ DIRECTLY
        FROM THE BUS (L.P. HIT DISABLE, STOP INTR ENA, ECT.).  TO VERIFY THESE
        RESTORED CONDITIONS, A JSR IS PREFORMED WHICH SAVES THIS INFORMATION
        ON THE STACK WHERE IT IS ACCESSIBLE FROM THE BUS.

10.0    LISTING
        -------

        %
.TITLE  MAINDEC-11-DZVSC-B  VS60 INSTRUCTION TEST PART III
;*COPYRIGHT (C) 1976
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
```

```
                        ;*
                        ;*PROGRAM BY R. MOORE
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-CO),MAR 21, 1976.
                        ;*
                        .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            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            LOOP ON TEST IN SWR<7:0>
                        .SBTTL  BASIC DEFINITIONS

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

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

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

                        ;*PRIORITY LEVEL DEFINITIONS
        000000          PR0=    0               ;;PRIORITY LEVEL 0
        000040          PR1=    40              ;;PRIORITY LEVEL 1
        000100          PR2=    100             ;;PRIORITY LEVEL 2
        000140          PR3=    140             ;;PRIORITY LEVEL 3
        000200          PR4=    200             ;;PRIORITY LEVEL 4
        000240          PR5=    240             ;;PRIORITY LEVEL 5
```

```
        000300          PR6=    300             ;;PRIORITY LEVEL 6
        000340          PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
        100000          SW15=   100000
        040000          SW14=   40000
        020000          SW13=   20000
        010000          SW12=   10000
        004000          SW11=   4000
        002000          SW10=   2000
        001000          SW09=   1000
        000400          SW08=   400
        000200          SW07=   200
        000100          SW06=   100
        000040          SW05=   40
        000020          SW04=   20
        000010          SW03=   10
        000004          SW02=   4
        000002          SW01=   2
        000001          SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
        100000          BIT15=  100000
        040000          BIT14=  40000
        020000          BIT13=  20000
        010000          BIT12=  10000
        004000          BIT11=  4000
        002000          BIT10=  2000
        001000          BIT09=  1000
        000400          BIT08=  400
        000200          BIT07=  200
        000100          BIT06=  100
        000040          BIT05=  40
        000020          BIT04=  20
        000010          BIT03=  10
        000004          BIT02=  4
        000002          BIT01=  2
        000001          BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
```

```
                                  .EQUIV  BIT01,BIT1
                                  .EQUIV  BIT00,BIT0

                                  ;*BASIC "CPU" TRAP VECTOR ADDRESSES
        000004                    ERRVEC= 4                 ;;TIME OUT AND OTHER ERRORS
        000010                    RESVEC= 10                ;;RESERVED AND ILLEGAL INSTRUCTIONS
        000014                    TBITVEC=14                ;;"T" BIT
        000014                    TRTVEC= 14                ;;TRACE TRAP
        000014                    BPTVEC= 14                ;;BREAKPOINT TRAP (BPT)
        000020                    IOTVEC= 20                ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
        000024                    PWRVEC= 24                ;;POWER FAIL
        000030                    EMTVEC= 30                ;;EMULATOR TRAP (EMT) **ERROR**
        000034                    TRAPVEC=34                ;;"TRAP" TRAP
        000060                    TKVEC=  60                ;;TTY KEYBOARD VECTOR
        000064                    TPVEC=  64                ;;TTY PRINTER VECTOR
        000240                    PIRQVEC=240               ;;PROGRAM INTERRUPT REQUEST VECTOR
        172000                            ABASE=172000
        000320                            AVECT1=320
        000004                            APRIOR=4
                                  .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  002304                    JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
                                  .SBTTL  ACT11 HOOKS

                                  ;;*************************************************************
                                  ;HOOKS REQUIRED BY ACT11
        000204                            $SVPC=.                    ;SAVE PC
        000046                            .=46
000046  024372                            $ENDAD                     ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
        000052                            .=52
000052  000000                            .WORD   0                  ;;2)SET LOC.52 TO ZERO
        000204                            .=$SVPC                    ;; RESTORE PC
        001000                            .=1000

                                  .SBTTL  APT PARAMETER BLOCK.

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

```

 001000                               $APTHD:
 001000  000000                       $HIBTS:  .WORD    0         ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
 001002  001202                       $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  000031                                .WORD    $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
                                               ;RESTART ADDRESS 204
         000204                                .=204
 000204  000137  003244                        JMP      @#RSTART           ;RESTART ADRS
```

```
                                 .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    177607   000377        $BELL:  .ASCIZ  <207><377><377>           ;;CODE FOR BELL
001176       077                 $QUES:  .ASCII  /?/                       ;;QUESTION MARK
001177       015                 $CRLF:  .ASCII  <15>                      ;;CARRIAGE RETURN
001200    000012                 $LF:    .ASCIZ  <12>                      ;;LINE FEED
                                 ;;*************************************************************
                                 .SBTTL  APT MAILBOX-ETABLE

                                 ;;*************************************************************
                                 .EVEN
001202                           $MAIL:                          ;;APT MAILBOX
001202    000000                 $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001204    000000                 $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001206    000000                 $TESTN: .WORD   ATESTN          ;;TEST NUMBER
```

```
001210 000000             $PASS:   .WORD   APASS    ;;PASS COUNT
001212 000000             $DEVCT:  .WORD   ADEVCT   ;;DEVICE COUNT
001214 000000             $UNIT:   .WORD   AUNIT    ;;I/O UNIT NUMBER
001216 000000             $MSGAD:  .WORD   AMSGAD   ;;MESSAGE ADDRESS
001220 000000             $MSGLG:  .WORD   AMSGLG   ;;MESSAGE LENGTH
001222                    $ETABLE:                  ;;APT ENVIRONMENT TABLE
001222    000             $ENV:    .BYTE   AENV     ;;ENVIRONMENT BYTE
001223    000             $ENVM:   .BYTE   AENVM    ;;ENVIRONMENT MODE BITS
001224 000000             $SWREG:  .WORD   ASWREG   ;;APT SWITCH REGISTER
001226 000000             $USWR:   .WORD   AUSWR    ;;USER SWITCHES
001230 000000             $CPUOP:  .WORD   ACPUOP   ;;CPU TYPE,OPTIONS
                          ;*                        BITS 15-11=CPU TYPE
                          ;*                                11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                          ;*                                11/70=06,PDQ=07,Q=10
                          ;*                        BIT 10=REAL TIME CLOCK
                          ;*                        BIT  9=FLOATING POINT PROCESSOR
                          ;*                        BIT  8=MEMORY MANAGEMENT
001232    000             $MAMS1:  .BYTE   AMAMS1   ;;HIGH ADDRESS,M.S. BYTE
001233    000             $MTYP1:  .BYTE   AMTYP1   ;;MEM. TYPE,BLK#1
                          ;*                        MEM.TYPE BYTE   --  (HIGH BYTE)
                          ;*                                900 NSEC CORE=001
                          ;*                                300 NSEC BIPOLAR=002
                          ;*                                500 NSEC MOS=003
001234 000000             $MADR1:  .WORD   AMADR1   ;;HIGH ADDRESS,BLK#1
                          ;*                          MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001236    000             $MAMS2:  .BYTE   AMAMS2   ;;HIGH ADDRESS,M.S. BYTE
001237    000             $MTYP2:  .BYTE   AMTYP2   ;;MEM.TYPE,BLK#2
001240 000000             $MADR2:  .WORD   AMADR2   ;;MEM.LAST ADDRESS,BLK#2
001242    000             $MAMS3:  .BYTE   AMAMS3   ;;HIGH ADDRESS,M.S.BYTE
001243    000             $MTYP3:  .BYTE   AMTYP3   ;;MEM.TYPE,BLK#3
001244 000000             $MADR3:  .WORD   AMADR3   ;;MEM.LAST ADDRESS,BLK#3
001246    000             $MAMS4:  .BYTE   AMAMS4   ;;HIGH ADDRESS,M.S.BYTE
001247    000             $MTYP4:  .BYTE   AMTYP4   ;;MEM.TYPE,BLK#4
001250 000000             $MADR4:  .WORD   AMADR4   ;;MEM.LAST ADDRESS,BLK#4
001252 000320             $VECT1:  .WORD   AVECT1   ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001254 000000             $VECT2:  .WORD   AVECT2   ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001256 172000             $BASE:   .WORD   ABASE    ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001260 000000             $DEVM:   .WORD   ADEVM    ;;DEVICE MAP
001262 000000             $CDW1:   .WORD   ACDW1    ;;CONTROLLER DESCRIPTION WORD#1
001264                    $ETEND:
                          .MEXIT
```

```
                        .SBTTL  ERROR POINTER TABLE

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

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


001264                  $ERRTB:
```

```
                                ;ERROR 1 - MAINT MODE 3 & L.P. FLAG FAILED TO CAUSE AN INTERRUPT

001264  027522                        EM1                     ;L.P. FLAG INTR FAILURE
001266  031731                        DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001270  032256                        DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001272  000000                        0                       ;ALL ARE OCTAL

                                ;ERROR 2 - SILO FAILED TO RESTORE AT BUSADRS & LEVEL INDICATED

001274  027551                        EM2                     ;SILO FAILURE AT LEVEL INDICATED
001276  032200                        DH5                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    LEVEL
001300  032272                        DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001302  000000                        0

                                ;ERROR 3 - STACK POINTER FAILED TO DECREMENT PROPERLY ON JSR INSTR

001304  027611                        EM3                     ;STK PTR DECREMENT ERROR
001306  031731                        DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001310  032256                        DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001312  000000                        0

                                ;ERROR 4 - STACK OVERFLOW BIT FAILED TO SET OR RESET

001314  027630                        EM4                     ;STACK OVERFLOW ERROR
001316  031731                        DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001320  032256                        DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001322  000000                        0

                                ;ERROR 5 - JSR RELATIVE INSTR FAILED TO PUSH DPC ONTO STACK

001324  027652                        EM5                     ;DPC STACKING ERROR
001326  031776                        DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001330  032272                        DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001332  000000                        0
```

```
                                        ;ERROR 6 - DPU NAME REG FAILED ON STACKING

 001334  027670                                 EM6                 ;DPU NAME STACKING ERROR
 001336  031776                                 DH2                 ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
 001340  032272                                 DT2                 ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
 001342  000000                                 0

                                        ;ERROR 7 - MODE BITS FAILED ON STACKING

 001344  027713                                 EM7                 ;MODE STACKING ER
 001346  031776                                 DH2                 ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
 001350  032272                                 DT2                 ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
 001352  000000                                 0

                                        ;ERROR 10 - VECTOR SCALE BITS FAILED ON STACKING

 001354  027732                                 EM10                ;VECTOR SCALE STACKING ERROR
 001356  031776                                 DH2                 ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
 001360  032272                                 DT2                 ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
 001362  000000                                 0

                                        ;ERROR 11 - CHARACTER SCALE FAILED ON STACKING

 001364  027761                                 EM11                ;CHARACTER SCALE STACKING ERROR
 001366  031776                                 DH2                 ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
 001370  032272                                 DT2                 ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
 001372  000000                                 0

                                        ;ERROR 12 - CHARACTER ROTATE FAILED ON STACKING

 001374  030013                                 EM12                ;CHARACTER ROTATE STACKING ERROR
 001376  031776                                 DH2                 ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
 001400  032272                                 DT2                 ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
 001402  000000                                 0

                                        ;ERROR 13 - INTENSITY LEVEL FAILED ON STACKING

 001404  030046                                 EM13                ;INTENSITY LEVEL STACKING ERROR
 001406  031776                                 DH2                 ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
 001410  032272                                 DT2                 ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
 001412  000000                                 0

                                        ;ERROR 14 - COLOR LEVEL FAILED ON STACKING

 001414  030100                                 EM14                ;COLOR LEVEL STACKING-ER
 001416  031776                                 DH2                 ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
 001420  032272                                 DT2                 ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
 001422  000000                                 0
```

```
                                ;ERROR 15 - ITALICS FAILED ON STACKING

001424  030126                          EM15                    ;ITALICS STACKING ER
001426  031776                          DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001430  032272                          DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001432  000000                          0

                                ;ERROR 16 - MENU FAILED ON STACKING

001434  030150                          EM16                    ;MENU STACKING ER
001436  031776                          DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001440  032272                          DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001442  000000                          0

                                ;ERROR 17 - BLINK ENA FAILED ON STACKING

001444  030167                          EM17                    ;BLINK ENA STACKING-ER
001446  031776                          DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001450  032272                          DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001452  000000                          0

                                ;ERROR 20 - STOP INT ENA FAILED ON STACKING

001454  030213                          EM20                    ;STOP INT ENA STACKING ER
001456  031776                          DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001460  032272                          DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001462  000000                          0

                                ;ERROR 21 - L.P. HIT FAILED ON STACKING

001464  030242                          EM21                    ;LP HIT STACKING ER
001466  031776                          DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001470  032272                          DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001472  000000                          0

                                ;ERROR 22 - LOAD STATUS A OR B' INSTR FAILED WHEN ENA=0 (VERIFIED ON STACK)

001474  030273                          EM22                    ;LD STATUS STKING ER
001476  031776                          DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001500  032272                          DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001502  000000                          0

                                ;ERROR 23 - EDGE INT ENA FAILED ON STACKING

001504  030323                          EM23                    ;EDGE INT ENA STACKING ER
001506  031776                          DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001510  032272                          DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001512  000000                          0
```

```
                                        ;ERROR 24 - CHARACTER STRING ESCAPE FAILED ON STACKING

001514  030352                                 EM24                    ;CHAR STRING ESCAPE STACKING ER
001516  031776                                 DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001520  032272                                 DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001522  000000                                 0

                                        ;ERROR 25 - DEPTH QUEING OR DEPTH QUE CONTROL FAILED ON STACKING

001524  030407                                 EM25                    ;DEPTH QUE STACKING ER
001526  031776                                 DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001530  032272                                 DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001532  000000                                 0

                                        ;ERROR 26 - LINE TYPE FAILED ON STACKING

001534  030433                                 EM26                    ;LINE TYPE STACKING ER
001536  031776                                 DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001540  032272                                 DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001542  000000                                 0

                                        ;ERROR 27 - INTENSITY ENABLED FAILED ON STACKING

001544  030457                                 EM27                    ;INTENSITY ENA STKING ER
001546  031776                                 DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001550  032272                                 DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001552  000000                                 0

                                        ;ERROR 30 - L.P. INTR ENABLED FAILED ON STACKING

001554  030507                                 EM30                    ;L.P. INTR ENA STKING ER
001556  031776                                 DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001560  032272                                 DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001562  000000                                 0

                                        ;ERROR 31 - L.P. SWITCH INTR ENABLED FAILED ON STACKING

001564  030537                                 EM31                    ;L.P. SW INTR ENA STKING ER
001566  031776                                 DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001570  032272                                 DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001572  000000                                 0


                                        ;ERROR 32 - SHIFT OUT FAILED ON STACKING

001574  030572                                 EM32                    ;SHIFT OUT STKING ER
001576  031776                                 DH2                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD    STK SEL
001600  032272                                 DT2                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT  $TMP0
001602  000000                                 0
```

```
                                        ;ERROR 33 - POP INSTR FAILED TO INCREMENT THE STACK POINTER

001604  030616                                  EM33                    ;STK PTR INCREMENT ER
001606  031731                                  DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001610  032256                                  DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001612  000000                                  0

                                        ;ERROR 34 - STACK UNDERFLOW FAILED TO SET OR RESET

001614  030635                                  EM34                    ;STACK UNDERFLOW ER
001616  031731                                  DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001620  032256                                  DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001622  000000                                  0

                                        ;ERROR 35 - DPC FAILED TO RESTORE ON POP INSTR

001624  030656                                  EM35                    ;DPC POP ER
001626  031731                                  DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001630  032256                                  DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001632  000000                                  0

                                        ;ERROR 36 - NAME BITS FAILED TO RESTORE ON POP INSTR

001634  030671                                  EM36                    ;NAME POP ER
001636  031731                                  DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001640  032256                                  DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001642  000000                                  0

                                        ;ERROR 37 - VECTOR SCALE FAILED TO RESTORE ON POP INSTR

001644  030705                                  EM37                    ;VECTOR SCALE POP ER
001646  031731                                  DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001650  032256                                  DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001652  000000                                  0

                                        ;ERROR 40 - CHAR SCALE FAILED TO RESTORE ON POP INSTR

001654  030731                                  EM40                    ;CHAR SCALE POP ER
001656  031731                                  DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001660  032256                                  DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001662  000000                                  0

                                        ;ERROR 41 - CHAR ROTATE FAILED TO RESTORE ON POP INSTR

001664  030753                                  EM41                    ;CHAR ROTATE POP ER
001666  031731                                  DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001670  032256                                  DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001672  000000                                  0
```

```
                                        ;ERROR 42 - COLOR LEVEL FAILED TO RESTORE ON POP INSTR

001674  030776                                EM42                ;COLOR LEVEL POP ER
001676  031731                                DH1                 ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001700  032256                                DT1                 ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001702  000000                                0

                                        ;ERROR 43 - ITALICS FAILED TO RESTORE ON POP INSTR

001704  031021                                EM43                ;ITALICS POP ER
001706  031731                                DH1                 ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001710  032256                                DT1                 ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001712  000000                                0

                                        ;ERROR 44 - MENU FAILED TO RESTORE ON POP INSTR

001714  031040                                EM44                ;MENU POP ER
001716  031731                                DH1                 ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001720  032256                                DT1                 ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001722  000000                                0

                                        ;ERROR 45 - INTENSITY ENABLED FAILED TO RESTORE ON POP INSTR

001724  031054                                EM45                ;INTENSITY ENA POP ER
001726  031731                                DH1                 ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001730  032256                                DT1                 ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001732  000000                                0

                                        ;ERROR 46 - L.P. INTR ENABLED FAILED TO RESTORE ON POP INSTR

001734  031101                                EM46                ;L.P. INTR ENA POP ER
001736  031731                                DH1                 ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001740  032256                                DT1                 ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001742  000000                                0

                                        ;ERROR 47 - L.P. SWITCH INTR ENABLED FAILED TO RESTORE ON POP INSTR

001744  031126                                EM47                ;L.P. SW INTR ENA POP ER
001746  031731                                DH1                 ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001750  032256                                DT1                 ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001752  000000                                0

                                        ;ERROR 50 - LINE TYPE FAILED TO RESTORE ON POP INSTR

001754  031156                                EM50                ;LINE TYPE POP ER
001756  031731                                DH1                 ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001760  032256                                DT1                 ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001762  000000                                0
```

```
                                        ;ERROR 51 - INTENSITY LEVEL FAILED TO RESTORE ON POP INSTR

        001764  031177                          EM51                    ;INTENSITY LEVEL POP ER
        001766  031731                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
        001770  032256                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
        001772  000000                          0

                                        ;ERROR 52 - BLINK FAILED TO RESTORE ON POP INSTR

        001774  031226                          EM52                    ;BLINK POP ER
        001776  031731                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
        002000  032256                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
        002002  000000                          0

                                        ;ERROR 53 - MODE FAILED TO RESTORE ON POP INSTR

        002004  031243                          EM53                    ;MODE POP ER
        002006  031731                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
        002010  032256                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
        002012  000000                          0

                                        ;ERROR 54 - STOP INTR ENABLED FAILED TO RESTORE ON POP INSTR

        002014  031257                          EM54                    ;STOP INTR ENA POP ER
        002016  031731                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
        002020  032256                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
        002022  000000                          0

                                        ;ERROR 55 - DEPTH QUEING OR DEPTH QUE CONTROL FAILED TO RESTORE ON POP INSTR

        002024  031304                          EM55                    ;DEPTH QUE POP ER
        002026  031731                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
        002030  032256                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
        002032  000000                          0

                                        ;ERROR 56 - EDGE INTR ENABLED FAILED TO RESTORE ON POP INSTR

        002034  031325                          EM56                    ;EDGE INTR ENA POP ER
        002036  031731                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
        002040  032256                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
        002042  000000                          0

                                        ;ERROR 57 - CHAR STRING ESCAPE FAILED TO RESTORE ON POP INSTR

        002044  031352                          EM57                    ;CHAR STRING ESC POP ER
        002046  031731                          DH1                     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
        002050  032256                          DT1                     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
        002052  000000                          0
```

```
                                        ;ERROR 60 - L.P. HIT DISABLE FAILED TO RESTORE ON POP INSTR

002054  031401                                  EM60                    ;L.P. HIT DISABLE POP ER
002056  031731                                  DH1                     ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD
002060  032256                                  DT1                     ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT
002062  000000                                  0

                                        ;ERROR 61 - SHIFT OUT FAILED TO RESTORED ON POP INSTR

002064  031431                                  EM61                    ;SHIFT OUT ER
002066  031731                                  DH1                     ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD
002070  032256                                  DT1                     ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT
002072  000000                                  0

                                        ;ERROR 62 - TERMINATE CHARACTER FAILED TO POP THE STACK

002074  031452                                  EM62                    ;TERMINATE CHAR POP ER
002076  031731                                  DH1                     ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD
002100  032256                                  DT1                     ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT
002102  000000                                  0

                                        ;ERROR 63 - STACK OVERFLOW FAILED ON INTERRUPT

002104  031500                                  EM63                    ;STK OVFLO INT ER
002106  031731                                  DH1                     ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD
002110  032256                                  DT1                     ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT
002112  000000                                  0

                                        ;ERROR 64 - STACK UNDERFLOW FAILED ON INTERRUPT

002114  031521                                  EM64                    ;STK UNFLO INT ER
002116  031731                                  DH1                     ;ERRPC   TSTNUM  BUSADR  EXPCT   RCVD
002120  032256                                  DT1                     ;$ERRPC  TSTNUM  $BDADR  $GDDAT  $BDDAT
002122  000000                                  0

                                        ;ERROR 65 - INCORRECT DELTA LENGTH FOR X OR Y VECTOR INDICATED

002124  031542                                  EM65                    ;DELTA LENGTH ER
002126  032133                                  DH4                     ;ERRPC   TSTNUM  VECTOR  EXPCT   RCVD
002130  032326                                  DT4                     ;$ERRPC  TSTNUM  $TMP0   $GDDAT  $BDDAT
002132  000000                                  0

                                        ;ERROR 66 - TANGENT ERROR FOR X OR Y VECTOR INDICATED

002134  031562                                  EM66                    ;TANGENT ER
002136  032133                                  DH4                     ;ERRPC   TSTNUM  VECTOR  EXPCT   RCVD
002140  032326                                  DT4                     ;$ERRPC  TSTNUM  $TMP0   $GDDAT  $BDDAT
002142  000000                                  0
```

```
                                    ;ERROR 67 - DPC HIGH ORDER BIT 16 & 17 FAILED TO SET OR RESET

002144  031575                              EM67                        ;DPC 16-17 ER
002146  031731                              DH1                         ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
002150  032256                              DT1                         ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
002152  000000                              0

                                    ;ERROR 70 - VS60 ADDRESS FAILURE ABOVE 28K

002154  031612                              EM70                        ;ADRS ER
002156  032056                              DH3                         ;ERRPC  TSTNUM  GD-1716 GD-DPC  BD-1716 BD-DPC
002160  032310                              DT3                         ;$ERRPC TSTNUM  G1716   $GDDAT  B1716   $BDDAT
002162  000000                              0

                                    ;ERROR 71 - STOP FLAG FAILED TO SET WHEN VS60 OPERATING AT FULL SPEED

002164  031640                              EM71                        ;STOP FLAG FAILED TO SET
002166  031731                              DH1                         ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
002170  032256                              DT1                         ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
002172  000000                              0

                                    ;ERROR 72 - START FAILED TO CLR STACK BITS & SET 'TOP OF STACK'

002174  031670                              EM72                        ;START FAILED TO SET 'TOP OF STK'
002176  031731                              DH1                         ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
002200  032256                              DT1                         ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
002202  000000                              0
```

```
                                        ;MORE PROGRAM DEFINITIONS

                000000          OPEN=   0                       ;LOCATION WILL LATER BE REWRITTEN
                010000          MAINT1= 10000                   ;SINGLE STEP MAINTENANCE BIT - NO NPR ALLOWED
                020000          MAINT2= 20000                   ;SINGLE CYCLE MAINTENANCE BIT - ONE NPR ALLOWED

                000250          KTERRV= 250                     ;KT TRAP ADRS
                177572          KTSR0=  177572                  ;KT STATUS REG

                                        ;MORE PROGRAM TAGS

        002204  000000          TSTNUM: OPEN                    ;CONTAINS TEST NO OF TEST WITH ERROR
        002206  000000          DLYCNT: OPEN                    ;COUNTER USED BY DELAY ROUTINE
        002210  000000          LDRSV1: OPEN                    ;LOCATION SAVES DATA OF LOC USED IN MEM TEST
        002212  000000          LDRSV2: OPEN                    ;DITTO
        002214  000000          MEMMAX: OPEN                    ;CONTAINS STARTING ADRS OF HIGHEST 4K
        002216  000000          CENAB:  OPEN                    ;NON ZERO ENABLES THE DPC MEM TEST
        002220  000000          KTENAB: OPEN                    ;NON ZERO ENABLES THE ABOVE 28K DPC MEM TEST
        002222  000000          KTMAX:  OPEN                    ;CONTAINS LAST 4K PAGE ADRS AVAILABLE ABOVE 28K
        002224  000000          G1716:  OPEN                    ;CONTAINS EXPECTED ADRS BITS 17 & 16
        002226  000000          B1716:  OPEN                    ;CONTAINS ACTUAL ADRS BITS 17 & 16

                                        ;KT11 PAGE ADDRESS REGS & PAGE DESCRIPTOR REGS ADDRESSES

        002230  172340          KIPAR0: 172340                  ;PAR 0
        002232  172300          KIPDR0: 172300                  ;PDR 0
        002234  172344          KIPAR2: 172344                  ;PAR 2
        002236  172304          KIPDR2: 172304                  ;PDR 2
        002240  172356          KIPAR7: 172356                  ;PAR 7
        002242  172316          KIPDR7: 172316                  ;PDR 7

                                        ;VS60 REGISTER ADDRESS POINTERS

        002244  000000          DPC:    OPEN            ;DISPLAY PROCESSOR PC
        002246  000000          SREG0:  OPEN            ;STATUS REG 1
        002250  000000          XPOS:   OPEN            ;X POSITION & GRAPHPLOT INC REG
        002252  000000          YPOS:   OPEN            ;Y POSITION & CHAR REG
        002254  000000          RLO:    OPEN            ;RELOCATE REG
        002256  000000          SREG1:  OPEN            ;STATUS REG 2
        002260  000000          XDOFF:  OPEN            ;X DYNAMIC OFFSET REG
        002262  000000          YDOFF:  OPEN            ;Y DYNAMIC OFFSET REG
        002264  000000          ANAME:  OPEN            ;ASSOCIATE NAME REG
        002266  000000          CONS:   OPEN            ;CONSOLE INDICATORS & COLOR LEVEL
        002270  000000          DNAME:  OPEN            ;DPU NAME REG
        002272  000000          STKVAL: OPEN            ;STACK READ REG
        002274  000000          TERMCH: OPEN            ;TERMINATE CHAR REG
        002276  000000          STKPT:  OPEN            ;MAINTENANCE & STK PTR
        002300  000000          ZPOS:   OPEN            ;Z POSOITION REG
        002302  000000          ZDOFF:  OPEN            ;Z DYNAMIC OFFSET REG
```

```
                                        ;SOFTWARE INITIALIZATION ROUTINE

      002304                            START:
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
      002304  012706  001100                    MOV     #$CMTAG,R6              ;;FIRST LOCATION TO BE CLEARED
      002310  005026                            CLR     (R6)+                   ;;CLEAR MEMORY LOCATION
      002312  022706  001140                    CMP     #SWR,R6 ;;DONE?
      002316  001374                            BNE     .-6                     ;;LOOP BACK IF NO
      002320  012706  001100                    MOV     #STACK,SP               ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
      002324  012737  025072  000020            MOV     #$SCOPE,@#IOTVEC        ;;IOT VECTOR FOR SCOPE ROUTINE
      002332  012737  000340  000022            MOV     #340,@#IOTVEC+2         ;;LEVEL 7
      002340  012737  025616  000030            MOV     #$ERROR,@#EMTVEC        ;;EMT VECTOR FOR ERROR ROUTINE
      002346  012737  000340  000032            MOV     #340,@#EMTVEC+2         ;;LEVEL 7
      002354  012737  027110  000034            MOV     #$TRAP,@#TRAPVEC        ;;TRAP VECTOR FOR TRAP CALLS
      002362  012737  000340  000036            MOV     #340,@#TRAPVEC+2;LEVEL 7
      002370  012737  027144  000024            MOV     #$PWRDN,@#PWRVEC        ;;POWER FAILURE VECTOR
      002376  012737  000340  000026            MOV     #340,@#PWRVEC+2         ;;LEVEL 7
      002404  013737  024340  024332            MOV     $ENDCT,$EOPCT           ;;SETUP END-OF-PROGRAM COUNTER
      002412  005037  001166                    CLR     $TIMES                  ;;INITIALIZE NUMBER OF ITERATIONS
      002416  005037  001170                    CLR     $ESCAPE                 ;;CLEAR THE ESCAPE ON ERROR ADDRESS
      002422  112737  000001  001115            MOVB    #1,$ERMAX               ;;ALLOW ONE ERROR PER TEST
      002430  012737  002430  001106            MOV     #.,$LPADR               ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
      002436  012737  002436  001110            MOV     #.,$LPERR               ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
      002444  013746  000004                    MOV     @#ERRVEC,-(SP)          ;;SAVE ERROR VECTOR
      002450  012737  002504  000004            MOV     #64$,@#ERRVEC           ;;SET UP ERROR VECTOR
      002456  012737  177570  001140            MOV     #DSWR,SWR               ;;SETUP FOR A HARDWARE SWICH REGISTER
      002464  012737  177570  001142            MOV     #DDISP,DISPLAY          ;;AND A HARDWARE DISPLAY REGISTER
      002472  022777  177777  176440            CMP     #-1,@SWR                ;;TRY TO REFERENCE HARDWARE SWR
      002500  001012                            BNE     66$                     ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
      002502  000403                            BR      65$                     ;;BRANCH IF NO TIMEOUT
      002504  012716  002512            64$:    MOV     #65$,(SP)               ;;SET UP FOR TRAP RETURN
      002510  000002                            RTI
      002512  012737  000176  001140    65$:    MOV     #SWREG,SWR              ;;POINT TO SOFTWARE SWR
      002520  012737  000174  001142            MOV     #DISPREG,DISPLAY
      002526  012637  000004            66$:    MOV     (SP)+,@#ERRVEC          ;;RESTORE ERROR VECTOR

      002532  005037  001210                    CLR     $PASS                   ;;CLEAR PASS COUNT
      002536  132737  000200  001223            BITB    #APTSIZE,$ENVM          ;;TEST USER SIZE UNDER APT
      002544  001403                            BEQ     67$                     ;;YES,USE NON-APT SWITCH
      002546  012737  001224  001140            MOV     #$SWREG,SWR             ;;NO,USE APT SWITCH REGISTER
      002554                            67$:
      002554  005737  000042                    TST     @#42                    ;TEST IF RUNNING CHAIN MODE UNDER XXDP
      002560  001002                            BNE     START1                  ;BR IF SO
      002562  104400  027363                    TYPE    ,MSG1                   ;GO IDENTIFY TEST
      002566  013700  001256            START1: MOV     $BASE,R0                ;GET 1ST VS60 BUS ADRS
      002572  012701  002244                    MOV     #DPC,R1                 ;INITIALIZE VS60 REG POINTER
      002576  010011                    SETPTR: MOV     R0,(R1)                 ;SET UP THIS ADRS POINTER
      002600  062700  000002                    ADD     #2,R0                   ;ADVANCE TO NEXT REG ADRS
      002604  062701  000002                    ADD     #2,R1                   ;MOVE POINTER TO NEXT REG ADRS
      002610  020127  002304                    CMP     R1,#DPC+40              ;HAVE WE DONE ALL REG POINTERS?
      002614  001370                            BNE     SETPTR                  ;BR IF NOT
```

```
002616  005037  002216          CSIZR:  CLR     CENAB           ;TURN OFF DPC MEM TEST
002622  005037  002220                  CLR     KTENAB          ;TURN OFF KT UPPER MEM TEST
002626  012737  002666  000004          MOV     #2$,@#ERRVEC    ;SET UP MEM TIMEOUT SERVICE ADRS
002634  012700  040000                  MOV     #40000,R0       ;START MEM SIZER GREATER THAN 8K
002640  010001                  1$:     MOV     R0,R1           ;DO ACTUAL TESTING IN R1
002642  005721                          TST     (R1)+           ;CHECK EVEN ADRS
002644  005711                          TST     (R1)            ;CHECK ODD ADRS
002646  062700  020000                  ADD     #20000,R0       ;BUMP IT BY 4K
002652  022700  160000                  CMP     #160000,R0      ;ARE WE TO I/O PAGE?
002656  001370                          BNE     1$              ;BR IF NOT
002660  005237  002220                  INC     KTENAB          ;'KTENAB' NON-ZERO MEANS KT11 SIZER REQ
002664  000401                          BR      3$              ;GO SAVE 28K MEM LIMIT
002666  022626                  2$:     CMP     (R6)+,(R6)+     ;FIX STACK SINCE NO RTI ON MEM TIMEOUT
002670  012737  000006  000004  3$:     MOV     #ERRVEC+2,@#ERRVEC      ;RESTORE LOC 4 & 6
002676  162700  020000                  SUB     #20000,R0       ;BACK UP TO LAST 4K MEM BLK
002702  010037  002214                  MOV     R0,MEMMAX       ;SAVE HIGHEST NON-KT MEM ADRS
002706  020027  020000                  CMP     R0,#20000       ;IS THERE MORE THAN 8K?
002712  101002                          BHI     4$              ;BR IF SO
002714  000137  003202                  JMP     TYPCOR          ;GO TYPE CORE AMT - NOTHING OVER 8K
002720  005237  002216          4$:     INC     CENAB           ;NON-ZERO REQ DPC MEM TEST
002724  005737  002220                  TST     KTENAB          ;DO WE NEED TO DO KT MEM SIZER?
002730  001002                          BNE     KTSIZR          ;BR IF SO
002732  000137  003202                  JMP     TYPCOR          ;GO TYPE CORE AMT - LESS THAN 28K
002736  012737  024610  000250  KTSIZR: MOV     #KTSER,@#KTERRV ;GET UP KT11 TRAP SERVICE ADRS
002744  012737  003022  000004          MOV     #2$,@#ERRVEC            ;RETURN TO 2$ ON KT TIMOUT
002752  005737  177572                  TST     @#KTSR0         ;LOOK FOR STATUS REG
002756  005077  177246          1$:     CLR     @KIPAR0         ;SET UP FOR TRAPS TO PAGE 0
002762  012777  077406  177242          MOV     #77406,@KIPDR0  ;4K PAGE LENGTH, EXPAND UP, R/W ACCESS
002770  012777  001600  177236          MOV     #1600,@KIPAR2   ;START AT 28K
002776  012777  077406  177232          MOV     #77406,@KIPDR2  ;4K PAGE LENGTH, EXPAND UP, R/W ACCESS
003004  012777  007600  177226          MOV     #7600,@KIPAR7   ;SET UP I/O 4K PAGE ADRS
003012  012777  077406  177222          MOV     #77406,@KIPDR7  ;4K PAGE LENGTH, EXPAND UP, R/W ACCESS
003020  000410                          BR      3$              ;SKIP OVER TIMEOUT STUFF
003022  022626                  2$:     CMP     (R6)+,(R6)+     ;FIX STACK SINCE NO RTI
003024  005037  002220                  CLR     KTENAB          ;THERE IS NO KT11
003030  012737  000006  000004          MOV     #ERRVEC+2,@#ERRVEC      ;RESTORE 4 & 6
003036  000137  003202                  JMP     TYPCOR          ;GO TYPE CORE AMT - KT NOT THERE
003042  012737  003114  000004  3$:     MOV     #5$,@#ERRVEC    ;SET UP NON-EXISTENT MEM TIMEOUT ADRS
003050  012700  040000          4$:     MOV     #40000,R0       ;START WITH ADRS 0 AT PAGE ADRS IN KIPAR2
003054  052737  001400  177572          BIS     #1400,@#KTSR0   ;ENAB KT (MAINT)
003062  005720                          TST     (R0)+           ;IS THIS BLK THERE?
003064  005720                          TST     (R0)+           ;CK INTERLEAVING ALSO
003066  042737  001400  177572          BIC     #1400,@#KTSR0   ;DISABLE KT
003074  062777  000200  177132          ADD     #200,@KIPAR2    ;ADVANCE TO NEXT 4K BLK
003102  027777  177126  177130          CMP     @KIPAR2,@KIPAR7 ;ARE WE OUT TO 128K?
003110  001357                          BNE     4$              ;BR IF NOT
003112  000403                          BR      6$              ;GO SET UP KTMAX
003114  005037  177572          5$:     CLR     @#KTSR0         ;DISABLE KT
003120  022626                          CMP     (R6)+,(R6)+     ;FIX STACK SINCE NO RTI
003122  012737  000006  000004  6$:     MOV     #ERRVEC+2,@#ERRVEC      ;RESTORE ERRVEC
003130  162777  000200  177076          SUB     #200,@KIPAR2    ;BACK UP TO LAST AVAILABLE BLK
003136  017737  177072  002222          MOV     @KIPAR2,KTMAX   ;SAVE IT
003144  023727  002222  001600          CMP     KTMAX,#1600     ;ARE WE OVER 28K?
003152  103004                          BHIS    7$              ;BR IF SO
003154  005037  002220                  CLR     KTENAB          ;DON'T USE KT
003160  000137  003202                  JMP     TYPCOR          ;KT THERE BUT NOTHING OVER 28K?
```

```
003164  013700  002222            7$:      MOV     KTMAX,R0        ;GET MAX PAGE ADRS VALUE
003170  006300                             ASL     R0              ;JUSTIFY RIGHT FOR TYPE
003172  000300                             SWAB    R0              ;
003174  006300                             ASL     R0              ;
003176  006300                             ASL     R0              ;
003200  000406                             BR      TYPMEM          ;GO TYPE IT
003202  013700  002214            TYPCOR:  MOV     MEMMAX,R0       ;GET TOP 4K BLK NO BELOW 28K
003206  000300                             SWAB    R0              ;SET IT UP FOR TYPE
003210  006200                             ASR     R0              ;ADJUST FOR TYPE
003212  006200                             ASR     R0              ;
003214  006200                             ASR     R0              ;
003216  062700  000004            TYPMEM:  ADD     #4,R0           ;POINT TO LAST ADRS
003222  005737  000042                     TST     @#42            ;TEST IF RUNNING CHAIN MODE UNDER XXDP
003226  001006                             BNE     RSTART          ;BR IF SO
003230  104400  027476                     TYPE    ,MSG3           ;GO TYPE 'MEMORY SIZE='
003234  010046                             MOV     R0,-(SP)        ;SAVE IT ON STK FOR DECIMAL TYPE
003236  104404                             TYPDS                   ;GO TYPE IT
003240  104400  027516                     TYPE    ,MSG4           ;TYPE 'K'
003244  000005                    RSTART:  RESET                   ;INITIALIZE THE VS60
003246  012706  001100                     MOV     #STACK,SP       ;SET UP STACK POINTER
003252  012737  000340  177776             MOV     #340,PSW        ;SET UP PSW TO HIGHEST LEVEL
003260  105037  001102                     CLRB    $TSTNM          ;CLR TEST NO. ON RESTARTS
```

```
                                                .SBTTL
                                                .SBTTL  THESE TESTS USE MAINT. SW 2
                                                .SBTTL

                                        ;;*************************************************************
                                        ;*TEST 1         TEST THAT THE JSR REL INSTR WILL DECREMENT THE STACK POINTER
                                        ;;*************************************************************
003264  000004                          TST1:   SCOPE
003266  012737  000001  001206                  MOV     #1,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
003274  012737  003330  001110                  MOV     #1$,$LPERR              ;SET UP SCOPE LOOP ADRS
003302  013737  002276  001122                  MOV     STKPT,$BDADR            ;SET UP REG 32 ADRS
003310  012700  020040                          MOV     #20040,R0               ;SET TOP OF STACK & MAINT SW2 IN R0
003314  012737  000034  001124                  MOV     #34,$GDDAT              ;WILL EXPECT PTR TO BE 34 INITIALLY
003322  012737  163000  032342                  MOV     #163000,BUFFER          ;SET UP JSR INSTR
003330  010077  176742                  1$:     MOV     R0,@STKPT               ;SET UP STACK PTR
003334  012777  032342  176702                  MOV     #BUFFER,@DPC            ;START
003342  004537  024426                          JSR     R5,DELAY                ;STALL FOR STACKING
003346  000001                                  BIT0                            ;COUNT TO 1
003350  017737  176722  001126                  MOV     @STKPT,$BDDAT           ;READ STACK POINTER
003356  042737  177703  001126                  BIC     #177703,$BDDAT          ;ONLY WANT STACK POINTER - TOP OF STK SB 0
003364  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;IS IT CORRECT?
003372  001401                                  BEQ     2$                      ;BR IF OK
003374  104003                                  ERROR   3                       ;JSR REL INSTR FAILED TO DECREMENT STACK PTR
003376  162700  000004                  2$:     SUB     #4,R0                   ;SET UP NEXT STK PTR VALUE IN R0
003402  162737  000004  001124                  SUB     #4,$GDDAT               ;ADVANCE TO NEXT EXPECTED PTR VALUE
003410  001347                                  BNE     1$                      ;BR IF THIS VALUE NOT TESTED YET

                                        ;;*************************************************************
                                        ;*TEST 2         TEST THAT THE STACK OVERFLOW BIT CAN SET
                                        ;;*************************************************************
003412  000004                          TST2:   SCOPE
003414  012737  000002  001206                  MOV     #2,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
003422  012737  000340  177776                  MOV     #340,PSW                ;SET PRIORITY TO HIGHEST LEVEL
003430  013737  002256  001122                  MOV     SREG1,$BDADR            ;SET UP REG 12 ADRS
003436  012777  020040  176632                  MOV     #20040,@STKPT           ;RESET THE STACK PTR
003444  005037  001124                          CLR     $GDDAT                  ;EXPECT NO OVERFLOW FOR 8 JSR'S
003450  012700  000011                          MOV     #11,R0                  ;SET UP COUNT FOR 9 JSR'S
003454  012737  163000  032342                  MOV     #163000,BUFFER          ;SET UP JSR INSTR
003462  012777  032342  176554          1$:     MOV     #BUFFER,@DPC            ;START
003470  004537  024426                          JSR     R5,DELAY                ;STALL FOR STACKING
003474  000001                                  BIT0                            ;COUNT TO 1
003476  005300                                  DEC     R0                      ;COUNT STACKING OPERATION
003500  001410                                  BEQ     2$                      ;BR IF THIS IS OVERFLOW
003502  032777  020000  176546                  BIT     #20000,@SREG1           ;SEE THAT OVERFLOW IS NOT SET
003510  001764                                  BEQ     1$                      ;BR IF OVERFLOW NOT SET
003512  012737  020000  001126                  MOV     #20000,$BDDAT           ;INDICATE STACK OVERFLOW
003520  104004                                  ERROR   4                       ;STACK OVERFLOW SET PREMATURELY
003522  032777  020000  176526          2$:     BIT     #20000,@SREG1           ;IS STACK OVERFLOW SET?
003530  001006                                  BNE     TST3                    ;;ADVANCE TO NEXT TEST IF OK
003532  012737  020000  001124                  MOV     #20000,$GDDAT           ;EXPECTED STACK OVERFLOW
003540  005037  001126                          CLR     $BDDAT                  ;INDICATE IT WAS NOT SET
003544  104004                                  ERROR   4                       ;STACK OVERFLOW FAILED TO SET

                                        ;;*************************************************************
                                        ;*TEST 3         TEST THAT START WILL SET 'TOP OF STACK'
                                        ;;*************************************************************
```

```
003546  000004                    TST3:   SCOPE
003550  012737  000003  001206            MOV     #3,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
003556  013737  002276  001122            MOV     STKPT,$BDADR      ;SET UP REG 32 ADRS
003564  012737  000040  001124            MOV     #40,$GDDAT        ;EXPECT TOP OF STACK
003572  012777  010037  176476            MOV     #10037,@STKPT     ;SET ALL STACK BITS & MAINT 1
003600  005077  176440                    CLR     @DPC              ;START - NO NPR
003604  017737  176466  001126            MOV     @STKPT,$BDDAT     ;READ STK PTR REG
003612  042737  177701  001126            BIC     #177701,$BDDAT    ;SAVE ONLY STK PTR BITS
003620  023737  001124  001126            CMP     $GDDAT,$BDDAT     ;DID START SET 'TOP OF STACK'?
003626  001401                            BEQ     TST4              ;;NEXT TEST IF SO
003630  104072                            ERROR   72                ;START FAILED TO SET 'TOP OF STACK'

                                  ;;*****************************************************************
                                  ;*TEST 4         TEST THAT THE JSR RELATIVE INSTR WILL PUSH DPC ONTO STACK
                                  ;;*****************************************************************
003632  000004                    TST4:   SCOPE
003634  012737  000004  001206            MOV     #4,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
003642  013737  002272  001122            MOV     STKVAL,$BDADR     ;SET UP REG 26 ADRS
003650  012737  163000  032342            MOV     #163000,BUFFER    ;SET UP JSR RELATIVE INSTR
003656  012700  000034                    MOV     #34,R0            ;SET UP STK SEL AFTER JSR IN R0
003662  012701  020040                    MOV     #20040,R1         ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
003666  012737  032344  001124            MOV     #BUFFER+2,$GDDAT       ;EXPECT ADRS BUFFER+2 ON STACK
003674  012737  003702  001110            MOV     #1$,$LPERR        ;SET UP SCOPE LOOP ADRS
003702  010177  176370            1$:     MOV     R1,@STKPT         ;SET STK PTR BEFORE JSR
003706  012777  032342  176330            MOV     #BUFFER,@DPC      ;START
003714  004537  024426                    JSR     R5,DELAY          ;STALL FOR STACKING
003720  000001                            BIT0                      ;COUNT TO 1
003722  010077  176350                    MOV     R0,@STKPT         ;SELECT STK WORD & BYTE
003726  017737  176340  001126            MOV     @STKVAL,$BDDAT    ;READ STACK BYTE
003734  042737  000001  001126            BIC     #BIT0,$BDDAT      ;DON'T WANT LSB
003742  023737  001124  001126            CMP     $GDDAT,$BDDAT     ;DID THE DPC GET STORED OK?
003750  001403                            BEQ     2$                ;BR IF OK
003752  010037  001164                    MOV     R0,$TMP0          ;SET UP STK LEVEL FOR ER TYPE
003756  104005                            ERROR   5                 ;DPC FAILED TO STACK AT LEVEL INDICATED
003760  162701  000004            2$:     SUB     #4,R1             ;ADVANCE TO NEXT STK LEVEL
003764  162700  000004                    SUB     #4,R0             ;ALL STACK LOCATIONS TESTED?
003770  100344                            BPL     1$                ;BR IF NOT

                                  ;;*****************************************************************
                                  ;*TEST 5         TEST THAT THE JSR ABS INSTR WILL PUSH DPC ONTO STACK
                                  ;;*****************************************************************
003772  000004                    TST5:   SCOPE
003774  012737  000005  001206            MOV     #5,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
004002  013737  002272  001122            MOV     STKVAL,$BDADR     ;SET UP REG 26 ADRS
004010  012737  162000  032342            MOV     #162000,BUFFER    ;SET UP JSR ABS INSTR
004016  012737  032346  001124            MOV     #BUFFER+4,$GDDAT       ;EXPCT ADRS BUFFER+4 ON STACK
004024  012701  020040                    MOV     #20040,R1         ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
004030  012700  000034                    MOV     #34,R0            ;SET UP STACK SEL AFTER JSR IN R0
004034  012737  004042  001110            MOV     #1$,$LPERR        ;SET UP SCOPE LOOP ADRS
004042  010177  176230            1$:     MOV     R1,@STKPT         ;SET UP STACK PTR BEFORE JSR
004046  012777  032342  176170            MOV     #BUFFER,@DPC      ;START
004054  004537  024426                    JSR     R5,DELAY          ;STALL FOR STACKING
004060  000001                            BIT0                      ;COUNT TO 1
004062  005277  176156                    INC     @DPC              ;ADVANCE TO ABS ADRS
004066  004537  024426                    JSR     R5,DELAY          ;STALL FOR STACKING
004072  000001                            BIT0                      ;COUNT TO 1
```

```
004074  010077  176176                          MOV     R0,@STKPT               ;SELECT STK WORD & BYTE
004100  017737  176166  001126                  MOV     @STKVAL,$BDDAT          ;READ STACK BYTE
004106  042737  000001  001126                  BIC     #BIT0,$BDDAT            ;DON'T WANT LSB
004114  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;DID DPC GET STORED OK?
004122  001403                                  BEQ     2$                      ;BR IF OK
004124  010037  001164                          MOV     R0,$TMP0                ;SET UP STK LEVEL FOR ER TYPE
004130  104005                                  ERROR   5                       ;DPC FAILED TO STACK AT LEVEL INDICATED
004132  162701  000004                  2$:     SUB     #4,R1                   ;ADVANCE TO NEXT STK LEVEL
004136  162700  000004                          SUB     #4,R0                   ;ALL STACK LOCATIONS TESTED?
004142  100337                                  BPL     1$                      ;BR IF NOT


                                        ;;*****************************************************************
                                        ;*TEST 6         TEST THAT THE JSR RELATIVE INSTR WILL PUSH DPC NAME REG ONTO STACK
                                        ;;*****************************************************************
004144  000004                          TST6:   SCOPE
004146  012737  000004  001166                  MOV     #4,$TIMES               ;;DO 4 ITERATIONS
004154  012737  000006  001206                  MOV     #6,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
004162  012737  153777  032342                  MOV     #153777,BUFFER          ;SET UP NAME INSTR WITH 3777
004170  012737  163000  032344                  MOV     #163000,BUFFER+2                ;SET UP JSR INSTR
004176  012737  077760  001124                  MOV     #77760,$GDDAT           ;EXPECT 7776 INITIALLY
004204  012737  004222  001110                  MOV     #2$,$LPERR              ;SET SCOPE LOOP ADRS
004212  012701  020040                  1$:     MOV     #20040,R1               ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
004216  012700  000035                          MOV     #35,R0                  ;SET STACK SELECTION AFTER JSR IN R0
004222  010177  176050                  2$:     MOV     R1,@STKPT               ;SET STK PTR BEFORE JSR
004226  012777  032342  176010                  MOV     #BUFFER,@DPC            ;START
004234  013737  002272  001122                  MOV     STKVAL,$BDADR           ;SET UP REG ADRS 26
004242  005277  175776                          INC     @DPC                    ;DO NEXT INSTR
004246  004537  024426                          JSR     R5,DELAY                ;ALLOW TIME FOR STACKING
004252  000001                                  BIT0                            ;COUNT TO 1
004254  010077  176016                          MOV     R0,@STKPT               ;SELECT STK WORD & BYTE
004260  017737  176006  001126                  MOV     @STKVAL,$BDDAT          ;READ STACK BYTE
004266  042737  100017  001126                  BIC     #100017,$BDDAT          ;ONLY WANT NAME BITS
004274  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;DID THE NAME BITS GET STORED OK?
004302  001403                                  BEQ     3$                      ;BR IF OK
004304  010037  001164                          MOV     R0,$TMP0                ;SET UP STK LEVEL FOR ER TYPE
004310  104006                                  ERROR   6                       ;DPC NAME FAILED TO STACK AT LEVEL INDICATED
004312  162701  000004                  3$:     SUB     #4,R1                   ;ADVANCE TO NEXT STK LEVEL
004316  162700  000004                          SUB     #4,R0                   ;ALL STACK LOCATIONS TESTED?
004322  100337                                  BPL     2$                      ;BR IF NOT
004324  162737  000020  001124                  SUB     #20,$GDDAT              ;ADVANCE TO NEXT PATTERN
004332  100403                                  BMI     TST7                    ;ADVANCE TO NEXT TEST IF ALL PATTERNS TESTED
004334  005337  032342                          DEC     BUFFER                  ;SET UP NAME INSTR WITH NEXT PATTERN
004340  000724                                  BR      1$                      ;GO TRY NEXT PATTERN


                                        ;;*****************************************************************
                                        ;*TEST 7         TEST THAT THE MODE BITS GET PUSHED ON THE STACK
                                        ;;*****************************************************************
004342  000004                          TST7:   SCOPE
004344  012737  000010  001166                  MOV     #10,$TIMES              ;;DO 10 ITERATIONS
004352  012737  000007  001206                  MOV     #7,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
004360  005037  001124                          CLR     $GDDAT                  ;INITIALLY EXPECT ZERO FOR MODE
004364  005002                                  CLR     R2                      ;SET INSTR TO CHAR MODE INITIALLY
004366  012737  163000  032344                  MOV     #163000,BUFFER+2                ;SET UP JSR INSTR
004374  012737  004412  001110                  MOV     #2$,$LPERR              ;SET UP LOOP ADRS
004402  012701  020040                  1$:     MOV     #20040,R1               ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
```

```
004406  012700  000035                    MOV     #35,R0            ;SET UP STK LEVEL AFTER JSR IN R0
004412  010177  175660          2$:       MOV     R1,@STKPT         ;SET STK PTR BEFORE JSR
004416  012737  100000  032342            MOV     #100000,BUFFER    ;SET UP INSTR
004424  050237  032342                    BIS     R2,BUFFER         ;ADD IN CURRENT INSTR MODE
004430  012777  032342  175606            MOV     #BUFFER,@DPC      ;START
004436  013737  002272  001122            MOV     STKVAL,$BDADR     ;SET UP REG ADRS 26
004444  005277  175574                    INC     @DPC              ;RESUME
004450  004537  024426                    JSR     R5,DELAY          ;STALL FOR STACKING
004454  000001                            BIT0                      ;COUNT TO 1
004456  010077  175614                    MOV     R0,@STKPT         ;SELECT STACK WORD & BYTE
004462  017737  175604  001126            MOV     @STKVAL,$BDDAT    ;READ MODE BITS
004470  042737  177760  001126            BIC     #177760,$BDDAT    ;DON'T WANT NAME BITS
004476  023737  001124  001126            CMP     $GDDAT,$BDDAT     ;ARE THE MODE BITS CORRECT?
004504  001403                            BEQ     3$                ;BR IF OK
004506  010037  001164                    MOV     R0,$TMP0          ;SET UP STK LEVEL BEFORE ER TYPE
004512  104007                            ERROR   7                 ;MODE FAILED TM STACK AT LEVEL INDICATED
004514  162701  000004          3$:       SUB     #4,R1             ;ADVANCE TO NEXT STK LEVEL
004520  162700  000004                    SUB     #4,R0             ;HAS THIS INSTR MODE BEEN WRITTEN THRU STACK?
004524  100332                            BPL     2$                ;BR IF NOT
004526  022702  044000                    CMP     #44000,R2         ;IS THIS THE ABS VECTOR INSTR?
004532  001405                            BEQ     TST10             ;;ADVANCE TO NEXT TEST IF SPARE HAS BEEN TESTED
004534  062702  004000                    ADD     #4000,R2          ;ADVANCE TO NEXT INSTR MODE
004540  005237  001124                    INC     $GDDAT            ;ADVANCE TO NEXT MODE EXPECTED
004544  000716                            BR      1$                ;TRY NEXT INSTR

                                ;;*****************************************************************
                                ;*TEST 10        TEST THAT THE VECTOR SCALE BITS GET PUSHED ON THE STACK
                                ;;*****************************************************************
004546  000004                  TST10:    SCOPE
004550  012737  000010  001166            MOV     #10,$TIMES        ;;DO 10 ITERATIONS
004556  012737  000010  001206            MOV     #10,$TESTN        ;;SET TEST NUMBER IN APT MAIL BOX
004564  012737  074000  001124            MOV     #74000,$GDDAT     ;EXPECT ALL ONES INITIALLY
004572  012737  154037  032342            MOV     #154037,BUFFER    ;SET UP LOAD STATUS C INSTR
004600  012737  163000  032344            MOV     #163000,BUFFER+2          ;SET UP JSR INSTR
004606  012737  004624  001110            MOV     #2$,$LPERR        ;SET UP SCOPE LOOP ADRS
004614  012701  020040          1$:       MOV     #20040,R1         ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
004620  012700  000036                    MOV     #36,R0            ;SET UP STK LEVEL AFTER JSR IN R0
004624  010177  175446          2$:       MOV     R1,@STKPT         ;SET STK PTR BEFORE JSR
004630  012777  032342  175406            MOV     #BUFFER,@DPC      ;START
004636  013737  002272  001122            MOV     STKVAL,$BDADR     ;SET UP REG ADRS 26
004644  005277  175374                    INC     @DPC              ;ADVANCE TO JSR INSTR
004650  004537  024426                    JSR     R5,DELAY          ;STALL FOR STACKING
004654  000001                            BIT0                      ;COUNT TO 1
004656  010077  175414                    MOV     R0,@STKPT         ;SELECT STACK WORD & BYTE
004662  017737  175404  001126            MOV     @STKVAL,$BDDAT    ;READ VECTOR SCALE FROM STACK
004670  042737  103777  001126            BIC     #103777,$BDDAT    ;ONLY WANT VECTOR SCALE BITS
004676  023737  001124  001126            CMP     $GDDAT,$BDDAT     ;ARE THEY CORRECT?
004704  001403                            BEQ     3$                ;BR IF OK
004706  010037  001164                    MOV     R0,$TMP0          ;SET UP STK LEVEL FOR ER TYPE
004712  104010                            ERROR   10                ;VECTOR SCALE FAILED TO STACK AT LEVEL INDICATED
004714  162701  000004          3$:       SUB     #4,R1             ;ADVANCE TO NEXT STK LEVEL
004720  162700  000004                    SUB     #4,R0             ;ALL STACK LOCATIONS TESTED?
004724  100337                            BPL     2$                ;BR IF NOT
004726  162737  004000  001124            SUB     #4000,$GDDAT      ;ADVANCE TO NEXT VECTOR SCALE PATTERN
004734  100403                            BMI     TST11             ;;ADVANCE TO NEXT TEST IF ALL PATTERNS TESTED
004736  005337  032342                    DEC     BUFFER            ;UPDATE INSTRUCTION
```

```
004742  000724                           BR      1$                  ;TRY NEXT VECTOR SCALE VALUE

                                  ;;*********************************************************************
                                  ;*TEST 11       TEST THAT THE CHARACTER SCALE BITS GET PUSHED ON THE STACK
                                  ;;*********************************************************************
004744  000004                    TST11: SCOPE
004746  012737  000010  001166           MOV     #10,$TIMES          ;;DO 10 ITERATIONS
004754  012737  000011  001206           MOV     #11,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
004762  012737  003000  001124           MOV     #3000,$GDDAT        ;EXPECT ALL ONES INITIALLY
004770  012737  154340  032342           MOV     #154340,BUFFER      ;SET UP LOAD STATUS C INSTR
004776  012737  163000  032344           MOV     #163000,BUFFER+2            ;SET UP JSR INSTR
005004  012737  005022  001110           MOV     #2$,$LPERR          ;SET UP LOOP ADRS
005012  012701  020040            1$:    MOV     #20040,R1           ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
005016  012700  000036                   MOV     #36,R0              ;SET UP STK LEVEL AFTER JSR IN R0
005022  010177  175250            2$:    MOV     R1,@STKPT           ;SET STK PTR BEFORE JSR
005026  012777  032342  175210           MOV     #BUFFER,@DPC        ;START
005034  013737  002272  001122           MOV     STKVAL,$BDADR       ;SET UP REG ADRS 26
005042  005277  175176                   INC     @DPC                ;ADVANCE TO JSR INSTR
005046  004537  024426                   JSR     R5,DELAY            ;STALL FOR STACKING
005052  000001                           BIT0                        ;COUNT TO 1
005054  010077  175216                   MOV     R0,@STKPT           ;SELECT STACK WORD & BYTE
005060  017737  175206  001126           MOV     @STKVAL,$BDDAT      ;READ CHARACTER SCALE
005066  042737  174777  001126           BIC     #174777,$BDDAT      ;ONLY WANT CHAR SCALE
005074  023737  001124  001126           CMP     $GDDAT,$BDDAT       ;ARE THEY CORRECT?
005102  001403                           BEQ     3$                  ;BR IF OK
005104  010037  001164                   MOV     R0,$TMP0            ;SET UP STK LEVEL FOR ER TYPE
005110  104011                           ERROR   11                  ;CHARACTER SCALE FAILED TO STACK AT LEVEL INDICATED
005112  162701  000004            3$:    SUB     #4,R1               ;ADVANCE TO NEXT STK LEVEL
005116  162700  000004                   SUB     #4,R0               ;ALL STACK LOCATIONS TESTED?
005122  100337                           BPL     2$                  ;BR IF NOT
005124  162737  001000  001124           SUB     #1000,$GDDAT        ;ADVANCE CHARACTER SCALE PATTERN
005132  100404                           BMI     TST12               ;;ADVANCE TO NEXT TEST IF ALL PATTERNS TESTED
005134  162737  000040  032342           SUB     #40,BUFFER          ;UPDATE INSTRUCTION
005142  000723                           BR      1$                  ;TRY NEXT CHARACTER SCALE VALUE

                                  ;;*********************************************************************
                                  ;*TEST 12       TEST THAT THE CHARACTER ROTATE BIT GETS PUSHED ON THE STACK
                                  ;;*********************************************************************
005144  000004                    TST12: SCOPE
005146  012737  000012  001206           MOV     #12,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
005154  012737  000200  001124           MOV     #200,$GDDAT         ;EXPECT IT TO BE SET INITIALLY
005162  012737  155400  032342           MOV     #155400,BUFFER      ;SET UP LOAD STATUS C INSTR
005170  012737  163000  032344           MOV     #163000,BUFFER+2            ;SET UP JSR INSTR
005176  012737  005214  001110           MOV     #2$,$LPERR          ;SET UP LOOP ADRS
005204  012701  020040            1$:    MOV     #20040,R1           ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
005210  012700  000036                   MOV     #36,R0              ;SET UP STK LEVEL AFTER JSR IN R0
005214  010177  175056            2$:    MOV     R1,@STKPT           ;SET STK PTR BEFORE JSR
005220  012777  032342  175016           MOV     #BUFFER,@DPC        ;START
005226  013737  002272  001122           MOV     STKVAL,$BDADR       ;SET UP REG ADRS 26
005234  005277  175004                   INC     @DPC                ;ADVANCE TO JSR INSTR
005240  004537  024426                   JSR     R5,DELAY            ;STALL FOR STACKING
005244  000001                           BIT0                        ;COUNT TO 1
005246  010077  175024                   MOV     R0,@STKPT           ;SELECT STACK WORD & BYTE
005252  017737  175014  001126           MOV     @STKVAL,$BDDAT      ;READ CHARACTER ROTATE BIT
005260  042737  177577  001126           BIC     #177577,$BDDAT      ;ONLY WANT CHARACTER ROTATE BIT
005266  023737  001124  001126           CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
```

```
005274  001403                          BEQ     3$                  ;BR IF OK
005276  010037  001164                  MOV     R0,$TMPO            ;SET UP STK LEVEL FOR ER TYPE
005302  104012                          ERROR   12                  ;CHARACTER ROTATE FEILED TO STACK AT LEVEL INDICATED
005304  162701  000004          3$:     SUB     #4,R1               ;ADVANCE TO NEXT STK LEVEL
005310  162700  000004                  SUB     #4,R0               ;ALL STACK LOCATIONS TESTED?
005314  100337                          BPL     2$                  ;BR IF NOT
005316  162737  000400  032342          SUB     #400,BUFFER         ;UPDATE INSTR
005324  162737  000200  001124          SUB     #200,$GDDAT         ;EXPECT ZERO NEXT
005332  100324                          BPL     1$                  ;BR IF RESET STATE NOT TESTED

                                ;;*************************************************************
                                ;*TEST 13        TEST THAT THE INTENSITY LEVEL BITS GET PUSHED ON THE STACK
                                ;;*************************************************************
005334  000004                  TST13:  SCOPE
005336  012737  000010  001166          MOV     #10,$TIMES          ;;DO 10 ITERATIONS
005344  012737  000013  001206          MOV     #13,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
005352  012737  000160  001124          MOV     #160,$GDDAT         ;EXPECT ALL BITS INITIALLY
005360  012737  103600  032342          MOV     #103600,BUFFER      ;SET UP ALL INTENSITY LEVEL BITS INITIALLY
005366  012737  163000  032344          MOV     #163000,BUFFER+2        ;SET UP JSR INSTR
005374  012737  005412  001110          MOV     #2$,$LPERR          ;SET UP LOOP ADRS
005402  012701  020040          1$:     MOV     #20040,R1           ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
005406  012700  000036                  MOV     #36,R0              ;SET UP STK LEVEL AFTER JSR IN R0
005412  010177  174660          2$:     MOV     R1,@STKPT           ;SET STK PTR BEFORE JSR
005416  012777  032342  174620          MOV     #BUFFER,@DPC        ;START
005424  013737  002272  001122          MOV     STKVAL,$BDADR       ;SET UP REG ADRS26
005432  012777  032344  174604          MOV     #BUFFER+2,@DPC      ;START
005440  004537  024426                  JSR     R5,DELAY            ;STALL FOR STACKING
005444  000001                          BIT0                        ;COUNT TO 1
005446  010077  174624                  MOV     R0,@STKPT           ;SELECT STACK WORD & BYTE
005452  017737  174614  001126          MOV     @STKVAL,$BDDAT      ;READ INTENSITY LEVEL
005460  042737  177617  001126          BIC     #177617,$BDDAT      ;ONLY WANT INTENISTY LEVEL BITS
005466  023737  001124  001126          CMP     $GDDAT,$BDDAT       ;ARE THEY CORRECT?
005474  001403                          BEQ     3$                  ;BR IF OK
005476  010037  001164                  MOV     R0,$TMPO            ;SET UP STK LEVEL FOR ER TYPE
005502  104013                          ERROR   13                  ;INTENSITY LEVEL FAILED TO STACK AT LEVEL INDICATED
005504  162701  000004          3$:     SUB     #4,R1               ;ADVANCE TO NEXT STK LEVEL
005510  162700  000004                  SUB     #4,R0               ;ALL STACK LOCATIONS TESTED?
005514  100336                          BPL     2$                  ;BR IF NOT
005516  162737  000200  032342          SUB     #200,BUFFER         ;SET UP NEXT INTENSITY LEVEL
005524  162737  000020  001124          SUB     #20,$GDDAT          ;ADVANCE TO NEXT INTENSITY LEVEL
005532  100323                          BPL     1$                  ;BR IF RESET STATE NOT TESTED

                                ;;*************************************************************
                                ;*TEST 14        TEST THAT THE COLOR LEVEL BITS GET PUSHED ON THE STACK
                                ;;*************************************************************
005534  000004                  TST14:  SCOPE
005536  012737  000010  001166          MOV     #10,$TIMES          ;;DO 10 ITERATIONS
005544  012737  000014  001206          MOV     #14,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
005552  012737  000014  001124          MOV     #14,$GDDAT          ;EXPECT ALL BIT INITIALLY
005560  012737  175600  032342          MOV     #175600,BUFFER      ;SET UP ALL COLOR LEVEL BITS INITIALLY
005566  012737  163000  032344          MOV     #163000,BUFFER+2        ;SET UP JSR INSTR
005574  012737  005602  001110          MOV     #1$,$LPERR          ;SET UP LOOP ADRS
005602  012701  020040          1$:     MOV     #20040,R1           ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
005606  012700  000036                  MOV     #36,R0              ;SET UP STK LEVEL AFTER JSR IN R0
005612  010177  174460          2$:     MOV     R1,@STKPT           ;SET STK PTR BEFORE JSR
005616  012777  032342  174420          MOV     #BUFFER,@DPC        ;START
```

```
005624  013737  002272  001122          MOV     STKVAL,$BDADR   ;SET UP REG ADRS 26
005632  005277  174406                  INC     @DPC            ;ADVANCE TO JSR INSTR
005636  004537  024426                  JSR     R5,DELAY        ;STALL FOR STACKING
005642  000001                          BIT0                    ;COUNT TO 1
005644  010077  174426                  MOV     R0,@STKPT       ;SELECT STACK WORD & BYTE
005650  017737  174416  001126          MOV     @STKVAL,$BDDAT  ;READ COLOR LEVEL FROM STACK
005656  042737  177763  001126          BIC     #177763,$BDDAT  ;SAVE ONLY COLOR LEVEL
005664  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY CORRECT?
005672  001403                          BEQ     3$              ;BR IF SO
005674  010037  001164                  MOV     R0,$TMP0        ;SET UP STK LEVEL FOR ER TYPE
005700  104014                          ERROR   14              ;COLOR LEVEL FAILED TO STACK AT LEVEL INDICATED
005702  162701  000004          3$:     SUB     #4,R1           ;ADVANCE TO NEXT STK LEVEL
005706  162700  000004                  SUB     #4,R0           ;ALL STACK LOCATIONS TESTED?
005712  100337                          BPL     2$              ;BR IF NOT
005714  162737  000200  032342          SUB     #200,BUFFER     ;ADVANCE COLOR LEVEL AT INSTR LOC
005722  162737  000004  001124          SUB     #4,$GDDAT       ;ADVANCE COLOR LEVEL AT EXPECTED LOC
005730  100324                          BPL     1$              ;BR IF RESET STATE NOT TESTED

                                ;;*****************************************************************
                                ;*TEST 15       TEST THAT ITALICS GETS PUSHED ON THE STACK
                                ;;*****************************************************************
005732  000004                  TST15:  SCOPE
005734  012737  000015  001206          MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
005742  012737  100000  001124          MOV     #100000,$GDDAT  ;EXPECT IT TO BE SET INITIALLY
005750  012737  170060  032342          MOV     #170060,BUFFER  ;SET UP ITALICS AT INSTR LOC
005756  012737  163000  032344          MOV     #163000,BUFFER+2        ;SET UP JSR INSTR
005764  012737  006002  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
005772  012701  020040          1$:     MOV     #20040,R1       ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
005776  012700  000036                  MOV     #36,R0          ;SET UP STK LEVEL AFTER JSR IN R0
006002  010177  174270          2$:     MOV     R1,@STKPT       ;SET STK PTR BEFORE JSR
006006  012777  032342  174230          MOV     #BUFFER,@DPC    ;START
006014  013737  002272  001122          MOV     STKVAL,$BDADR   ;SET UP REG ADRS 26
006022  005277  174216                  INC     @DPC            ;ADVANCE TO JSR INSTR
006026  004537  024426                  JSR     R5,DELAY        ;STALL FOR STACKING
006032  000001                          BIT0                    ;COUNT TO 1
006034  010077  174236                  MOV     R0,@STKPT       ;SELECT STACK WORD & BYTE
006040  017737  174226  001126          MOV     @STKVAL,$BDDAT  ;READ ITALICS FROM STACK
006046  042737  077777  001126          BIC     #77777,$BDDAT   ;SAVE AONLY ITALICS BIT
006054  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
006062  001403                          BEQ     3$              ;BR IF SO
006064  010037  001164                  MOV     R0,$TMP0        ;SET UP STK LEVEL FOR ER TYPE
006070  104015                          ERROR   15              ;ITALICS FAILED TO STACK AT LEVEL INDICATED
006072  162701  000004          3$:     SUB     #4,R1           ;ADVANCE TO NEXT STK LEVEL
006076  162700  000004                  SUB     #4,R0           ;ALL STACK LOCATIONS TESTED?
006102  100337                          BPL     2$              ;BR IF NOT
006104  162737  000020  032342          SUB     #20,BUFFER      ;RESET ITALICS BIT AT INSTR LOC
006112  162737  100000  001124          SUB     #100000,$GDDAT  ;RESET ITALICS AT EXPECTED LOCATION
006120  100324                          BPL     1$              ;BR IF RESET STATE NOT TESTED

                                ;;*****************************************************************
                                ;*TEST 16       TEST THAT MENU GETS PUSHED ON THE STACK
                                ;;*****************************************************************
006122  000004                  TST16:  SCOPE
006124  012737  000016  001206          MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
006132  012737  020000  001124          MOV     #20000,$GDDAT   ;EXPECT IT TO BE SET INITIALLY
006140  012737  170003  032342          MOV     #170003,BUFFER  ;SET UP MENU AT INSTR LOC
```

```
006146  012737  163000  032344          MOV     #163000,BUFFER+2        ;SET UP JSR INSTR
006154  012737  006172  001110          MOV     #2$,$LPERR              ;SET UP SCOPE LOOP ADRS
006162  012701  020040          1$:     MOV     #20040,R1               ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
006166  012700  000037                  MOV     #37,R0                  ;SET UP STK LEVEL AFTER JSR IN R0
006172  010177  174100          2$:     MOV     R1,@STKPT               ;SET STK PTR BEFORE JSR
006176  012777  032342  174040          MOV     #BUFFER,@DPC            ;START
006204  013737  002272  001122          MOV     STKVAL,$BDADR           ;SET UP REG ADRS 26
006212  005277  174026                  INC     @DPC                    ;ADVANCE TO JSR INSTR
006216  004537  024426                  JSR     R5,DELAY                ;STALL FOR STACKING
006222  000001                          BIT0                            ;COUNT TO 1
006224  010077  174046                  MOV     R0,@STKPT               ;SELECT STACK WORD & BYTE
006230  017737  174036  001126          MOV     @STKVAL,$BDDAT          ;READ MENU BIT
006236  042737  157777  001126          BIC     #157777,$BDDAT          ;SAVE ONLY MENU BIT
006244  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;IS IT CORRECT?
006252  001403                          BEQ     3$                      ;BR IF SO
006254  010037  001164                  MOV     R0,$TMP0                ;SET UP STK LEVEL FOR ER TYPE
006260  104016                          ERROR   16                      ;MENU FAILED TO STACK AT LEVEL INDICATED
006262  162701  000004          3$:     SUB     #4,R1                   ;ADVANCE TO NEXT STK LEVEL
006266  162700  000004                  SUB     #4,R0                   ;ALL STACK LOCATIONS TESTED?
006272  100337                          BPL     2$                      ;BR IF NOT
006274  005337  032342                  DEC     BUFFER                  ;RESET MENU BIT AT INSTR LOC
006300  162737  020000  001124          SUB     #20000,$GDDAT           ;RESET MENU BIT AT EXPECTED LOC
006306  100325                          BPL     1$                      ;BR IF RESET STATE NOT TESTED

                                ;;*************************************************************
                                ;*TEST 17       TEST THAT BLINK ENA GETS PUSHED ON THE STACK
                                ;;*************************************************************

006310  000004                  TST17:  SCOPE
006312  012737  000017  001206          MOV     #17,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
006320  012737  100000  001124          MOV     #100000,$GDDAT          ;EXPECT IT TO BE SET INITIALLY
006326  012737  100030  032342          MOV     #100030,BUFFER          ;SET UP BLINK ENA AT INSTR LOC
006334  012737  163000  032344          MOV     #163000,BUFFER+2        ;SET UP JSR INSTR
006342  012737  006360  001110          MOV     #2$,$LPERR              ;SET UP SCOPE LOOP ADRS
006350  012701  020040          1$:     MOV     #20040,R1               ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
006354  012700  000037                  MOV     #37,R0                  ;SET UP STK LEVEL AFTER JSR IN R0
006360  010177  173712          2$:     MOV     R1,@STKPT               ;SET STK PTR BEFORE JSR
006364  012777  032342  173652          MOV     #BUFFER,@DPC            ;START
006372  013737  002272  001122          MOV     STKVAL,$BDADR           ;SET UP REG ADRS 26
006400  012777  032344  173636          MOV     #BUFFER+2,@DPC          ;START
006406  004537  024426                  JSR     R5,DELAY                ;STALL FOR STACKING
006412  000001                          BIT0                            ;COUNT TO 1
006414  010077  173656                  MOV     R0,@STKPT               ;SELECT STACK WORD & BYTE
006420  017737  173646  001126          MOV     @STKVAL,$BDDAT          ;READ BLINK ENA BIT
006426  042737  077777  001126          BIC     #77777,$BDDAT           ;SAVE ONLY BLINK ENA
006434  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;IS IT CORRECT?
006442  001403                          BEQ     3$                      ;BR IF SO
006444  010037  001164                  MOV     R0,$TMP0                ;SET UP STK LEVEL FOR ER TYPE
006450  104017                          ERROR   17                      ;BLINK ENA FAILED TO STACK AT LEVEL INDICATED
006452  162701  000004          3$:     SUB     #4,R1                   ;ADVANCE TO NEXT STK LEVEL
006456  162700  000004                  SUB     #4,R0                   ;ALL STACK LOCATIONS TESTED?
006462  100336                          BPL     2$                      ;BR IF NOT
006464  162737  000010  032342          SUB     #10,BUFFER              ;RESET BLINK ENA BIT INSTR LOC
006472  162737  100000  001124          SUB     #100000,$GDDAT          ;RESET BLINK ENA AT EXPECTED LOC
006500  100323                          BPL     1$                      ;BR IF RESET STATE NOT TESTED

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

```
                                        ;*TEST 20       TEST THAT STOP INTR ENA GETS PUSHED ON THE STACK
                                        ;;*******************************************************************
006502  000004                          TST20:  SCOPE
006504  012737  000020  001206                  MOV     #20,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
006512  012737  000001  001124                  MOV     #1,$GDDAT           ;EXPECT IT TO BE SET INITIALLY
006520  012737  171400  032342                  MOV     #171400,BUFFER      ;SET UP STOP INTR ENA AT INSTR LOC
006526  012737  163000  032344                  MOV     #163000,BUFFER+2            ;SET UP JSR INSTR
006534  012737  006552  001110                  MOV     #2$,$LPERR          ;SET UP SCOPE LOOP ADRS
006542  012701  020040                  1$:     MOV     #20040,R1           ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
006546  012700  000037                          MOV     #37,R0              ;SET UP STK LEVEL AFTER JSR IN R0
006552  010177  173520                  2$:     MOV     R1,@STKPT           ;SET STK PTR BEFORE JSR
006556  012777  032342  173460                  MOV     #BUFFER,@DPC        ;START
006564  013737  002272  001122                  MOV     STKVAL,$BDADR       ;SET UP REG ADRS 26
006572  005277  173446                          INC     @DPC                ;ADVANCE TO JSR INSTR
006576  004537  024426                          JSR     R5,DELAY            ;STALL FOR STACKING
006602  000001                                  BIT0                        ;COUNT TO 1
006604  010077  173466                          MOV     R0,@STKPT           ;SELECT WORD & BYTE
006610  017737  173456  001126                  MOV     @STKVAL,$BDDAT      ;READ STOP INTR ENA WORD
006616  042737  177776  001126                  BIC     #177776,$BDDAT      ;SAVE ONLY STOP INTR ENA BIT
006624  023737  001124  001126                  CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
006632  001403                                  BEQ     3$                  ;BR IF SO
006634  010037  001164                          MOV     R0,$TMP0            ;SET UP STK LEVEL FOR ER TYPE
006640  104020                                  ERROR   20                  ;STOP INTR ENA FAILED TO STACK AT LEVEL INDICATED
006642  162701  000004                  3$:     SUB     #4,R1               ;ADVANCE TO NEXT STK LEVEL
006646  162700  000004                          SUB     #4,R0               ;ALL STACK LOCATIONS TESTED?
006652  100337                                  BPL     2$                  ;BR IF NOT
006654  162737  000400  032342                  SUB     #400,BUFFER         ;RESET STOP INTR ENA AT INSTR LOC
006662  005337  001124                          DEC     $GDDAT              ;RESET STOP INTR ENA AT EXPECTED LOC
006666  100325                                  BPL     1$                  ;BR IF RESET STATE NOT TESTED


                                        ;;*******************************************************************
                                        ;*TEST 21       TEST THAT L.P. HIT DISABLE GETS PUSHED ON THE STACK
                                        ;;*******************************************************************
006670  000004                          TST21:  SCOPE
006672  012737  000021  001206                  MOV     #21,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
006700  012737  000002  001124                  MOV     #2,$GDDAT           ;EXPECT IT TO BE SET INITIALLY
006706  012737  170300  032342                  MOV     #170300,BUFFER      ;SET UP HIT DISABLE AT INSTR LOC
006714  012737  163000  032344                  MOV     #163000,BUFFER+2            ;SET UP JSR INSTR
006722  012737  006740  001110                  MOV     #2$,$LPERR          ;SET UP SCOPE LOOP ADRS
006730  012701  020040                  1$:     MOV     #20040,R1           ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
006734  012700  000037                          MOV     #37,R0              ;SET UP STK LEVEL AFTER JSR IN R0
006740  010177  173332                  2$:     MOV     R1,@STKPT           ;SET STK PTR BEFORE JSR
006744  012777  032342  173272                  MOV     #BUFFER,@DPC        ;START
006752  013737  002272  001122                  MOV     STKVAL,$BDADR       ;SET UP REG ADRS 26
006760  005277  173260                          INC     @DPC                ;ADVANCE TO JSR INSTR
006764  004537  024426                          JSR     R5,DELAY            ;STALL FOR STACKING
006770  000001                                  BIT0                        ;COUNT TO 1
006772  010077  173300                          MOV     R0,@STKPT           ;SELECT WORD & BYTE
006776  017737  173270  001126                  MOV     @STKVAL,$BDDAT      ;READ LP HIT DISABLE WORD
007004  042737  177775  001126                  BIC     #177775,$BDDAT      ;SAVE ONLY LP HIT DISABLE BIT
007012  023737  001124  001126                  CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
007020  001403                                  BEQ     3$                  ;BR IF SO
007022  010037  001164                          MOV     R0,$TMP0            ;SET UP STK LEVEL FOR ER TYPE
007026  104021                                  ERROR   21                  ;L.P. HIT DISABLE FAILED TO STACK AT LEVEL INDICATED
007030  162701  000004                  3$:     SUB     #4,R1               ;ADVANCE TO NEXT STK LEVEL
007034  162700  000004                          SUB     #4,R0               ;ALL STACK LOCATIONS TESTED?
```

```
007040  100337                          BPL     2$              ;BR IF NOT
007042  162737  000100  032342          SUB     #100,BUFFER     ;RESET LP HIT DISABLE BIT AT INSTR LOC
007050  162737  000002  001124          SUB     #2,$GDDAT       ;RESET LP HIT DISABLE BIT AT EXPECTED LOC
007056  100324                          BPL     1$              ;BR IF RESET STATE NOT TESTED
                                ;;*****************************************************************
                                ;*TEST 22       TEST THAT STOP INTR & L.P. HIT ENABLES DON'T CHANGE WHEN CHANGE ENA=0
                                ;;*****************************************************************
007060  000004                  TST22:  SCOPE
007062  012737  000022  001206          MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
007070  012737  000003  001124          MOV     #3,$GDDAT       ;EXPECT STOP INTR ENABLED & L.P. HIT FROM STK
007076  012737  171700  032342          MOV     #171700,BUFFER  ;SET UP LOAD STATUS A INSTR
007104  012737  163000  032344          MOV     #163000,BUFFER+2        ;SET UP JSR REL INSTR
007112  012777  020040  173156  1$:     MOV     #20040,@STKPT   ;SET 'TOP OF STACK'
007120  012777  032342  173116          MOV     #BUFFER,@DPC    ;START
007126  013737  002272  001122          MOV     STKVAL,$BDADR   ;SET UP REG ADRS 26
007134  005277  173104                  INC     @DPC            ;PICK UP JSR INSTR
007140  004537  024426                  JSR     R5,DELAY        ;STALL FOR STACKING
007144  000001                          BITO                    ;COUNT TO ONE
007146  012777  020037  173122          MOV     #20037,@STKPT   ;SET STACK PTR
007154  017737  173112  001126          MOV     @STKVAL,$BDDAT  ;READ STACK
007162  042737  177774  001126          BIC     #177774,$BDDAT  ;SAVE ONLY STOP INTR & L.P. HIT
007170  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;SHOULD STAY SET ONCE CHANGE ENABLES ARE OFF
007176  001404                          BEQ     2$              ;BR IF CORRECT
007200  012737  000037  001164          MOV     #37,$TMP0       ;SAVE STACK LEVEL FOR TYPE
007206  104022                          ERROR   22              ;LD STATUS A STROBED WHEN CHANGE ENA=0
007210  022737  171700  032342  2$:     CMP     #171700,BUFFER  ;IS THIS THE 2ND PASS?
007216  001004                          BNE     TST23           ;;GO TO NEXT TEST IF TESTED WITH CHANGE ENA=0
007220  042737  001700  032342          BIC     #1700,BUFFER    ;CLR CHANGE ENABLES FOR 2ND PASS THIS TEST
007226  000731                          BR      1$              ;GO REPEAT TEST WITH CHANGE ENABLES OFF


                                ;;*****************************************************************
                                ;*TEST 23       TEST THAT EDGE INT ENA GETS PUSHED ON THE STACK
                                ;;*****************************************************************
007230  000004                  TST23:  SCOPE
007232  012737  000023  001206          MOV     #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
007240  012737  004000  001124          MOV     #4000,$GDDAT    ;EXPECT IT TO BE SET INITIALLY
007246  012737  176060  032342          MOV     #176060,BUFFER  ;SET UP EDGE INT ENA AT INSTR LOC
007254  012737  163000  032344          MOV     #163000,BUFFER+2        ;SET UP JSR INSTR
007262  012737  007300  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
007270  012701  020040          1$:     MOV     #20040,R1       ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
007274  012700  000037                  MOV     #37,R0          ;SET UP STK LEVEL AFTER JSR IN R0
007300  010177  172772          2$:     MOV     R1,@STKPT       ;SET STK PTR BEFORE JSR
007304  012777  032342  172732          MOV     #BUFFER,@DPC    ;START
007312  013737  002272  001122          MOV     STKVAL,$BDADR   ;SET UP REG ADRS 26
007320  005277  172720                  INC     @DPC            ;ADVANCE TO JSR INSTR
007324  004537  024426                  JSR     R5,DELAY        ;STALL FOR STACKING
007330  000001                          BITO                    ;COUNT TO 1
007332  010077  172740                  MOV     R0,@STKPT       ;SELECT WORD & BYTE
007336  017737  172730  001126          MOV     @STKVAL,$BDDAT  ;READ STACK WORD
007344  042737  173777  001126          BIC     #173777,$BDDAT  ;SAVE ONLY EDGE INT ENA BIT
007352  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
007360  001403                          BEQ     3$              ;BR IF SO
007362  010037  001164                  MOV     R0,$TMP0        ;SET UP STK LEVEL FOR ER TYPE
007366  104023                          ERROR   23              ;EDGE INTR ENA FAILED TO STACK AT LEVEL INDICATED
007370  162701  000004          3$:     SUB     #4,R1           ;ADVANCE TO NEXT STK LEVEL
```

```
007374  162700  000004                  SUB     #4,R0           ;ALL STACK LOCATIONS TESTED?
007400  100337                          BPL     2$              ;BR IF NOT
007402  162737  000020  032342          SUB     #20,BUFFER      ;RESET EDGE INT ENA BIT AT INSTR LOC
007410  162737  004000  001124          SUB     #4000,$GDDAT    ;RESET EDGE INT ENA BIT AT EXPECTED LOC
007416  100324                          BPL     1$              ;BR IF RESET STATE NOT TESTED

                                ;;***************************************************************
                                ;*TEST 24       TEST THAT CHAR STRING ESCAPE BIT GETS PUSHED ON THE STACK
                                ;;***************************************************************
007420  000004                  TST24:  SCOPE
007422  012737  000024  001206          MOV     #24,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
007430  012737  010000  001124          MOV     #10000,$GDDAT   ;EXPECT IT TO BE SET INITIALLY
007436  012737  176003  032342          MOV     #176003,BUFFER  ;SET UP CHAR STRING ESCAPE ENA AT INSTR LOC
007444  012737  163000  032344          MOV     #163000,BUFFER+2        ;SET UP JSR INSTR
007452  012737  007470  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
007460  012701  020040          1$:     MOV     #20040,R1       ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
007464  012700  000037                  MOV     #37,R0          ;SET UP STK LEVEL AFTER JSR IN R0
007470  010177  172602          2$:     MOV     R1,@STKPT       ;SET STK PTR BEFORE JSR
007474  012777  032342  172542          MOV     #BUFFER,@DPC    ;START
007502  013737  002272  001122          MOV     STKVAL,$BDADR   ;SET UP REG ADRS 26
007510  005277  172530                  INC     @DPC            ;ADVANCE TO JSR INSTR
007514  004537  024426                  JSR     R5,DELAY        ;STALL FOR STACKING
007520  000001                          BIT0                    ;COUNT TO 1
007522  010077  172550                  MOV     R0,@STKPT       ;SELECT STACK WORD & BYTE
007526  017737  172540  001126          MOV     @STKVAL,$BDDAT  ;READ STACK WORD
007534  042737  167777  001126          BIC     #167777,$BDDAT  ;SAVE ONLY CHAR STRING ESCAPE BIT
007542  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
007550  001403                          BEQ     3$              ;BR IF OK
007552  010037  001164                  MOV     R0,$TMP0        ;SET UP STK LEVEL FOR ER TYPE
007556  104024                          ERROR   24              ;CHAR STRING ESCAPE FAILED TO STACK AT LEVEL INDICATED
007560  162701  000004          3$:     SUB     #4,R1           ;ADVANCE TO NEXT STK LEVEL
007564  162700  000004                  SUB     #4,R0           ;ALL STACK LOCATIONS TESTED?
007570  100337                          BPL     2$              ;BR IF NOT
007572  005337  032342                  DEC     BUFFER          ;RESET CHAR STRING ESCAPE AT INSTR LOC
007576  162737  010000  001124          SUB     #10000,$GDDAT   ;RESET CHAR STRING ESCAPE AT EXPECTED LOC
007604  100325                          BPL     1$              ;BR IF RESET STATE NOT TESTED

                                ;;***************************************************************
                                ;*TEST 25       TEST THAT DEPTH QUEING BIT GETS PUSHED ON THE STACK
                                ;;***************************************************************
007606  000004                  TST25:  SCOPE
007610  012737  000025  001206          MOV     #25,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
007616  012737  002000  001124          MOV     #2000,$GDDAT    ;EXPECT IT TO BE SET INITIALLY
007624  012737  176014  032342          MOV     #176014,BUFFER  ;SET UP DEPTH QUEING AT INSTR LOC
007632  012737  163000  032344          MOV     #163000,BUFFER+2        ;SET UP JSR REL INSTR
007640  012737  007656  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
007646  012701  020040          1$:     MOV     #20040,R1       ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
007652  012700  000037                  MOV     #37,R0          ;SET UP STK LEVEL AFTER JSR IN R0
007656  010177  172414          2$:     MOV     R1,@STKPT       ;SET STK PTR BEFORE JSR
007662  012777  032342  172354          MOV     #BUFFER,@DPC    ;START
007670  013737  002272  001122          MOV     STKVAL,$BDADR   ;SET UP REG ADRS 26
007676  005277  172342                  INC     @DPC            ;ADVANCE TO JSR INSTR
007702  004537  024426                  JSR     R5,DELAY        ;STALL FOR STACKING
007706  000001                          BIT0                    ;COUNT TO 1
007710  010077  172362                  MOV     R0,@STKPT       ;SELECT STACK WORD & BYTE
007714  017737  172352  001126          MOV     @STKVAL,$BDDAT  ;READ STACK WORD
```

```
007722  042737  175777  001126          BIC     #175777,$BDDAT   ;SAVE ONLY DEPTH QUEING BIT
007730  023737  001124  001126          CMP     $GDDAT,$BDDAT    ;IS IT CORRECT?
007736  001403                          BEQ     3$               ;BR IF SO
007740  010037  001164                  MOV     R0,$TMP0         ;SET UP STK LEVEL FOR ER TYPE
007744  104025                          ERROR   25               ;DEPTH QUEING FAILED TO STACK AT LEVEL INDICATED
007746  162701  000004          3$:     SUB     #4,R1            ;ADVANCE TO NEXT STK LEVEL
007752  162700  000004                  SUB     #4,R0            ;ALL STACK LOCATIONS TESTED?
007756  100337                          BPL     2$               ;BR IF NOT
007760  162737  000004  032342          SUB     #4,BUFFER        ;RESET DEPTH QUEING BIT AT INSTR LOC
007766  162737  002000  001124          SUB     #2000,$GDDAT     ;RESET DEPTH QUEING BIT AT EXPECTED LOC
007774  100324                          BPL     1$               ;BR IF RESET STATE NOT TESTED

                                ;;*********************************************************************
                                ;*TEST 26        TEST THAT THE DEPTH QUE CONTROL GETS PUSHED ON THE STACK
                                ;;*********************************************************************
007776  000004                  TST26:  SCOPE
010000  012737  000026  001206          MOV     #26,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
010006  012737  000400  001124          MOV     #400,$GDDAT      ;EXPECT IT TO BE SET INITIALLY
010014  012737  176300  032342          MOV     #176300,BUFFER   ;SET UP DEPTH QUE CONTROL AT INSTR LOC
010022  012737  163000  032344          MOV     #163000,BUFFER+2         ;SET UP JSR REL INSTR
010030  012737  010046  001110          MOV     #2$,$LPERR       ;SET UP SCOPE LOOP ADRS
010036  012701  020040          1$:     MOV     #20040,R1        ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
010042  012700  000037                  MOV     #37,R0           ;SET UP STK LEVEL AFTER JSR IN R0
010046  010177  172224          2$:     MOV     R1,@STKPT        ;SET STK PTR BEFORE JSR
010052  012777  032342  172164          MOV     #BUFFER,@DPC     ;START
010060  013737  002272  001122          MOV     STKVAL,$BDADR    ;SET UP REG ADRS 26
010066  005277  172152                  INC     @DPC             ;ADVANCE TO JSR INSTR
010072  004537  024426                  JSR     R5,DELAY         ;STALL FOR STACKING
010076  000001                          BIT0                     ;COUNT TO 1
010100  010077  172172                  MOV     R0,@STKPT        ;SEL STK WORD & BYTE
010104  017737  172162  001126          MOV     @STKVAL,$BDDAT   ;READ STACK WORD
010112  042737  177377  001126          BIC     #177377,$BDDAT   ;SAVE ONLY DEPTH QUE CONTROL
010120  023737  001124  001126          CMP     $GDDAT,$BDDAT    ;IS IT CORRECT?
010126  001403                          BEQ     3$               ;BR IF SO
010130  010037  001164                  MOV     R0,$TMP0         ;SET UP STK LEVEL FOR ER TYPE
010134  104025                          ERROR   25               ;DEPTH QUE CONTROL FAILED TO STACK
010136  162701  000004          3$:     SUB     #4,R1            ;ADVANCE TO NEXT STK LEVEL
010142  162700  000004                  SUB     #4,R0            ;ALL STACK LOCATIONS TESTED?
010146  100337                          BPL     2$               ;BR IF NOT
010150  162737  000100  032342          SUB     #100,BUFFER      ;RESET DEPTH QUE CONTROL BIT AT INSTR LOC
010156  162737  000400  001124          SUB     #400,$GDDAT      ;RESET DEPTH QUE CONTROL AT EXPECTED LOC
010164  100324                          BPL     1$               ;BR IF RESET STATE NOT TESTED

                                ;;*********************************************************************
                                ;*TEST 27        TEST THAT BITS SET BY LOAD STATUS B' DON'T CHANGE WHEN CHANGE ENA=0
                                ;;*********************************************************************
010166  000004                  TST27:  SCOPE
010170  012737  000040  001166          MOV     #40,$TIMES       ;;DO 40 ITERATIONS
010176  012737  000027  001206          MOV     #27,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
010204  012737  016400  001124          MOV     #16400,$GDDAT    ;EXPECT ALL LOAD STATUS B' ENABLED BITS FROM STK
010212  012737  176377  032342          MOV     #176377,BUFFER   ;SET UP LOAD STATUS B' INSTR
010220  012737  163000  032344          MOV     #163000,BUFFER+2         ;SET UP JSR REL INSTR
010226  012777  020040  172042  1$:     MOV     #20040,@STKPT    ;SET 'TOP OF STACK'
010234  012777  032342  172002          MOV     #BUFFER,@DPC     ;START
010242  013737  002272  001122          MOV     STKVAL,$BDADR    ;SET UP REG ADRS 26
010250  000240                          NOP                      ;ALLOW TIME FOR NPR
```

```
010252  005277  171766                  INC     @DPC            ;PICK UP JSR INSTR
010256  004537  024426                  JSR     R5,DELAY        ;STALL FOR STACKING
010262  000001                          BIT0                    ;COUNT TO 1
010264  012777  020037  172004          MOV     #20037,@STKPT   ;SET STK PTR
010272  017737  171774  001126          MOV     @STKVAL,$BDDAT  ;READ STACK
010300  042737  161377  001126          BIC     #161377,$BDDAT  ;SAVE ONLY DEPTH QUE, EDGE FLG & CHAR STRING
010306  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;THEY SHOULD STAY SET ONCE CHANGE ENABLES ARE OFF
010314  001404                          BEQ     2$              ;BR IF SO
010316  012737  000037  001164          MOV     #37,$TMP0       ;SAVE STACK LEVEL FOR TYPE
010324  104022                          ERROR   22              ;LD STATUS B' STROBED WHEN CHANGE ENA=0
010326  105737  032342          2$:     TSTB    BUFFER          ;IS THIS THE SECOND PASS THIS TEST?
010332  100004                          BPL     3$              ;GO TO NEXT TEST IF TESTED WITH CHANGE ENA=0
010334  042737  000377  032342          BIC     #377,BUFFER     ;CLR OUT CHANGE ENABLES FOR 2ND PASS THIS TEST
010342  000731                          BR      1$              ;GO REPEAT TEST WITH ENA=0
010344  000005                  3$:     RESET                   ;CLR ENABLES BEFORE ADVANCING

                                ;;*****************************************************************
                                ;*TEST 30        TEST THAT THE LINE TYPE BITS GETS PUSHED ON THE STACK
                                ;;*****************************************************************
010346  000004                  TST30:  SCOPE
010350  012737  000030  001206          MOV     #30,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
010356  012737  000003  001124          MOV     #3,$GDDAT       ;EXPECT BOTH BITS INITIALLY
010364  012737  100007  032342          MOV     #100007,BUFFER  ;SET UP BOTH LINE TYP BITS AT INSTR LOC
010372  012737  163000  032344          MOV     #163000,BUFFER+2        ;SET UP JSR INSTR
010400  012737  010416  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
010406  012701  020040          1$:     MOV     #20040,R1       ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
010412  012700  000036                  MOV     #36,R0          ;SET UP STK LEVEL AFTER JSR IN R0
010416  010177  171654          2$:     MOV     R1,@STKPT       ;SET STK PTR BEFORE JSR
010422  012777  032342  171614          MOV     #BUFFER,@DPC    ;START
010430  013737  002272  001122          MOV     STKVAL,$BDADR   ;SET UP REG ADRS 26
010436  012777  032344  171600          MOV     #BUFFER+2,@DPC  ;START
010444  004537  024426                  JSR     R5,DELAY        ;STALL FOR STACKING
010450  000001                          BIT0                    ;COUNT TO 1
010452  010077  171620                  MOV     R0,@STKPT       ;SELECT STACK WORD & BYTE
010456  017737  171610  001126          MOV     @STKVAL,$BDDAT  ;READ STACK
010464  042737  177774  001126          BIC     #177774,$BDDAT  ;SAVE LINE TYPE ONLY
010472  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY CORRECT?
010500  001403                          BEQ     3$              ;BR IF SO
010502  010037  001164                  MOV     R0,$TMP0        ;SET UP STK LEVEL FOR ER TYPE
010506  104026                          ERROR   26              ;LINE TYPE FAILED TO STACK AT LEVEL INDICATED
010510  162701  000004          3$:     SUB     #4,R1           ;ADVANCE TO NEXT STK LEVEL
010514  162700  000004                  SUB     #4,R0           ;ALL STACK LOCATIONS TESTED?
010520  100336                          BPL     2$              ;BR IF NOT
010522  005337  032342                  DEC     BUFFER          ;ADVANCE LINE TYPE AT INSTR LOC
010526  005337  001124                  DEC     $GDDAT          ;ADVANCE LINE TYPE AT EXPECTED LOC
010532  100325                          BPL     1$              ;BR IF RESET STATE NOT TESTED

                                ;;*****************************************************************
                                ;*TEST 31        TEST THAT INTENSITY ENABLED (CONSOLE 0) GETS PUSHED ON THE STACK
                                ;;*****************************************************************
010534  000004                  TST31:  SCOPE
010536  012737  000031  001206          MOV     #31,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
010544  012737  000004  001124          MOV     #4,$GDDAT       ;EXPECT IT TO BE SET INITIALLY
010552  012737  164300  032342          MOV     #164300,BUFFER  ;SET UP INT EN AT INSTR LOC
010560  012737  163000  032344          MOV     #163000,BUFFER+2        ;SET UP JSR REL INSTR
010566  012737  010604  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
```

```
010574  012701  020040           1$:   MOV    #20040,R1          ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
010600  012700  000037                 MOV    #37,R0             ;SET UP STK LEVEL AFTER JSR IN R0
010604  010177  171466           2$:   MOV    R1,@STKPT          ;SET STK PTR BEFORE JSR
010610  012777  032342  171426         MOV    #BUFFER,@DPC       ;START
010616  013737  002272  001122         MOV    STKVAL,$BDADR      ;SET UP REG ADRS 26
010624  005277  171414                 INC    @DPC               ;ADVANCE TO JSR INSTR
010630  004537  024426                 JSR    R5,DELAY           ;STALL FOR STACKING
010634  000001                         BIT0                      ;COUNT TO 1
010636  010077  171434                 MOV    R0,@STKPT          ;SELECT STK WORD & BYTE
010642  017737  171424  001126         MOV    @STKVAL,$BDDAT     ;READ STK BYTE
010650  042737  177773  001126         BIC    #177773,$BDDAT     ;SAVE ONLY INTENSITY ENABLED BIT
010656  023737  001124  001126         CMP    $GDDAT,$BDDAT      ;IS IT CORRECT?
010664  001403                         BEQ    3$                 ;BR IF OK
010666  010037  001164                 MOV    R0,$TMP0           ;SET UP STK LEVEL FOR ER TYPE
010672  104027                         ERROR  27                 ;INTENSITY ENABLE FAILED TO STACK AT LEVEL INDICATED
010674  162701  000004           3$:   SUB    #4,R1              ;ADVANCE TO NEXT STK LEVEL
010700  162700  000004                 SUB    #4,R0              ;ALL STACK LOCATIONS TESTED?
010704  100337                         BPL    2$                 ;BR IF NOT
010706  162737  000100  032342         SUB    #100,BUFFER        ;SET UP RESET STATE AT INSTR LOC
010714  162737  000004  001124         SUB    #4,$GDDAT          ;SET UP RESET STATE AT EXPECTED LOC
010722  100324                         BPL    1$                 ;BR IF RESET STATE NOT TESTED

                                 ;;*********************************************************************
                                 ;*TEST 32      TEST THAT INTENSITY ENABLED (CONSOLE 1) GETS PUSHED ON THE STACK
                                 ;;*********************************************************************
010724  000004                   TST32: SCOPE
010726  012737  000032  001206         MOV    #32,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
010734  013737  002272  001122         MOV    STKVAL,$BDADR      ;SET UP REG 26 ADRS
010742  012737  000020  001124         MOV    #20,$GDDAT         ;EXPECT IT TO BE SET INITIALLY
010750  012737  164700  032342         MOV    #164700,BUFFER     ;SET UP INT EN AT INSTR LOC
010756  012737  163000  032344         MOV    #163000,BUFFER+2       ;SET UP JSR REL INSTR
010764  012737  011002  001110         MOV    #2$,$LPERR         ;SET UP SCOPE LOOP ADRS
010772  012701  020040           1$:   MOV    #20040,R1          ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
010776  012700  000037                 MOV    #37,R0             ;SET UP STK LEVEL AFTER JSR IN R0
011002  010177  171270           2$:   MOV    R1,@STKPT          ;SET STK PTR BEFORE JSR
011006  012777  032342  171230         MOV    #BUFFER,@DPC       ;START
011014  013737  002272  001122         MOV    STKVAL,$BDADR      ;SET UP REG ADRS 26
011022  005277  171216                 INC    @DPC               ;ADVANCE TO JSR INSTR
011026  004537  024426                 JSR    R5,DELAY           ;STALL FOR STACKING
011032  000001                         BIT0                      ;COUNT TO 1
011034  010077  171236                 MOV    R0,@STKPT          ;SEL STK WORD & BYTE
011040  017737  171226  001126         MOV    @STKVAL,$BDDAT     ;READ STK BYTE
011046  042737  177757  001126         BIC    #177757,$BDDAT     ;SAVE ONLY INTENSITY ENABLED BIT
011054  023737  001124  001126         CMP    $GDDAT,$BDDAT      ;IS IT CORRECT?
011062  001403                         BEQ    3$                 ;BR IF OK
011064  010037  001164                 MOV    R0,$TMP0           ;SET UP STK LEVEL FOR ER TYPE
011070  104027                         ERROR  27                 ;INTENSITY ENABLE FAILED TO STACK AT LEVEL INDICATED
011072  162701  000004           3$:   SUB    #4,R1              ;ADVANCE TO NEXT STK LEVEL
011076  162700  000004                 SUB    #4,R0              ;ALL STACK LOCATIONS TESTED?
011102  100337                         BPL    2$                 ;BR IF NOT
011104  162737  000100  032342         SUB    #100,BUFFER        ;SET UP RESET STATE AT INSTR LOC
011112  162737  000020  001124         SUB    #20,$GDDAT         ;SET UP RESET STATE AT EXPECTED LOC
011120  100324                         BPL    1$                 ;BR IF RESET STATE NOT TESTED

                                 ;;*********************************************************************
                                 ;*TEST 33      TEST THAT L.P. INTR ENABLED (CONSOLE 0) GETS PUSHED ON THE STACK
```

```
                                                ;;***********************************************************
        011122  000004                          TST33:  SCOPE
        011124  012737  000033  001206                  MOV     #33,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
        011132  012737  000010  001124                  MOV     #10,$GDDAT              ;EXPECT IT TO BE SET INITIALLY
        011140  012737  164060  032342                  MOV     #164060,BUFFER          ;SET UP L.P. INTR EN AT INSTR LOC
        011146  012737  163000  032344                  MOV     #163000,BUFFER+2                ;SET UP JSR INSTR
        011154  012737  011172  001110                  MOV     #2$,$LPERR              ;SET UP SCOPE LOOP ADRS
        011162  012701  020040                  1$:     MOV     #20040,R1               ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
        011166  012700  000037                          MOV     #37,R0                  ;SET UP STK LEVEL AFTER JSR IN R0
        011172  010177  171100                  2$:     MOV     R1,@STKPT               ;SET STK PTR BEFORE JSR
        011176  012777  032342  171040                  MOV     #BUFFER,@DPC            ;START
        011204  013737  002272  001122                  MOV     STKVAL,$BDADR           ;SET UP REG ADRS 26
        011212  005277  171026                          INC     @DPC                    ;ADVANCE TO JSR INSTR
        011216  004537  024426                          JSR     R5,DELAY                ;STALL FOR STACKING
        011222  000001                                  BIT0                            ;COUNT TO 1
        011224  010077  171046                          MOV     R0,@STKPT               ;SEL STK WORD & BYTE
        011230  017737  171036  001126                  MOV     @STKVAL,$BDDAT          ;READ STK BYTE
        011236  042737  177767  001126                  BIC     #177767,$BDDAT          ;SAVE ONLY L.P. INTR ENABLED BIT
        011244  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;IS IT CORRECT?
        011252  001403                                  BEQ     3$                      ;BR IF SO
        011254  010037  001164                          MOV     R0,$TMP0                ;SET UP STK LEVEL FOR ER TYPE
        011260  104030                                  ERROR   30                      ;L.P. INTR ENABLE FAILED TO STACK AT LEVEL INDICATED
        011262  162701  000004                  3$:     SUB     #4,R1                   ;ADVANCE TO NEXT STK LEVEL
        011266  162700  000004                          SUB     #4,R0                   ;ALL STACK LOCATIONS TESTED?
        011272  100337                                  BPL     2$                      ;BR IF NOT
        011274  162737  000020  032342                  SUB     #20,BUFFER              ;SET UP RESET STATE AT INSTR LOC
        011302  162737  000010  001124                  SUB     #10,$GDDAT              ;SET UP RESET STATE AT EXPECTED LOC
        011310  100324                                  BPL     1$                      ;BR IF RESET STATE NOT TESTED


                                                ;;***********************************************************
                                                ;;TEST 34       TEST THAT L.P. INTR ENABLED (CONSOLE 1) GETS PUSHED ON THE STACK
                                                ;;***********************************************************
        011312  000004                          TST34:  SCOPE
        011314  012737  000034  001206                  MOV     #34,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
        011322  012737  000040  001124                  MOV     #40,$GDDAT              ;EXPECT IT TO BE SET INITIALLY
        011330  012737  164460  032342                  MOV     #164460,BUFFER          ;SET UP L.P. INTR EN AT INSTR LOC
        011336  012737  163000  032344                  MOV     #163000,BUFFER+2                ;SET UP JSR REL INSTR
        011344  012737  011362  001110                  MOV     #2$,$LPERR              ;SET UP SCOPE LOOP ADRS
        011352  012701  020040                  1$:     MOV     #20040,R1               ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
        011356  012700  000037                          MOV     #37,R0                  ;SET UP STK LEVEL AFTER JSR IN R0
        011362  010177  170710                  2$:     MOV     R1,@STKPT               ;SET STK PTR BEFORE JSR
        011366  012777  032342  170650                  MOV     #BUFFER,@DPC            ;START
        011374  013737  002272  001122                  MOV     STKVAL,$BDADR           ;SET UP REG ADRS 26
        011402  005277  170636                          INC     @DPC                    ;ADVANCE TO JSR INSTR
        011406  004537  024426                          JSR     R5,DELAY                ;STALL FOR STACKING
        011412  000001                                  BIT0                            ;COUNT TO 1
        011414  010077  170656                          MOV     R0,@STKPT               ;SEL STK WORD & BYTE
        011420  017737  170646  001126                  MOV     @STKVAL,$BDDAT          ;READ STK BYTE
        011426  042737  177737  001126                  BIC     #177737,$BDDAT          ;SAVE ONLY L.P. INTR ENABLED BIT
        011434  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;IS IT CORRECT?
        011442  001403                                  BEQ     3$                      ;BR IF SO
        011444  010037  001164                          MOV     R0,$TMP0                ;SET UP STK LEVEL FOR ER TYPE
        011450  104030                                  ERROR   30                      ;L.P. INTR ENABLE FAILED TO STACK AT LEVEL INDICATED
        011452  162701  000004                  3$:     SUB     #4,R1                   ;ADVANCE TO NEXT STK LEVEL
        011456  162700  000004                          SUB     #4,R0                   ;ALL STACK LOCATIONS TESTED?
        011462  100337                                  BPL     2$                      ;BR IF NOT
```

```
011464  162737  000020  032342             SUB     #20,BUFFER       ;SET UP RESET STATE AT INSTR LOC
011472  162737  000040  001124             SUB     #40,$GDDAT       ;SET UP RESET STATE AT EXPECTED LOC
011500  100324                             BPL     1$               ;BR IF RESET STATE NOT TESTED

                                   ;;***********************************************************************
                                   ;*TEST 35        TEST THAT L.P. SWITCH INTR ENABLED (CONSOLE 0) GETS PUSHED ON THE STACK
                                   ;;***********************************************************************
011502  000004                     TST35:  SCOPE
011504  012737  000035  001206             MOV     #35,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
011512  012737  000100  001124             MOV     #100,$GDDAT      ;EXPECT IT TO BE SET INITIALLY
011520  012737  164014  032342             MOV     #164014,BUFFER   ;SET UP L.P. SW INTR EN AT INSTR LOC
011526  012737  163000  032344             MOV     #163000,BUFFER+2         ;SET UP JSR REL INSTR
011534  012737  011552  001110             MOV     #2$,$LPERR       ;SET UP SCOPE LOOP ADRS
011542  012701  020040             1$:     MOV     #20040,R1        ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
011546  012700  000037                     MOV     #37,R0           ;SET UP STK LEVEL AFTER JSR IN R0
011552  010177  170520             2$:     MOV     R1,@STKPT        ;SET STK PTR BEFOE JSR
011556  012777  032342  170460             MOV     #BUFFER,@DPC     ;START
011564  013737  002272  001122             MOV     STKVAL,$BDADR    ;SET UP REG ADRS 26
011572  005277  170446                     INC     @DPC             ;ADVANCE TO JSR INSTR
011576  004537  024426                     JSR     R5,DELAY         ;STALL FOR STACKING
011602  000001                             BIT0                     ;COUNT TO 1
011604  010077  170466                     MOV     R0,@STKPT        ;SEL STK WORD & BYTE
011610  017737  170456  001126             MOV     @STKVAL,$BDDAT   ;READ STK BYTE
011616  042737  177677  001126             BIC     #177677,$BDDAT   ;SAVE ONLY L.P. SW INTR ENABLED BIT
011624  023737  001124  001126             CMP     $GDDAT,$BDDAT    ;IS IT CORRECT?
011632  001403                             BEQ     3$               ;BR IF OK
011634  010037  001164                     MOV     R0,$TMP0         ;SET UP STK LEVEL FOR ER TYPE
011640  104031                             ERROR   31               ;L.P. SW INTR ENABLE FAILED TO STACK AT LEVEL INDICATED
011642  162701  000004             3$:     SUB     #4,R1            ;ADVANCE TO NEXT STK LEVEL
011646  162700  000004                     SUB     #4,R0            ;ALL STACK LOCATIONS TESTED?
011652  100337                             BPL     2$               ;BR IF NOT
011654  162737  000004  032342             SUB     #4,BUFFER        ;SET UP RESET STATE AT INSTR. LOC.
011662  162737  000100  001124             SUB     #100,$GDDAT      ;SET UP RESET STATE AT EXPECTED LOC
011670  100324                             BPL     1$               ;BR IF RESET STATE NOT TESTED

                                   ;;***********************************************************************
                                   ;*TEST 36        TEST THAT L.P. SWITCH INTR ENABLED (CONSOLE 1) GETS PUSHED ON THE STACK
                                   ;;***********************************************************************
011672  000004                     TST36:  SCOPE
011674  012737  000036  001206             MOV     #36,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
011702  012737  000200  001124             MOV     #200,$GDDAT      ;EXPECT IT TO BE SET INITIALLY
011710  012737  164414  032342             MOV     #164414,BUFFER   ;SET UP L.P. SW INSTR EN AT INSTR LOC
011716  012737  163000  032344             MOV     #163000,BUFFER+2         ;SET UP JSR REL INSTR
011724  012737  011742  001110             MOV     #2$,$LPERR       ;SET UP SCOPE LOOP ADRS
011732  012701  020040             1$:     MOV     #20040,R1        ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
011736  012700  000037                     MOV     #37,R0           ;SET UP STK LEVEL AFTER JSR IN R0
011742  010177  170330             2$:     MOV     R1,@STKPT        ;SET STK PTR BEFORE JSR
011746  012777  032342  170270             MOV     #BUFFER,@DPC     ;START
011754  013737  002272  001122             MOV     STKVAL,$BDADR    ;SET UP REG ADRS 26
011762  005277  170256                     INC     @DPC             ;ADVANCE TO JSR INSTR
011766  004537  024426                     JSR     R5,DELAY         ;STALL FOR STACKING
011772  000001                             BIT0                     ;COUNT TO 1
011774  010077  170276                     MOV     R0,@STKPT        ;SEL STK WORD & BYTE
012000  017737  170266  001126             MOV     @STKVAL,$BDDAT   ;READ STK BYTE
012006  042737  177577  001126             BIC     #177577,$BDDAT   ;SAVE ONLY L.P. SW INTR ENABLED BIT
012014  023737  001124  001126             CMP     $GDDAT,$BDDAT    ;IS IT CORRECT?
```

```
     012022  001403                            BEQ     3$                 ;BR IF OK
     012024  010037  001164                    MOV     R0,$TMP0           ;SET UP STK LEVEL FOR ER TYPE
     012030  104031                            ERROR   31                 ;L.P. SW INTR ENABLE FAILED TO STACK AT LEVEL INDICATED
     012032  162701  000004            3$:     SUB     #4,R1              ;ADVANCE TO NEXT STK LEVEL
     012036  162700  000004                    SUB     #4,R0              ;ALL STACK LOCATIONS TESTED?
     012042  100337                            BPL     2$                 ;BR IF NOT
     012044  162737  000004  032342            SUB     #4,BUFFER          ;SET UP RESET STATE AT INSTR LOC
     012052  162737  000200  001124            SUB     #200,$GDDAT        ;SET UP RESET STATE AT EXPECTED LOC
     012060  100324                            BPL     1$                 ;BR IF RESET STATE NOT TESTED

                                       ;;*********************************************************
                                       ;*TEST 37        TEST THAT THE SHIFT OUT BIT GETS PUSHED ON THE STACK
                                       ;;*********************************************************
     012062  000004                    TST37:  SCOPE
     012064  012737  000037  001206            MOV     #37,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
     012072  012737  040000  001124            MOV     #40000,$GDDAT      ;EXPECT IT TO BE SET INITIALLY
     012100  012737  100000  032342            MOV     #100000,BUFFER     ;SET UP CHAR INSTR
     012106  012737  007000  032344            MOV     #7000,BUFFER+2     ;SET CHAR TO SHIFT OUT
     012114  012737  163000  032346            MOV     #163000,BUFFER+4        ;SET UP JSR REL INSTR
     012122  012737  012140  001110            MOV     #2$,$LPERR         ;SET UP SCOPE LOOP ADRS
     012130  012701  020040            1$:     MOV     #20040,R1          ;SET TOP OF STACK & MAINT SW2 INITIALLY IN R1
     012134  012700  000037                    MOV     #37,R0             ;SET UP STK LEVEL AFTER JSR IN R0
     012140  010177  170132            2$:     MOV     R1,@STKPT          ;SET STK PTR BEFORE JSR
     012144  012777  032342  170072            MOV     #BUFFER,@DPC       ;START
     012152  013737  002272  001122            MOV     STKVAL,$BDADR      ;SET UP REG ADRS 26
     012160  005277  170060                    INC     @DPC               ;RESUME
     012164  004537  024426                    JSR     R5,DELAY           ;STALL FOR CHAR DISPLAY
     012170  000010                            BIT3                       ;COUNT TO 10
     012172  005277  170046                    INC     @DPC               ;RESUME
     012176  004537  024426                    JSR     R5,DELAY           ;STALL FOR STKING
     012202  000001                            BIT0                       ;COUNT TO 1
     012204  010077  170066                    MOV     R0,@STKPT          ;SEL STK WD & BYTE
     012210  017737  170056  001126            MOV     @STKVAL,$BDDAT     ;READ STK
     012216  042737  137777  001126            BIC     #137777,$BDDAT     ;SAVE SHIFT OUT BIT ONLY
     012224  023737  001124  001126            CMP     $GDDAT,$BDDAT      ;IS IT CORRECT?
     012232  001403                            BEQ     3$                 ;BR IF SO
     012234  010037  001164                    MOV     R0,$TMP0           ;SET UP STK LEVEL FOR ER TYPE
     012240  104032                            ERROR   32                 ;SHIFT OUT FAILED TO STACK AT LEVEL INDICATED
     012242  162701  000004            3$:     SUB     #4,R1              ;ADVANCE TO NEXT STK LEVEL
     012246  162700  000004                    SUB     #4,R0              ;ALL STACK LOCATIONS TESTED?
     012252  100332                            BPL     2$                 ;BR IF MORE STK LOC TO TEST
     012254  012737  000017  032344            MOV     #17,BUFFER+2       ;PROVIDE CHAR THAT WON'T SET SHIFT OUT
     012262  162737  040000  001124            SUB     #40000,$GDDAT      ;EXPECT ZERO STATE NEXT
     012270  100317                            BPL     1$                 ;BR IF RESET STATE NOT TESTED

                                       ;;*********************************************************
                                       ;*TEST 40        TEST THAT THE 'POP' INSTR WILL INCREMENT THE STACK POINTER
                                       ;;*********************************************************
     012272  000004                    TST40:  SCOPE
     012274  012737  000040  001206            MOV     #40,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
     012302  012777  020000  167766            MOV     #20000,@STKPT      ;SET STACK POINTER TO ZERO
     012310  012737  000004  001124            MOV     #4,$GDDAT          ;EXPECT STK PTR TO BE 1 INITIALLY
     012316  012737  165000  032342            MOV     #165000,BUFFER     ;SET UP POP INSTR
     012324  012777  032342  167712    1$:     MOV     #BUFFER,@DPC       ;START
     012332  013737  002276  001122            MOV     STKPT,$BDADR       ;SET UP REG ADRS 32
     012340  017737  167732  001126            MOV     @STKPT,$BDDAT      ;READ STACK SELECTION
```

```
012346  042737  177740  001126          BIC     #177740,$BDDAT   ;SAVE ONLY STK SEL BITS
012354  023737  001124  001126          CMP     $GDDAT,$BDDAT    ;ARE THEY CORRECT?
012362  001401                          BEQ     2$               ;BR IF SO
012364  104033                          ERROR   33               ;POP INSTR FAILED TO INCREMENT STACK PTR
012366  062737  000004  001124  2$:     ADD     #4,$GDDAT        ;ADVANCE EXPECTED STACK POINTER VALUE
012374  022737  000040  001124          CMP     #40,$GDDAT       ;HAVE DONE ALL STK LOCS?
012402  001350                          BNE     1$               ;BR IF NOT

                                ;;*****************************************************************
                                ;*TEST 41        TEST THAT STACK UNDERFLOW WILL SET
                                ;;*****************************************************************
012404  000004                  TST41:  SCOPE
012406  012737  000040  001166          MOV     #40,$TIMES       ;;DO 40 ITERATIONS
012414  012737  000041  001206          MOV     #41,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
012422  012777  020000  167646          MOV     #20000,@STKPT    ;SET STACK TO ZERO
012430  005037  001124                  CLR     $GDDAT           ;EXPECT NO UNDERFLOW FOR 8 POPS
012434  012700  000011                  MOV     #11,R0           ;SET UP POP COUNT
012440  012737  165000  032342          MOV     #165000,BUFFER   ;SET UP POP INSTR
012446  012777  032342  167570  1$:     MOV     #BUFFER,@DPC     ;START
012454  013737  002256  001122          MOV     SREG1,$BDADR     ;SET UP REG ADRS 12
012462  005300                          DEC     R0               ;COUNT POP OPERATION
012464  001410                          BEQ     2$               ;BR IF UNDERFLOW EXPECTED
012466  032777  010000  167562          BIT     #10000,@SREG1    ;SEE THAT UNDERFLOW  NOT SET
012474  001764                          BEQ     1$               ;BR IF OK
012476  012737  010000  001126          MOV     #10000,$BDDAT    ;INDICATE STACK UNDERFLOW
012504  104034                          ERROR   34               ;STACK UNDERFLOW SET PREMATURELY
012506  032777  010000  167542  2$:     BIT     #10000,@SREG1    ;SEE THAT UNDERFLOW IS SET
012514  001006                          BNE     3$               ;BR IF SO
012516  012737  010000  001124          MOV     #10000,$GDDAT    ;EXPECTED STACK UNDERFLOW
012524  005037  001126                  CLR     $BDDAT           ;INDICATE IT WAS NOT THERE
012530  104034                          ERROR   34               ;STACK UNDERFLOW FAILED TO SET
012532  000005                  3$:     RESET                    ;RESET STACK UNDERFLOW INT REQ BEFORE ADVANCING

                                ;;*****************************************************************
                                ;*TEST 42        TEST THAT THE POP INSTR CAN RESTORE THE DPC
                                ;;*****************************************************************
012534  000004                  TST42:  SCOPE
012536  012737  000042  001206          MOV     #42,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
012544  013737  002244  001122          MOV     DPC,$BDADR       ;SET UP REG 00 ADRS
012552  012737  032344  001124          MOV     #BUFFER+2,$GDDAT         ;EXPECT BUFFER+2 ADRS AFTER POP
012560  012777  020040  167510          MOV     #20040,@STKPT    ;SET TOP OF STACK
012566  012737  163000  032342          MOV     #163000,BUFFER   ;SET UP JSR REL INSTR
012574  012737  166000  032352          MOV     #166000,BUFFER+10        ;SET UP POP INSTR
012602  012777  032342  167434          MOV     #BUFFER,@DPC     ;START
012610  004537  024426                  JSR     R5,DELAY         ;STALL FOR STACKING
012614  000001                          BIT0                     ;COUNT TO 1
012616  012777  032352  167420          MOV     #BUFFER+10,@DPC  ;START
012624  004537  024426                  JSR     R5,DELAY         ;STALL FOR UNSTACKING
012630  000001                          BIT0                     ;COUNT TO 1
012632  017737  167406  001126          MOV     @DPC,$BDDAT      ;READ DPC
012640  023737  001124  001126          CMP     $GDDAT,$BDDAT    ;DID THE DPC GET RESTORED OK
012646  001401                          BEQ     TST43            ;;ADVANCE TO NEXT TEST IF OK
012650  104035                          ERROR   35               ;POP INSTR FAILED TO RESTORE DPC FROM STACK

                                ;;*****************************************************************
                                ;*TEST 43        TEST THAT THE NAME BITS WILL GET RESTORED
```

```
                                        ;;*******************************************************************
012652  000004                          TST43:  SCOPE
012654  012737  000004  001166                  MOV     #4,$TIMES               ;;DO 4 ITERATIONS
012662  012737  000043  001206                  MOV     #43,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
012670  012737  012720  001110                  MOV     #1$,$LPERR              ;SET UP SCOPE LOOP ADRS
012676  013737  002270  001122                  MOV     DNAME,$BDADR            ;SET UP REG 24 ADRS
012704  012737  003777  001124                  MOV     #3777,$GDDAT            ;START NAME REG WITH MAXIMUM VALUE
012712  004437  024442                          JSR     R4,POPSET               ;SET UP JSR, POP & NAME INSTRUCTIONS
012716  150000                                  150000                          ;NAME INSTR
012720  012737  150000  032342          1$:     MOV     #150000,BUFFER          ;SET UP NAME INSTR
012726  053737  001124  032342                  BIS     $GDDAT,BUFFER           ;SET NAME WORD
012734  004737  024472                          JSR     PC,DO                   ;GO CYCLE THRU INSTRUCTIONS
012740  017737  167324  001126                  MOV     @DNAME,$BDDAT           ;READ NAME REG
012746  042737  174000  001126                  BIC     #174000,$BDDAT          ;SAVE ONLY NAME BITS
012754  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;DID THEY GET RESTORED?
012762  001401                                  BEQ     2$                      ;BR IF SO
012764  104036                                  ERROR   36                      ;NAME FAILED TO RESTORE
012766  005337  001124                  2$:     DEC     $GDDAT                  ;ADVANCE PATTERN
012772  100405                                  BMI     TST44                   ;;ADVANCE TO NEXT TEST IF ALL VALUES TESTED
012774  001351                                  BNE     1$                      ;BR IF NAME BITS NOT EQ TO ZERO
012776  052737  003777  032346                  BIS     #3777,BUFFER+4          ;SET NAME REG TO ALL ONES WHEN RESTORING ZERO
013004  000745                                  BR      1$                      ;GO RESTORE ALL ZEROS

                                        ;;*******************************************************************
                                        ;*TEST 44       TEST THAT THE VECTOR SCALE WILL GET RESTORED
                                        ;;*******************************************************************
013006  000004                          TST44:  SCOPE
013010  012737  000100  001166                  MOV     #100,$TIMES             ;;DO 100 ITERATIONS
013016  012737  000044  001206                  MOV     #44,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
013024  012737  013054  001110                  MOV     #1$,$LPERR              ;SET UP SCOPE LOOP ADRS
013032  013737  002256  001122                  MOV     SREG1,$BDADR            ;SET UP REG 12 ADRS
013040  012737  000017  001124                  MOV     #17,$GDDAT              ;START VECTOR SCALE AT MAXIMUM VALUE
013046  004437  024442                          JSR     R4,POPSET               ;SET UP JSR,POP & LOAD STATUS C INSTRS
013052  154020                                  154020                          ;LOAD STATUS C INSTR
013054  012737  154020  032342          1$:     MOV     #154020,BUFFER          ;SET UP LOAD STATUS C INSTR
013062  053737  001124  032342                  BIS     $GDDAT,BUFFER           ;SET VECTOR SCALE
013070  004737  024472                          JSR     PC,DO                   ;GO CYCLE THRU INSTRUCTIONS
013074  017737  167156  001126                  MOV     @SREG1,$BDDAT           ;READ VECTOR SCALE REG
013102  042737  177760  001126                  BIC     #177760,$BDDAT          ;SAVE ONLY THE VECTOR SCALE BITS
013110  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;DID THEY GET RESTORED?
013116  001401                                  BEQ     2$                      ;BR IF SO
013120  104037                                  ERROR   37                      ;VECTOR SCALE FAILED TO RESTORE
013122  005337  001124                  2$:     DEC     $GDDAT                  ;ADVANCE PATTERN
013126  100405                                  BMI     TST45                   ;;ADVANCE TO NEXT TEST IF ALL VALUES TESTED
013130  001351                                  BNE     1$                      ;BR IF VECTOR SCALE NOT EQ TO ZERO
013132  052737  000017  032346                  BIS     #17,BUFFER+4            ;SET VECTOR SCALE TO ALL ONES WHEN RESTORING ZEROS
013140  000745                                  BR      1$                      ;GO RESTORE ALL ZEROS

                                        ;;*******************************************************************
                                        ;*TEST 45       TEST THAT THE CHAR SCALE WILL GET RESTORED
                                        ;;*******************************************************************
013142  000004                          TST45:  SCOPE
013144  012737  000045  001206                  MOV     #45,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
013152  012737  013206  001110                  MOV     #1$,$LPERR              ;SET UP SCOPE LOOP ADRS
013160  013737  002256  001122                  MOV     SREG1,$BDADR            ;SET UP REG 12 ADRS
013166  012737  001400  001124                  MOV     #1400,$GDDAT            ;EXPECT CHAR SCALE AT MAXIMUM VALUE
```

```
 013174  012700  000140                   MOV     #140,R0          ;START CHAR SCALE AT MAXIMUM VALUE
 013200  004437  024442                   JSR     R4,POPSET        ;SET UP JSR,POP & LOAD STATUS C INSTR
 013204  154200                           154200                   ;LOAD STATUS C INSTR
 013206  012737  154200  032342   1$:     MOV     #154200,BUFFER   ;SET UP LOAD STATUS C INSTR
 013214  050037  032342                   BIS     R0,BUFFER        ;SET CHAR SCALE
 013220  004737  024472                   JSR     PC,DO            ;GO CYCLE THRU INSTRUCTIONS
 013224  017737  167026  001126           MOV     @SREG1,$BDDAT    ;READ CHAR SCALE REG
 013232  042737  176377  001126           BIC     #176377,$BDDAT   ;SAVE ONLY THE CHAR SCALE BITS
 013240  023737  001124  001126           CMP     $GDDAT,$BDDAT    ;DID THE CHAR SCALE GET RESTORED OK?
 013246  001401                           BEQ     2$               ;BR IF SO
 013250  104040                           ERROR   40               ;CHARACTER SCALE FAILED TO RESTORE
 013252  162737  000400  001124   2$:     SUB     #400,$GDDAT      ;ADVANCE CHAR SCALE VALUE
 013260  162700  000040                   SUB     #40,R0           ;SET UP R0 WITH NEXT VALUE
 013264  100405                           BMI     TST46            ;;ADVANCE TO NEXT TEST IF ALL VALUES TESTED
 013266  001347                           BNE     1$               ;GO TRY NEXT VALUE IF CHAR SCALE NOT EQ ZERO
 013270  052737  000140  032346           BIS     #140,BUFFER+4    ;SET CHAR SCALE TO ALL ONES WHEN RETORING ZEROS
 013276  000743                           BR      1$               ;GO RESTORE ALL ZEROS

                                  ;;*************************************************************
                                  ;*TEST 46        TEST THAT THE CHAR ROTATE BIT GETS RESTORED
                                  ;;*************************************************************
 013300  000004                   TST46:  SCOPE
 013302  012737  000046  001206           MOV     #46,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
 013310  012737  013346  001110           MOV     #1$,$LPERR       ;SET UP SCOPE LOOP ADRS
 013316  013737  002256  001122           MOV     SREG1,$BDADR     ;SET UP REG 12 ADRS
 013324  012737  002000  001124           MOV     #2000,$GDDAT     ;EXPECT CHAR ROTATE TO BE SET INITIALLY
 013332  004437  024442                   JSR     R4,POPSET        ;SET UP JSR,POP & LOAD STATUS C INSTR
 013336  155000                           155000                   ;LOAD STATUS C INSTR
 013340  012737  155400  032342           MOV     #155400,BUFFER   ;SET UP LOAD STATUS C INSTR
 013346  004737  024472           1$:     JSR     PC,DO            ;GO CYCLE THRU INSTRUCTIONS
 013352  017737  166700  001126           MOV     @SREG1,$BDDAT    ;READ CHAR ROTATE REG
 013360  042737  175777  001126           BIC     #175777,$BDDAT   ;SAVE ONLY CHAR ROTATE BIT
 013366  023737  001124  001126           CMP     $GDDAT,$BDDAT    ;IS IT CORRECT?
 013374  001401                           BEQ     2$               ;BR IF SO
 013376  104041                           ERROR   41               ;CHARACTER ROTATE FAILED TO RESTORE
 013400  162737  002000  001124   2$:     SUB     #2000,$GDDAT     ;CLR CHAR ROTATE
 013406  100407                           BMI     TST47            ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
 013410  042737  000400  032342           BIC     #400,BUFFER      ;SET UP FOR RESET STATE
 013416  052737  000400  032346           BIS     #400,BUFFER+4    ;SET CHAR ROTATE BIT
 013424  000750                           BR      1$               ;GO TRY RESET STATE

                                  ;;*************************************************************
                                  ;*TEST 47        TEST THAT THE COLOR LEVEL WILL GET RESTORED
                                  ;;*************************************************************
 013426  000004                   TST47:  SCOPE
 013430  012737  000047  001206           MOV     #47,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
 013436  012737  013472  001110           MOV     #1$,$LPERR       ;SET UP SCOPE LOOP ADRS
 013444  013737  002266  001122           MOV     CONS,$BDADR      ;SET UP REG 22 ADRS
 013452  012737  000014  001124           MOV     #14,$GDDAT       ;EXPECT MAX VALUE INITIALLY
 013460  012700  000600                   MOV     #600,R0          ;SET UP COLOR LEVEL FOR INSTR USE
 013464  004437  024442                   JSR     R4,POPSET        ;SET UP JSR,POP & LOAD STATUS B INSTR
 013470  175000                           175000                   ;LOAD STATUS B INSTR
 013472  012737  175000  032342   1$:     MOV     #175000,BUFFER   ;SET UP LOAD STATUS B INSTR
 013500  050037  032342                   BIS     R0,BUFFER        ;SET UP COLOR LEVEL
 013504  004737  024472                   JSR     PC,DO            ;GO CYCLE THRU INSTRUCTIONS
 013510  017737  166552  001126           MOV     @CONS,$BDDAT     ;READ COLOR LEVEL REG
```

```
 013516  042737  177763  001126             BIC     #177763,$BDDAT      ;SAVE COLOR LEVEL ONLY
 013524  023737  001124  001126             CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
 013532  001401                             BEQ     2$                  ;BR IF SO
 013534  104042                             ERROR   42                  ;COLOR LEVEL FAILED TO RESTORE
 013536  162737  000004  001124     2$:     SUB     #4,$GDDAT           ;ADVANCE TO NEXT VALUE
 013544  162700  000200                     SUB     #200,R0             ;SET UP R4 WITH NEXT VALUE
 013550  100405                             BMI     TST50               ;;ADVANCE TO NEXT TEST IF ALL COLOR LEVELS TESTED
 013552  001347                             BNE     1$                  ;GO TEST NEXT COLOR LEVEL IF NOT ZERO
 013554  052737  000600  032346             BIS     #600,BUFFER+4       ;SET COLOR LEVEL TO ALL ONES WHEN RESTORING ZEROS
 013562  000743                             BR      1$                  ;GO RESTORE ALL ZEROS

                                    ;;*****************************************************************
                                    ;*TEST 50       TEST THAT THE ITALICS BIT GETS RETORED
                                    ;;*****************************************************************
 013564  000004                     TST50:  SCOPE
 013566  012737  000050  001206             MOV     #50,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
 013574  012737  013624  001110             MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
 013602  013737  002246  001122             MOV     SREG0,$BDADR        ;SET UP REG 02 ADRS
 013610  012737  000020  001124             MOV     #20,$GDDAT          ;EXPECT ITALICS INITIALLY
 013616  004437  024442                     JSR     R4,POPSET           ;SET UP JSR,POP & LOAD STATUS A
 013622  170040                             170040                      ;LOAD STATUS A INSTR
 013624  012737  170040  032342     1$:     MOV     #170040,BUFFER      ;SET UP LOAD STATUS A INSTR
 013632  053737  001124  032342             BIS     $GDDAT,BUFFER       ;SET UP ITALICS
 013640  004737  024472                     JSR     PC,DO               ;GO CYCLE THRU INSTRUCTIONS
 013644  017737  166376  001126             MOV     @SREG0,$BDDAT       ;READ ITALICS REG
 013652  042737  177757  001126             BIC     #177757,$BDDAT      ;SAVE ONLY ITALICS
 013660  023737  001124  001126             CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
 013666  001401                             BEQ     2$                  ;BR IF SO
 013670  104043                             ERROR   43                  ;ITALICS FAILED TO RESTORE
 013672  162737  000020  001124     2$:     SUB     #20,$GDDAT          ;GO TO RESET STATE
 013700  100404                             BMI     TST51               ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
 013702  052737  000020  032346             BIS     #20,BUFFER+4        ;SET ITALICS BIT TO ONE WHEN RESTORING ZERO
 013710  000745                             BR      1$                  ;GO RESTORE RESET STATE

                                    ;;*****************************************************************
                                    ;*TEST 51       TEST THAT THE MENU BIT GETS RESTORED
                                    ;;*****************************************************************
 013712  000004                     TST51:  SCOPE
 013714  012737  000051  001206             MOV     #51,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
 013722  012737  013760  001110             MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
 013730  013737  002256  001122             MOV     SREG1,$BDADR        ;SET UP REG 12 ADRS
 013736  012737  000100  001124             MOV     #100,$GDDAT         ;EXPECT MENU INITIALLY
 013744  004437  024442                     JSR     R4,POPSET           ;SET UP JSR,POP & LOAD STATUS A INSTR
 013750  170002                             170002                      ;LOAD STATUS A INSTR
 013752  012737  170003  032342             MOV     #170003,BUFFER      ;SET UP LOAD STATUS A INSTR
 013760  004737  024472             1$:     JSR     PC,DO               ;GO CYCLE THRU INSTRUCTIONS
 013764  017737  166266  001126             MOV     @SREG1,$BDDAT       ;READ MENU REG
 013772  042737  177677  001126             BIC     #177677,$BDDAT      ;SAVE ONLY THE MENU BIT
 014000  023737  001124  001126             CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
 014006  001401                             BEQ     2$                  ;BR IF SO
 014010  104044                             ERROR   44                  ;MENU FAILED TO RESTORE
 014012  162737  000100  001124     2$:     SUB     #100,$GDDAT         ;RESET MENU BIT
 014020  100407                             BMI     TST52               ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
 014022  042737  000001  032342             BIC     #1,BUFFER           ;RESET MENU FOR INSTR USE
 014030  052737  000001  032346             BIS     #1,BUFFER+4         ;SET MENU WHEN RESTORING ZERO
 014036  000750                             BR      1$                  ;GO RESTORE ZERO
```

```

                                        ;;*****************************************************************
                                        ;*TEST 52        TEST THAT THE INTENSITY ENA GETS RESTORED ON CONSOLE 0
                                        ;;*****************************************************************
 014040  000004                         TST52:  SCOPE
 014042  012737  000052  001206                 MOV     #52,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 014050  012737  014106  001110                 MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
 014056  013737  002266  001122                 MOV     CONS,$BDADR     ;SET UP REG 22 ADRS
 014064  012737  100000  001124                 MOV     #100000,$GDDAT  ;EXPECT INTENSITY ENABLED TO BE SET INITIALLY
 014072  004437  024442                         JSR     R4,POPSET       ;SET UP JSR, POP & NOP INSTRS
 014076  164200                                 164200                  ;NOP INSTR
 014100  012737  164300  032342                 MOV     #164300,BUFFER  ;SET UP NOP INSTR
 014106  004737  024472                 1$:     JSR     PC,DO           ;GO CYCLE THRU INSTRS
 014112  017737  166150  001126                 MOV     @CONS,$BDDAT    ;READ REG 22
 014120  042737  077777  001126                 BIC     #77777,$BDDAT   ;SAVE ONLY INTENSITY ENA BIT
 014126  023737  001124  001126                 CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
 014134  001401                                 BEQ     2$              ;BR IF SO
 014136  104045                                 ERROR   45              ;INTENSITY ENA FAILED TO RESTORE
 014140  162737  100000  001124         2$:     SUB     #100000,$GDDAT  ;CLR INTENSITY ENA BIT AT EXPECTED LOC
 014146  100407                                 BMI     TST53           ;;ADVANCE TO NEXT STATE IF BOTH STATES TESTED
 014150  042737  000100  032342                 BIC     #100,BUFFER     ;SET UP FOR RESET STATE
 014156  052737  000100  032346                 BIS     #100,BUFFER+4   ;SET INTENSITY ENA AT INSTR LOC
 014164  000750                                 BR      1$              ;GO TRY RESET STATE

                                        ;;*****************************************************************
                                        ;*TEST 53        TEST THAT THE INTENSITY ENA GETS RESTORED ON CONSOLE 1
                                        ;;*****************************************************************
 014166  000004                         TST53:  SCOPE
 014170  012737  000053  001206                 MOV     #53,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 014176  012737  014234  001110                 MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
 014204  013737  002266  001122                 MOV     CONS,$BDADR     ;SET UP REG 22 ADRS
 014212  012737  001000  001124                 MOV     #1000,$GDDAT    ;EXPECT INTENSITY ENABLED TO BE SET INITIALLY
 014220  004437  024442                         JSR     R4,POPSET       ;SET UP JSR, POP & NOP INSTRS
 014224  164600                                 164600                  ;NOP INSTR
 014226  012737  164700  032342                 MOV     #164700,BUFFER  ;SET UP NOP INSTR
 014234  004737  024472                 1$:     JSR     PC,DO           ;GO CYCLE THRU INSTRS
 014240  017737  166022  001126                 MOV     @CONS,$BDDAT    ;READ REG 22
 014246  042737  176777  001126                 BIC     #176777,$BDDAT  ;SAVE ONLY INTENSITY ENA BIT
 014254  023737  001124  001126                 CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
 014262  001401                                 BEQ     2$              ;BR IS SO
 014264  104045                                 ERROR   45              ;INTENSITY ENA FAILED TO RESTORE
 014266  162737  001000  001124         2$:     SUB     #1000,$GDDAT    ;CLR INTENSITY ENA BIT AT EXPECTED LOC
 014274  100407                                 BMI     TST54           ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
 014276  042737  000100  032342                 BIC     #100,BUFFER     ;SET UP FOR RESET STATE
 014304  052737  000100  032346                 BIS     #100,BUFFER+4   ;SET INTENSITY ENA AT INSTR LOC
 014312  000750                                 BR      1$              ;GO TRY RESET STATE

                                        ;;*****************************************************************
                                        ;*TEST 54        TEST THAT L.P. INTR ENA GETS RESTORED ON CONSOLE 0
                                        ;;*****************************************************************
 014314  000004                         TST54:  SCOPE
 014316  012737  000054  001206                 MOV     #54,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 014324  012737  014362  001110                 MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
 014332  013737  002266  001122                 MOV     CONS,$BDADR     ;SET UP REG 22 ADRS
 014340  012737  004000  001124                 MOV     #4000,$GDDAT    ;EXPECT L.P. INTR ENA TO BE SET INITIALLY
 014346  012737  164060  032342                 MOV     #164060,BUFFER  ;SET UP NOP INSTR
```

```
014354  004437  024442                  JSR     R4,POPSET           ;SET UP JSR, POP & NOP INSTRS
014360  164040                          164040                      ;NOP INSTR
014362  004737  024472          1$:     JSR     PC,DO               ;GO CYCLE THRU INSTRS
014366  017737  165674  001126          MOV     @CONS,$BDDAT        ;READ REG 22
014374  042737  173777  001126          BIC     #173777,$BDDAT      ;SAVE ONLY L.P. INTR ENA
014402  023737  001124  001126          CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
014410  001401                          BEQ     2$                  ;BR IF SO
014412  104046                          ERROR   46                  ;L.P. INTR ENA FAILED TO RESTORE
014414  162737  004000  001124  2$:     SUB     #4000,$GDDAT        ;SET UP FOR RESET STATE
014422  100407                          BMI     TST55               ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
014424  042737  000020  032342          BIC     #20,BUFFER          ;SET UP FOR RESET STATE
014432  052737  000020  032346          BIS     #20,BUFFER+4        ;SET IT BEFORE POP
014440  000750                          BR      1$                  ;GO TRY RESET STATE

                                ;;*************************************************************
                                ;*TEST 55       TEST THAT L.P. INTR ENA GETS RESTORED ON CONSOLE 1
                                ;;*************************************************************
014442  000004                  TST55:  SCOPE
014444  012737  000055  001206          MOV     #55,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
014452  012737  014510  001110          MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
014460  013737  002266  001122          MOV     CONS,$BDADR         ;SET UP REG 22 ADRS
014466  012737  000040  001124          MOV     #40,$GDDAT          ;EXPECT L.P. INTR ENABLED TO BE SET INITIALLY
014474  012737  164460  032342          MOV     #164460,BUFFER      ;SET UP NOP INSTR
014502  004437  024442                  JSR     R4,POPSET           ;SET UP JSR, POP & NOP INSTRS
014506  164440                          164440                      ;NOP INSTR
014510  004737  024472          1$:     JSR     PC,DO               ;GO CYCLE THRU INSTRS
014514  017737  165546  001126          MOV     @CONS,$BDDAT        ;READ REG 22
014522  042737  177737  001126          BIC     #177737,$BDDAT      ;SAVE ONLY L.P. INTR ENA
014530  023737  001124  001126          CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
014536  001401                          BEQ     2$                  ;BR IF SO
014540  104046                          ERROR   46                  ;L.P. INTR ENA FAILED TO RESTORE
014542  162737  000040  001124  2$:     SUB     #40,$GDDAT          ;SET UP FOR RESET STATE
014550  100407                          BMI     TST56               ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
014552  042737  000020  032342          BIC     #20,BUFFER          ;SET UP FOR RESET STATE
014560  052737  000020  032346          BIS     #20,BUFFER+4        ;SET IF BEFORE POP
014566  000750                          BR      1$                  ;GO TRY RESET STATE

                                ;;*************************************************************
                                ;*TEST 56       TEST THAT L.P. SW INTR ENA GETS RESTORED ON CONSOLE 0
                                ;;*************************************************************
014570  000004                  TST56:  SCOPE
014572  012737  000056  001206          MOV     #56,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
014600  012737  014636  001110          MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
014606  013737  002266  001122          MOV     CONS,$BDADR         ;SET UP REG 22 ADRS
014614  012737  002000  001124          MOV     #2000,$GDDAT        ;EXPECT L.P. SW INTR ENA TO BE SET INITIALLY
014622  012737  164014  032342          MOV     #164014,BUFFER      ;SET UP NOP INSTR
014630  004437  024442                  JSR     R4,POPSET           ;SET UP JSR, POP & NOP INSTRS
014634  164010                          164010                      ;NOP INSTR
014636  004737  024472          1$:     JSR     PC,DO               ;GO CYCLE THRU INSTRUCTIONS
014642  017737  165420  001126          MOV     @CONS,$BDDAT        ;READ REG 22
014650  042737  175777  001126          BIC     #175777,$BDDAT      ;SAVE ONLY L.P. SW INTR ENA
014656  023737  001124  001126          CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
014664  001401                          BEQ     2$                  ;BR IF NOT
014666  104047                          ERROR   47                  ;L.P. SW INTR ENA FAILED TO RESTORE
014670  162737  002000  001124  2$:     SUB     #2000,$GDDAT        ;SET UP FOR RESET STATE
014676  100407                          BMI     TST57               ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
```

```
014700  042737  000004  032342          BIC     #4,BUFFER       ;SET UP FOR RESET STATE
014706  052737  000004  032346          BIS     #4,BUFFER+4     ;SET IT BEFORE POP
014714  000750                          BR      1$              ;GO TRY RESET STATE

                                ;;*******************************************************************
                                ;*TEST 57       TEST THAT L.P. SW INTR ENA GETS RESTORED ON CONSOLE 1
                                ;;*******************************************************************
014716  000004                  TST57:  SCOPE
014720  012737  000057  001206          MOV     #57,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
014726  012737  014764  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
014734  013737  002266  001122          MOV     CONS,$BDADR     ;SET UP REG 22 ADRS
014742  012737  000020  001124          MOV     #20,$GDDAT      ;EXPECT L.P. SW INTR ENA TO BE SET INITIALLY
014750  012737  164414  032342          MOV     #164414,BUFFER  ;SET UP NOP INSTR
014756  004437  024442                  JSR     R4,POPSET       ;SET UP JSR, POP & NOP INSTRS
014762  164410                          164410                  ;NOP INSTR
014764  004737  024472          1$:     JSR     PC,DO           ;GO CYCLE THRU INSTRS
014770  017737  165272  001126          MOV     @CONS,$BDDAT    ;READ REG 22
014776  042737  177757  001126          BIC     #177757,$BDDAT  ;SAVE ONLY L.P. SW INTR ENA
015004  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
015012  001401                          BEQ     2$              ;BR IF NOT
015014  104047                          ERROR   47              ;L.P. SW INTR ENA FAILED TO RESTORE
015016  162737  000020  001124  2$:     SUB     #20,$GDDAT      ;SET UP FOR RESET STATE
015024  100407                          BMI     TST60           ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
015026  042737  000004  032342          BIC     #4,BUFFER       ;SET UP RESET STATE
015034  052737  000004  032346          BIS     #4,BUFFER+4     ;SET IT BEFORE POPS
015042  000750                          BR      1$              ;GO TRY RESET STATE

                                ;;*******************************************************************
                                ;*TEST 60       TEST THAT THE LINE TYPE WILL GET RESTORED
                                ;;*******************************************************************
015044  000004                  TST60:  SCOPE
015046  012737  000060  001206          MOV     #60,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
015054  012737  015104  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
015062  013737  002246  001122          MOV     SREG0,$BDADR    ;SET UP REG 02 ADRS
015070  012737  000003  001124          MOV     #3,$GDDAT       ;EXPECT MAX VALUE INITIALLY
015076  004437  024442                  JSR     R4,POPSET       ;SET UP JSR, POP & CHAR INSTRS
015102  100004                          100004                  ;CHAR INSTR
015104  012737  100004  032342  1$:     MOV     #100004,BUFFER  ;SET UP CHAR INSTR
015112  053737  001124  032342          BIS     $GDDAT,BUFFER   ;SET UP LINE TYPE
015120  004737  024472                  JSR     PC,DO           ;GO CYCLE THRU INSTRS
015124  017737  165116  001126          MOV     @SREG0,$BDDAT   ;READ LINE TYPE
015132  042737  077774  001126          BIC     #77774,$BDDAT   ;SAVE ONLY LINE TYPE
015140  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
015146  001401                          BEQ     2$              ;BR IF SO
015150  104050                          ERROR   50              ;LINE TYPE FAILED TO RESTORE
015152  005337  001124          2$:     DEC     $GDDAT          ;ADVANCE TO NEXT VALUE
015156  100405                          BMI     TST61           ;;ADVANCE TO NEXT TEST IF ALL VALUES TESTED
015160  001351                          BNE     1$              ;TRY NEXT LINE TYPE IF NOT ZERO
015162  052737  000003  032346          BIS     #3,BUFFER+4     ;SET LINE TYPE TO 3 WHEN RESTORING ZEROS
015170  000745                          BR      1$              ;GO RESTORE ALL ZEROS

                                ;;*******************************************************************
                                ;*TEST 61       TEST THAT THE INTENSITY LEVEL WILL GET RESTORED
                                ;;*******************************************************************
015172  000004                  TST61:  SCOPE
015174  012737  000061  001206          MOV     #61,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
```

```
015202 012737 015236 001110            MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
015210 013737 002246 001122            MOV     SREGO,$BDADR        ;SET UP REG 02 ADRS
015216 012737 003400 001124            MOV     #3400,$GDDAT        ;EXPECT MAX VALUE INITIALLY
015224 012700 001600                   MOV     #1600,R0            ;SET UP INTENSITY LEVEL FOR INSTR USE
015230 004437 024442                   JSR     R4,POPSET           ;SET UP JSR, POP & CHAR INSTRS
015234 102000                          102000                      ;CHAR INSTR
015236 012737 102000 032342    1$:     MOV     #102000,BUFFER      ;SET UP CHAR INSTR
015244 050037 032342                   BIS     R0,BUFFER           ;SET UP INTENSITY LEVEL
015250 004737 024472                   JSR     PC,DO               ;GO CYCLE THRU INSTR
015254 017737 164766 001126            MOV     @SREGO,$BDDAT       ;READ INTENSITY LEVEL
015262 042737 174377 001126            BIC     #174377,$BDDAT      ;SAVE ONLY INTENSITY LEVEL
015270 023737 001124 001126            CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
015276 001401                          BEQ     2$                  ;BR IF SO
015300 104051                          ERROR   51                  ;INTENSITY LEVEL FAILED TO RESTORE
015302 162700 000200           2$:     SUB     #200,R0             ;SET UP R0 WITH NEXT VALUE
015306 162737 000400 001124            SUB     #400,$GDDAT         ;ADVANCE TO NEXT VALUE
015314 100405                          BMI     TST62               ;;ADVANCE TO NEXT TEST IF ALL VALUES TESTED
015316 001347                          BNE     1$                  ;BR IF NEXT INTENSITY LEVEL NOT ZERO
015320 052737 001600 032346            BIS     #1600,BUFFER+4      ;SET INTENSITY LEVEL TO ALL ONES WHEN RESTORING ZEROS
015326 000743                          BR      1$                  ;GO RESTORE ALL ZEROS

                               ;;*****************************************************************
                               ;*TEST 62         TEST THAT BLINK WILL GET RESTORED
                               ;;*****************************************************************
015330 000004                  TST62:  SCOPE
015332 012737 000062 001206            MOV     #62,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
015340 012737 015376 001110            MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
015346 013737 002246 001122            MOV     SREGO,$BDADR        ;SET UP REG 02 ADRS
015354 012737 000010 001124            MOV     #10,$GDDAT          ;EXPECT BLINK INITIALLY
015362 012737 100030 032342            MOV     #100030,BUFFER      ;SET UP CHAR INSTR
015370 004437 024442                   JSR     R4,POPSET           ;SET UP JSR, POP & CHAR INSTRS
015374 100020                          100020                      ;CHAR INSTR
015376 004737 024472           1$:     JSR     PC,DO               ;GO CYCLE THRU INSTRS
015402 017737 164640 001126            MOV     @SREGO,$BDDAT       ;READ REG 02
015410 042737 177767 001126            BIC     #177767,$BDDAT      ;SAVE ONLY THE BLINK BIT
015416 023737 001124 001126            CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
015424 001401                          BEQ     2$                  ;BR IF SO
015426 104052                          ERROR   52                  ;BLINK FAILED TO RESTORE
015430 162737 000010 032342    2$:     SUB     #10,BUFFER          ;SET UP RESET STATE
015436 162737 000010 001124            SUB     #10,$GDDAT          ;GO TO RESET STATE
015444 100404                          BMI     TST63               ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
015446 052737 000010 032346            BIS     #10,BUFFER+4        ;SET BLINK WHEN RESTORING ZERO
015454 000750                          BR      1$                  ;GO TRY RESET STATE

                               ;;*****************************************************************
                               ;*TEST 63         TEST THAT THE MODE WILL GET RESTORED
                               ;;*****************************************************************
015456 000004                  TST63:  SCOPE
015460 012737 000100 001166            MOV     #100,$TIMES         ;;DO 100 ITERATIONS
015466 012737 000063 001206            MOV     #63,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
015474 012737 015524 001110            MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
015502 013737 002246 001122            MOV     SREGO,$BDADR        ;SET UP REG 02 ADRS
015510 012737 044000 001124            MOV     #44000,$GDDAT       ;EXPECT ABS VEC MODE INITIALLY
015516 004437 024442                   JSR     R4,POPSET           ;SET UP JSR, POP & MODE INSTR
015522 100000                          100000                      ;CHAR INSTR
015524 052737 100000 032346    1$:     BIS     #100000,BUFFER+4          ;RESTORE 'INSTR' BIT AT LOC BUFFER+4
```

```
015532  012737  100000  032342          MOV     #100000,BUFFER     ;SET UP CHAR INSTR
015540  053737  001124  032342          BIS     $GDDAT,BUFFER      ;SET UP CURRENT DISPLAY INSTR
015546  004737  024472                  JSR     PC,DO              ;GO CYCLE THRU INSTRS
015552  042737  100000  032346          BIC     #100000,BUFFER+4        ;USE LOC BUFFER+4 NOW AS DISPLAY DATA
015560  005277  164460                  INC     @DPC               ;PICK UP DATA FOR DISPLAY INSTR ISSUED
015564  000240                          NOP                        ; BEFORE JSR - THIS WILL LD MODE BITS
015566  017737  164454  001126          MOV     @SREG0,$BDDAT      ;READ REG 02
015574  042737  103777  001126          BIC     #103777,$BDDAT     ;SAVE ONLY MODE BITS
015602  023737  001124  001126          CMP     $GDDAT,$BDDAT      ;IS IT CORRECT?
015610  001401                          BEQ     2$                 ;BR IF OK
015612  104053                          ERROR   53                 ;MODE FAILED TO RESTORE
015614  162737  004000  001124  2$:     SUB     #4000,$GDDAT       ;GO TO NEXT DISPLAY MODE
015622  100340                          BPL     1$                 ;BR IF MORE MODES TO TEST

                                ;;*****************************************************************
                                ;*TEST 64        TEST THAT STOP INTR ENABLED GETS RESTORED-CK IT ON STK
                                ;;*****************************************************************
015624  000004                  TST64:  SCOPE
015626  012737  000064  001206          MOV     #64,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
015634  012737  015672  001110          MOV     #1$,$LPERR         ;SET UP SCOPE LOOP ADRS
015642  013737  002272  001122          MOV     STKVAL,$BDADR      ;SET UP REG 26 ADRS
015650  012737  000001  001124          MOV     #1,$GDDAT          ;EXPECT IT TO BE SET INITIALLY
015656  012737  171400  032342          MOV     #171400,BUFFER     ;SET UP LOAD STATUS A INSTR
015664  004437  024442                  JSR     R4,POPSET          ;SET UP JSR, POP & LOAD STATUS A INSTR
015670  171000                          171000                     ;LOAD STATUS A INSTR
015672  004737  024472          1$:     JSR     PC,DO              ;GO CYCLE THRU INSTRS
015676  012777  032344  164340          MOV     #BUFFER+2,@DPC     ;DO A JSR
015704  004537  024426                  JSR     R5,DELAY           ;STALL FOR STACKING
015710  000001                          BIT0                       ;COUNT TO 1
015712  012777  020037  164356          MOV     #20037,@STKPT      ;SET UP STK SEL
015720  017737  164346  001126          MOV     @STKVAL,$BDDAT     ;READ STOP INTR ENA BYTE FROM STACK
015726  042737  177776  001126          BIC     #177776,$BDDAT     ;SAVE ONLY STOP INTR ENABLED BIT
015734  023737  001124  001126          CMP     $GDDAT,$BDDAT      ;WAS IT RESTORED ON THE POP INSTR?
015742  001401                          BEQ     2$                 ;BR IF SO
015744  104054                          ERROR   54                 ;STOP INTR ENABLE FAILED TO RESTORE
015746  005337  001124          2$:     DEC     $GDDAT             ;SET UP FOR RESET STATE
015752  100407                          BMI     TST65              ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
015754  042737  000400  032342          BIC     #400,BUFFER        ;SET UP FOR STKING RESET STATE
015762  052737  000400  032346          BIS     #400,BUFFER+4      ;SET BIT BEFORE POP INSTR
015770  000740                          BR      1$                 ;GO TRY RESET STATE

                                ;;*****************************************************************
                                ;*TEST 65        TEST THAT DEPTH QUEING BIT GETS RESTORED-CK IT ON STK
                                ;;*****************************************************************
015772  000004                  TST65:  SCOPE
015774  012737  000065  001206          MOV     #65,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
016002  012737  016040  001110          MOV     #1$,$LPERR         ;SET UP SCOPE LOOP ADRS
016010  013737  002272  001122          MOV     STKVAL,$BDADR      ;SET UP REG 26 ADRS
016016  012737  002000  001124          MOV     #2000,$GDDAT       ;EXPECT IT TO BE SET INITIALLY
016024  012737  176014  032342          MOV     #176014,BUFFER     ;SET UP LOAD STATUS B' INSTR
016032  004437  024442                  JSR     R4,POPSET          ;SET UP JSR, POP & LOAD STATUS B' INSTRS
016036  176010                          176010                     ;LOAD STATUS B' INSTR
016040  004737  024472          1$:     JSR     PC,DO              ;GO CYCLE THRU INSTRS
016044  012777  032344  164172          MOV     #BUFFER+2,@DPC     ;DO A JSR
016052  004537  024426                  JSR     R5,DELAY           ;STALL FOR STACKING
016056  000001                          BIT0                       ;COUNT TO 1
```

```
016060  012777  020037  164210          MOV     #20037,@STKPT           ;SET UP STK SEL
016066  017737  164200  001126          MOV     @STKVAL,$BDDAT          ;READ DEPTH QUEING BYTE FROM STK
016074  042737  175777  001126          BIC     #175777,$BDDAT          ;SAVE ONLY DEPTH QUEING BIT
016102  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;WAS IT RESTORED ON THE POP INSTR?
016110  001401                          BEQ     2$                      ;BR IF SO
016112  104055                          ERROR   55                      ;DEPTH QUE FAILED TO RESTORE
016114  162737  002000  001124  2$:     SUB     #2000,$GDDAT            ;SET UP FOR RESET STATE
016122  100407                          BMI     TST66                   ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
016124  042737  000004  032342          BIC     #4,BUFFER               ;SET UP FOR STKING RESET STATE
016132  052737  000004  032346          BIS     #4,BUFFER+4             ;SET BIT BEFORE POP INSTR
016140  000737                          BR      1$                      ;GO TRY RESET STATE

                                ;;*************************************************************
                                ;*TEST 66        TEST THAT DEPTH QUE CONTROL BIT GETS RESTROED-CK IT ON STK
                                ;;*************************************************************
016142  000004                  TST66:  SCOPE
016144  012737  000066  001206          MOV     #66,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
016152  012737  016210  001110          MOV     #1$,$LPERR              ;SET UP SCOPE LOOP ADRS
016160  013737  002272  001122          MOV     STKVAL,$BDADR           ;SET UP REG 26 ADRS
016166  012737  000400  001124          MOV     #400,$GDDAT             ;EXPECT IT TO BE SET INITIALLY
016174  012737  176300  032342          MOV     #176300,BUFFER          ;SET UP LOAD STATUS B' INSTR
016202  004437  024442                  JSR     R4,POPSET               ;SET UP JSR, POP & LOAD STATUS B' INSTRS
016206  176200                          176200                          ;LOAD STATUS B' INSTR
016210  004737  024472          1$:     JSR     PC,DO                   ;GO CYCLE THRU INSTRS
016214  012777  032342  164022          MOV     #BUFFER,@DPC            ;DO A JSR REL
016222  004537  024426                  JSR     R5,DELAY                ;STALL FOR STACKING
016226  000001                          BIT0                            ;COUNT TO 1
016230  012777  020037  164040          MOV     #20037,@STKPT           ;SET UP STACK SEL
016236  017737  164030  001126          MOV     @STKVAL,$BDDAT          ;READ STACK
016244  042737  177377  001126          BIC     #177377,$BDDAT          ;SAVE ONLY DEPTH QUE CONTROL BIT
016252  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;WAS IT RESTORED ON THE POP INSTR?
016260  001401                          BEQ     2$                      ;BR IF SO
016262  104055                          ERROR   55                      ;DEPTH QUE CONTROL FAILED TO RESTORE
016264  162737  000400  001124  2$:     SUB     #400,$GDDAT             ;SET UP FOR RESET SATATE
016272  100407                          BMI     TST67                   ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
016274  042737  000100  032342          BIC     #100,BUFFER             ;SET UP FOR STACKING RESET STATE
016302  052737  000100  032346          BIS     #100,BUFFER+4           ;SET BIT BEFORE POP INSTR
016310  000737                          BR      1$                      ;GO TRY RESET STATE

                                ;;*************************************************************
                                ;*TEST 67        TEST THAT EDGE INTR ENABLED GETS RESTORED-CK IT ON STK
                                ;;*************************************************************
016312  000004                  TST67:  SCOPE
016314  012737  000067  001206          MOV     #67,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
016322  012737  016360  001110          MOV     #1$,$LPERR              ;SET UP SCOPE LOOP ADRS
016330  013737  002272  001122          MOV     STKVAL,$BDADR           ;SET UP REG 26 ADRS
016336  012737  004000  001124          MOV     #4000,$GDDAT            ;EXPECT IT TO BE SET INITIALLY
016344  012737  176060  032342          MOV     #176060,BUFFER          ;SET UP LOAD STATUS B' INSTR
016352  004437  024442                  JSR     R4,POPSET               ;SET UP JSR, POP & LOAD STATUS B' INSTR
016356  176040                          176040                          ;LOAD STATUS B' INSTR
016360  004737  024472          1$:     JSR     PC,DO                   ;GO CYCLE THRU INSTRS
016364  012777  032344  163652          MOV     #BUFFER+2,@DPC          ;DO A JSR
016372  004537  024426                  JSR     R5,DELAY                ;STALL FOR STACKING
016376  000001                          BIT0                            ;COUNT TO 1
016400  012777  020037  163670          MOV     #20037,@STKPT           ;SET UP STK SEL
016406  017737  163660  001126          MOV     @STKVAL,$BDDAT          ;READ EDGE INTR ENA BYTE FROM STK
```

```
016414  042737  173777  001126          BIC     #173777,$BDDAT   ;SAVE ONLY EDGD INTR ENA BIT
016422  023737  001124  001126          CMP     $GDDAT,$BDDAT    ;WAS IT RESTORED ON THE POP INSTR?
016430  001401                          BEQ     2$               ;BR IF SO
016432  104056                          ERROR   56               ;EDGE INTR ENABLE FAILED TO RESTORE
016434  162737  004000  001124  2$:     SUB     #4000,$GDDAT     ;SET UP FOR RESET STATE
016442  100407                          BMI     TST70            ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
016444  042737  000020  032342          BIC     #20,BUFFER       ;SET UP FOR STACKING RESET STATE
016452  052737  000020  032346          BIS     #20,BUFFER+4     ;SET BIT BEFORE POP INSTR
016460  000737                          BR      1$               ;GO TRY RESET STATE

                                ;;*****************************************************************
                                ;*TEST 70       TEST THAT CHAR STRING ESCAPE GETS RESTORED-CK IT ON STK
                                ;;*****************************************************************
016462  000004                  TST70:  SCOPE
016464  012737  000070  001206          MOV     #70,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
016472  012737  016530  001110          MOV     #1$,$LPERR       ;SET UP SCOPE LOOP ADRS
016500  013737  002272  001122          MOV     STKVAL,$BDADR    ;SET UP REG 26 ADRS
016506  012737  010000  001124          MOV     #10000,$GDDAT    ;EXPECT IT TO BE SET INITIALLY
016514  012737  176003  032342          MOV     #176003,BUFFER   ;SET UP LOAD STATUS B' INSTR
016522  004437  024442                  JSR     R4,POPSET        ;SET UP JSR, POP & LOAD STATUS B' INSTR
016526  176002                          176002                   ;LOAD STATUS B' INSTR
016530  004737  024472          1$:     JSR     PC,DO            ;GO CYCLE THRU INSTRS
016534  012777  032344  163502          MOV     #BUFFER+2,@DPC   ;DO A JSR
016542  004537  024426                  JSR     R5,DELAY         ;STALL FOR STACKING
016546  000001                          BIT0                     ;COUNT TO 1
016550  012777  020037  163520          MOV     #20037,@STKPT    ;SET UP STK SEL
016556  017737  163510  001126          MOV     @STKVAL,$BDDAT   ;READ CHAR STRING ESCAPE BYTE FROM STK
016564  042737  167777  001126          BIC     #167777,$BDDAT   ;SAVE ONLY CHAR STRING ESCAPE BIT
016572  023737  001124  001126          CMP     $GDDAT,$BDDAT    ;WAS IT RESTORED ON THE POP INSTR?
016600  001401                          BEQ     2$               ;BR IF SO
016602  104057                          ERROR   57               ;CHARACTER STRING ESCAPE FAILED TO RESTORE
016604  162737  010000  001124  2$:     SUB     #10000,$GDDAT    ;SET UP FOR RESET STATE
016612  100407                          BMI     TST71            ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
016614  042737  000001  032342          BIC     #1,BUFFER        ;SET UP FOR STACKING RESET STATE
016622  052737  000001  032346          BIS     #1,BUFFER+4      ;SET BIT BEFORE POP INSTR
016630  000737                          BR      1$               ;GO TRY RESET STATE

                                ;;*****************************************************************
                                ;*TEST 71       TEST THAT L.P. HIT DISABLE GETS RESTORED-CK IT ON STK
                                ;;*****************************************************************
016632  000004                  TST71:  SCOPE
016634  012737  000071  001206          MOV     #71,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
016642  012737  016700  001110          MOV     #1$,$LPERR       ;SET UP SCOPE LOOP ADRS
016650  013737  002272  001122          MOV     STKVAL,$BDADR    ;SET UP REG 26 ADRS
016656  012737  000002  001124          MOV     #2,$GDDAT        ;EXPECT IT TO BE SET INITIALLY
016664  012737  170300  032342          MOV     #170300,BUFFER   ;SET UP LOAD STATUS A INSTR
016672  004437  024442                  JSR     R4,POPSET        ;SET UP JSR, POP & LOAD STATUS A INSTR
016676  170200                          170200                   ;LOAD STATUS A INSTR
016700  004737  024472          1$:     JSR     PC,DO            ;GO CYCLE THRU INSTRS
016704  012777  032344  163332          MOV     #BUFFER+2,@DPC   ;DO A JSR
016712  004537  024426                  JSR     R5,DELAY         ;STALL FOR STACKING
016716  000001                          BIT0                     ;COUNT TO 1
016720  012777  020037  163350          MOV     #20037,@STKPT    ;SET UP STK SEL
016726  017737  163340  001126          MOV     @STKVAL,$BDDAT   ;READ L.P. HIS DISABLE BYTE FROM STK
016734  042737  177775  001126          BIC     #177775,$BDDAT   ;SAVE ONLY L.P. HIS DISABLE BIT
016742  023737  001124  001126          CMP     $GDDAT,$BDDAT    ;WAS IT RESTORED ON THE POP INSTR?
```

```
016750  001401                            BEQ     2$                  ;BR IF SO
016752  104060                            ERROR   60                  ;L.P. HIT DISABLE FAILED TO RESTORE
016754  162737  000002  001124    2$:     SUB     #2,$GDDAT           ;SET UP FOR RESET STATE
016762  100407                            BMI     TST72               ;;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
016764  042737  000100  032342            BIC     #100,BUFFER         ;SET UP FOR STKING RESET STATE
016772  052737  000100  032346            BIS     #100,BUFFER+4       ;SET BIT BEFORE POP INSTR
017000  000737                            BR      1$                  ;GO TRY RESET STATE

                                  ;;*****************************************************************
                                  ;*TEST 72        TEST THAT SHIFT OUT WILL GET RESTORED - CK IT ON STK
                                  ;;*****************************************************************
017002  000004                    TST72:  SCOPE
017004  012737  000072  001206            MOV     #72,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
017012  012737  000340  177776            MOV     #340,PSW            ;SET PRIORITY TO HIGEST LEVEL
017020  012737  017102  001110            MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
017026  013737  002272  001122            MOV     STKVAL,$BDADR       ;SET UP REG 26 ADRS
017034  012737  040000  001124            MOV     #40000,$GDDAT       ;EXPECT SHIFT OUT INITIALLY
017042  012704  032342                    MOV     #BUFFER,R4          ;SET UP START ADRS IN R4
017046  012724  100000                    MOV     #100000,(R4)+       ;SET UP CHAR INSTR
017052  012724  007000                    MOV     #7000,(R4)+         ;SET UP SHIFT OUT CHAR
017056  012724  163001                    MOV     #163001,(R4)+       ;SET UP JSR REL + 4
017062  012724  163000                    MOV     #163000,(R4)+       ;SET UP JSR REL - THIS WILL FORCE SHIFT OUT BACK ON STK
017066  012724  100000                    MOV     #100000,(R4)+       ;SET UP CHAR INSTR TO COM SHIFT OUT
017072  012724  000017                    MOV     #17,(R4)+           ;SET UP CHAR LOC - SHIFT IN
017076  012714  166000                    MOV     #166000,(R4)        ;SET UP POP & RESTORE INSTR
017102  012777  020040  163166    1$:     MOV     #20040,@STKPT       ;SET TOP OF STK
017110  012777  032342  163126            MOV     #BUFFER,@DPC        ;START
017116  012704  000006                    MOV     #6,R4               ;SET UP NPR CNTR
017122  005277  163116            2$:     INC     @DPC                ;ADVANCE TO NEXT INSTR OR DATA
017126  004537  024426                    JSR     R5,DELAY            ;ALLOW TIME FOR EACH OPERATION TO COMPLETE
017132  000004                            BIT2                        ;COUNT TO 4
017134  005304                            DEC     R4                  ;COUNT THIS NPR
017136  001371                            BNE     2$                  ;BR IF MORE NPRS
017140  012777  020037  163130            MOV     #20037,@STKPT       ;SET UP STK SEL
017146  017737  163120  001126            MOV     @STKVAL,$BDDAT      ;READ REG 26
017154  042737  137777  001126            BIC     #137777,$BDDAT      ;SAVE ONLY SHIFT OUT
017162  023737  001124  001126            CMP     $GDDAT,$BDDAT       ;IS IT CORRECT?
017170  001401                            BEQ     3$                  ;BR IF SO
017172  104061                            ERROR   61                  ;SHIFT OUT FAILED TO RESTORE
017174  012737  000017  032344    3$:     MOV     #17,BUFFER+2        ;SET UP FOR NO SHIFT OUT
017202  012737  000016  032354            MOV     #16,BUFFER+12       ;COMPLEMENT IT
017210  162737  040000  001124            SUB     #40000,$GDDAT       ;SET UP FOR RESET STATE
017216  100331                            BPL     1$                  ;BR IF RESET STATE NOT TESTED

                                  ;;*****************************************************************
                                  ;*TEST 73        TEST THAT CHAR STRING ESCAPE ON TERMINATE CHAR WILL POP THE STACK
                                  ;;*****************************************************************
017220  000004                    TST73:  SCOPE
017222  012737  000010  001166            MOV     #10,$TIMES          ;;DO 10 ITERATIONS
017230  012737  000073  001206            MOV     #73,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
017236  012737  032346  001124            MOV     #BUFFER+4,$GDDAT        ;EXPECT ADRS BUFFER+4 AFTER TERM CHAR
017244  012737  176003  032342            MOV     #176003,BUFFER  ;SET UP CHAR STRING ESCAPE INSTR
017252  012737  163002  032344            MOV     #163002,BUFFER+2        ;SET UP JSR REL INSTR
017260  012737  100000  032352            MOV     #100000,BUFFER+10       ;SET CHAR INSTR
017266  012700  000377                    MOV     #377,R0             ;START WITH MAX TERM CHAR VALUE
017272  012777  020040  162776    1$:     MOV     #20040,@STKPT       ;SET TOP OF STACK
```

```
 017300 010077 162770               MOV     R0,@TERMCH      ;SET UP TERM CHAR
 017304 010037 032354               MOV     R0,BUFFER+12            ;SET UP VALUE FOR INSTR
 017310 012777 032342 162726        MOV     #BUFFER,@DPC    ;START
 017316 012737 017272 001110        MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
 017324 005277 162714               INC     @DPC            ;ADVANCE TO JSR REL INSTR
 017330 004537 024426               JSR     R5,DELAY        ;STALL FOR STACKING
 017334 000001                      BIT0                    ;COUNT TO 1
 017336 005277 162702               INC     @DPC            ;RESUME - SHOULD PICK UP CHAR INSTR
 017342 013737 002244 001122        MOV     DPC,$BDADR      ;SET UP REG ADRS 00
 017350 005277 162670               INC     @DPC            ;RESUME - PICK UP TERM CHAR
 017354 004537 024426               JSR     R5,DELAY        ;STALL FOR POP
 017360 000020                      BIT4                    ;COUNT TO 20
 017362 017737 162656 001126        MOV     @DPC,$BDDAT     ;READ DPC
 017370 023737 001124 001126        CMP     $GDDAT,$BDDAT   ;DID IT DO THE POP?
 017376 001401                      BEQ     2$              ;BR IF SO
 017400 104062                      ERROR   62              ;TERMINATE CHARACTER FAILED TO POP STACK
 017402 105300              2$:     DECB    R0              ;ADVANCE TO NEXT TERMINATE VALUE
 017404 100732                      BMI     1$              ;BR IF MORE TO TEST

                            ;;**************************************************************
                            ;*TEST 74        TEST THAT CHAR STRING ESCAPE WHEN DISABLED WILL NOT POP THE STACK
                            ;;**************************************************************
 017406 000004              TST74:  SCOPE
 017410 012737 000074 001206        MOV     #74,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 017416 012737 032356 001124        MOV     #BUFFER+14,$GDDAT       ;EXPECT ADRS DPT3 AFTER TERM CHAR
 017424 012737 176002 032342        MOV     #176002,BUFFER  ;SET UP CHAR STRING ESCAPE INSTR-OFF
 017432 012737 100000 032352        MOV     #100000,BUFFER+10       ;SET UP CHAR INSTR
 017440 012737 000177 032354        MOV     #177,BUFFER+12  ;SET UP ASCII TERM CHAR
 017446 012777 000377 162620        MOV     #377,@TERMCH    ;SET UP TERM CHAR
 017454 012777 020040 162614        MOV     #20040,@STKPT   ;SET TOP OF STACK
 017462 012777 032342 162554        MOV     #BUFFER,@DPC    ;START
 017470 012737 163002 032344        MOV     #163002,BUFFER+2        ;SET UP JSR REL
 017476 005277 162542               INC     @DPC            ;ADVANCE TO JSR INSTR
 017502 004537 024426               JSR     R5,DELAY        ;STALL FOR STACKING
 017506 000001                      BIT0                    ;COUNT TO 1
 017510 005277 162530               INC     @DPC            ;ADVANCE TO CHAR INSTR
 017514 013737 002244 001122        MOV     DPC,$BDADR      ;SET UP REG ADRS 00
 017522 005277 162516               INC     @DPC            ;ADVANCE TO TERM CHAR
 017526 004537 024426               JSR     R5,DELAY        ;STALL FOR POTENTIAL POP
 017532 000020                      BIT4                    ;COUNT TO 20
 017534 017737 162504 001126        MOV     @DPC,$BDDAT     ;READ THE DPC
 017542 023737 001124 001126        CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
 017550 001401                      BEQ     TST75           ;;GO TO NEXT TEST IF OK
 017552 104062                      ERROR   62              ;TERM CHAR POPPED WHEN CHAR STRING TERM DISABLED

                            ;;**************************************************************
                            ;*TEST 75        TEST THAT STACK OVERFLOW WILL CAUSE AN INTERRUPT
                            ;;**************************************************************
 017554 000004              TST75:  SCOPE
 017556 012737 000040 001166        MOV     #40,$TIMES      ;;DO 40 ITERATIONS
 017564 012737 000075 001206        MOV     #75,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 017572 013701 001252               MOV     $VECT1,R1       ;GET BASIC VECTOR ADRS
 017576 012761 017720 000010        MOV     #2$,10(R1)      ;SET UP STK OVFL INT ADRS
 017604 012761 000340 000012        MOV     #340,12(R1)     ;SET UP PSW TO 7 ON INT
 017612 013737 002256 001122        MOV     $REG1,$BDADR    ;SET UP REG 12 ADRS
 017620 012777 020040 162450        MOV     #20040,@STKPT   ;SET TOP OF STK
```

```
017626 012700 000011                MOV     #11,R0              ;WILL COUNT TO OVFL IN R0
017632 012737 020000 001124         MOV     #20000,$GDDAT       ;WILL EXPECT STK OVFL
017640 012737 163000 032342         MOV     #163000,BUFFER      ;SET UP JSR INSTR
017646 012737 000340 177776 1$:     MOV     #340,PSW            ;SET PSW TO 7
017654 012777 032342 162362         MOV     #BUFFER,@DPC        ;START
017662 004537 024426                JSR     R5,DELAY            ;STALL FOR STKING
017666 000001                       BITO                        ;COUNT TO 1
017670 005037 177776                CLR     PSW                 ;ALLOW INT TO OCCUR
017674 005300                       DEC     R0                  ;COUNT THIS STK OPERATION
017676 001363                       BNE     1$                  ;BR IF NOT TO STK OVFL LEVEL
017700 017737 162352 001126         MOV     @SREG1,$BDDAT       ;READ REG 12
017706 042737 157777 001126         BIC     #157777,$BDDAT      ;SAVE ONLY STK OVFL
017714 104063                       ERROR   63                  ;STACK OVERFLOW FAILED TO INTERRUPT
017716 000424                       BR      4$                  ;GO LOOK FOR LOOP ON TEST SWITCH
017720 022626               2$:     CMP     (R6)+,(R6)+         ;FIX STK SINCE NO RTI
017722 032777 020000 162326         BIT     #20000,@SREG1       ;SEE IF STK OVFL BIT SET
017730 001004                       BNE     3$                  ;BR IF SO
017732 005037 001126                CLR     $BDDAT              ;INDICATE OVFLO BIT NOT SET
017736 104063                       ERROR   63                  ;STACK OVERFLOW INTERRUPTED BUT NO FLAG
017740 000413                       BR      4$                  ;LOOK FOR LOOP ON TEST SW
017742 005037 001124        3$:     CLR     $GDDAT              ;EXPECT ZERO
017746 000005                       RESET                       ;CLR FLAG
017750 032777 020000 162270         BIT     #20000,@SREG0       ;STILL SET?
017756 001404                       BEQ     4$                  ;BR IF NOT
017760 012737 020000 001126         MOV     #20000,$BDDAT       ;INDICATE IT IS STILL SET
017766 104063                       ERROR   63                  ;RESET FAILED TO CLR STACK OVERFLOW FLAG
017770 004737 024626        4$:     JSR     PC,RSTVEC           ;RESTORE STK OVERFLOW VECTOR HALT

                            ;;*****************************************************************
                            ;*TEST 76        TEST THAT STACK UNDERFLOW WILL CAUSE AN INTERRUPT
                            ;;*****************************************************************
017774 000004               TST76:  SCOPE
017776 012737 000040 001166         MOV     #40,$TIMES          ;;DO 40 ITERATIONS
020004 012737 000076 001206         MOV     #76,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
020012 013701 001252                MOV     $VECT1,R1           ;GET BASIC VECTOR ADRS
020016 012761 020140 000010         MOV     #2$,10(R1)          ;SET UP STK UNDERFLOW INT ADRS
020024 012761 000340 000012         MOV     #340,12(R1)         ;SET UP PSW TO 7 ON INT
020032 013737 002256 001122         MOV     SREG1,$BDADR        ;SET UP REG 12 ADRS
020040 012700 000011                MOV     #11,R0              ;WILL COUNT TO UNDERFLOW IN R0
020044 012777 020000 162224         MOV     #20000,@STKPT       ;SET STACK PTR TO ZERO
020052 012737 010000 001124         MOV     #10000,$GDDAT       ;WILL EXPECT STK UNDERFLOW
020060 012737 165000 032342         MOV     #165000,BUFFER      ;SET UP POP INSTR
020066 012737 000340 177776 1$:     MOV     #340,PSW            ;SET PSW TO 7
020074 012777 032342 162142         MOV     #BUFFER,@DPC        ;START
020102 004537 024426                JSR     R5,DELAY            ;STALL FOR POPING
020106 000001                       BITO                        ;COUNT TO 1
020110 005037 177776                CLR     PSW                 ;ALLOW INT TO OCCUR
020114 005300                       DEC     R0                  ;COUNT THIS POP OPERATION
020116 001363                       BNE     1$                  ;BR IF NOT ST STK UNDERFLO LEVEL
020120 017737 162132 001126         MOV     @SREG1,$BDDAT       ;READ REG 12
020126 043737 167777 001126         BIC     167777,$BDDAT       ;SAVE ONLY STK UNDERFLOW
020134 104064                       ERROR   64                  ;STACK UNDERFLOW FAILED TO INTERRUPT
020136 000424                       BR      4$                  ;GO LOOK FOR LOOP ON TEST SWITCH
020140 022626               2$:     CMP     (R6)+,(R6)+         ;FIX STACK SINCE NO RTI
020142 032777 010000 162106         BIT     #10000,@SREG1       ;SEE IF STK UNDERFLOW BIT SET
020150 001004                       BNE     3$                  ;BR IF SO
```

```
020152  005037  001126                   CLR     $BDDAT            ;INDICATE UNDERFLOW BIT NOT SET
020156  104064                           ERROR   64                ;STACK UNDERFLOW INTERRUPTED BUT NO FLAG
020160  000413                           BR      4$                ;GO LOOK FOR LOOP ON TEST SW
020162  005037  001124           3$:     CLR     $GDDAT            ;EXPECT ZERO
020166  000005                           RESET                     ;CLR FLAG
020170  032777  010000  162050           BIT     #10000,@SREG0     ;STILL SET?
020176  001404                           BEQ     4$                ;BR IF NOT
020200  012737  010000  001126           MOV     #10000,$BDDAT     ;INDICATE IT IS STILL SET
020206  104064                           ERROR   64                ;RESET FAILED TO CLR STACK UNDERFLOW FLAG
020210  004737  024626           4$:     JSR     PC,RSTVEC         ;RESTORE STK UNDERFLOW VECTOR WITH HALT

                                 ;;***********************************************************
                                 ;*TEST 77        TEST THAT X DELTA LENGTH COMPUTES WITH MAINT SW3 SET
                                 ;;***********************************************************
020214  000004                   TST77:  SCOPE
020216  012737  000040  001166           MOV     #40,$TIMES        ;;DO 40 ITERATIONS
020224  012737  000077  001206           MOV     #77,$TESTN        ;;SET TEST NUMBER IN APT MAIL BOX
020232  000005                           RESET                     ;MAKE SURE ALL FLAGS ARE CLEARED
020234  012737  020316  001110           MOV     #1$,$LPERR        ;SET UP SCOPE LOOP ADRS
020242  013737  002250  001122           MOV     XPOS,$BDADR       ;SET UP REG ADRS 04
020250  012700  032342                   MOV     #BUFFER,R0        ;GET INSTR ADRS POINTER
020254  012720  154024                   MOV     #154024,(R0)+     ;SET UP UNITY VECTOR SCALE
020260  012720  114000                   MOV     #114000,(R0)+     ;STORE POINT INSTR
020264  005020                           CLR     (R0)+             ;X=0
020266  005020                           CLR     (R0)+             ;Y=0
020270  012720  110000                   MOV     #110000,(R0)+     ;STORE LONG VECTOR INSTR
020274  005020                           CLR     (R0)+             ;X TO BE UPDATED
020276  005020                           CLR     (R0)+             ;Y=0
020300  012710  172000                   MOV     #172000,(R0)      ;STORE STOP INSTR
020304  012700  000001                   MOV     #1,R0             ;R0 CONTAINS X VECTOR LENGTH
020310  012737  000001  001124           MOV     #1,$GDDAT         ;CONTAINS EXPECTED DELTA LENGTH
020316  010037  032354           1$:     MOV     R0,BUFFER+12      ;SET UP NEXT VECTOR LENGTH
020322  004537  025020                   JSR     R5,EXECUTE        ;GO START DISPLAY
020326  000007                           7                         ;RESUME COUNT

020330  017737  161714  001126           MOV     @XPOS,$BDDAT      ;DELTA LENGTH READ BACK FROM X POSITION REG
020336  023737  001124  001126           CMP     $GDDAT,$BDDAT     ;IS IT CORRECT?
020344  001403                           BEQ     2$                ;BR IF SO
020346  010037  001164                   MOV     R0,$TMP0          ;SET UP X VECTOR LENGTH FOR ER TYPE
020352  104065                           ERROR   65                ;X DELTA LENGTH INCORRECT
020354  005237  001124           2$:     INC     $GDDAT            ;BUMP EXPECTED DELTA LENGTH
020360  005200                           INC     R0                ;ADVANCE TO NEXT X VECTOR VALUE
020362  032700  002000                   BIT     #2000,R0          ;HAVE WE DONE 10 BITS?
020366  001753                           BEQ     1$                ;BR IF NOT

                                 ;;***********************************************************
                                 ;*TEST 100       TEST THAT Y DELTA LENGTH COMPUTES WITH MAINT SW3 SET
                                 ;;***********************************************************
020370  000004                   TST100: SCOPE
020372  012737  000040  001166           MOV     #40,$TIMES        ;;DO 40 ITERATIONS
020400  012737  000100  001206           MOV     #100,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
020406  012737  020472  001110           MOV     #1$,$LPERR        ;SET UP SCOPE LOOP ADRS
020414  013737  002252  001122           MOV     YPOS,$BDADR       ;SET UP REG ADRS 06
020422  012700  032342                   MOV     #BUFFER,R0        ;GET INSTR ADRS POINTER
020426  012720  154024                   MOV     #154024,(R0)+     ;SET UP UNITY VECTOR SCALE
020432  012720  114000                   MOV     #114000,(R0)+     ;STORE POINT INSTR
```

```
020436  005020                           CLR    (R0)+              ;X=0
020440  012720  001777                   MOV    #1777,(R0)+        ;Y=1777
020444  012720  110000                   MOV    #110000,(R0)+      ;STORE LONG VECTOR INSTR
020450  005020                           CLR    (R0)+              ;X=0
020452  005020                           CLR    (R0)+              ;Y TO BE UPDATED
020454  012710  172000                   MOV    #172000,(R0)       ;STORE STOP INSTR
020460  012700  020001                   MOV    #20001,R0          ;R0 CONTAINS Y VECTOR LENGTH (DRAW TOP TO BOTTOM)
020464  012737  000001  001124           MOV    #1,$GDDAT          ;CONTAINS EXPECTED Y DELTA LENGTH
020472  010037  032356           1$:     MOV    R0,BUFFER+14       ;SET UP NEXT VECTOR LENGTH
020476  004537  025020                   JSR    R5,EXECUTE         ;GO START DISPLAY
020502  000007                           7                         ;RESUME COUNT

020504  017737  161542  001126           MOV    @YPOS,$BDDAT       ;DELTA LENGTH READ BACK FROM Y POSITION REG
020512  023737  001124  001126           CMP    $GDDAT,$BDDAT      ;IS IT CORRECT?
020520  001403                           BEQ    2$                 ;BR IF SO
020522  010037  001164                   MOV    R0,$TMP0           ;SET UP Y VECTOR LENGTH FOR ER TYPE
020526  104065                           ERROR  65                 ;Y DELTA LENGTH INCORRECT
020530  005237  001124           2$:     INC    $GDDAT             ;BUMP EXPECTED DELTA LENGTH
020534  005200                           INC    R0                 ;ADVANCE TO NEXT Y VECTOR VALUE
020536  032700  002000                   BIT    #2000,R0           ;HAVE WE DONE 10 BITS?
020542  001753                           BEQ    1$                 ;BR IF NOT

                                 ;;*****************************************************************
                                 ;*TEST 101        TEST THAT DELTA LENGTH FOLLOWS THE LONGER OF X OR Y WITH MAINT SW3 SET
                                 ;;*****************************************************************
020544  000004                   TST101: SCOPE
020546  012737  000040  001166           MOV    #40,$TIMES         ;;DO 40 ITERATIONS
020554  012737  000101  001206           MOV    #101,$TESTN        ;;SET TEST NUMBER IN APT MAIL BOX
020562  012737  020642  001110           MOV    #1$,$LPERR         ;SET UP SCOPE LOOP ADRS
020570  013702  002252                   MOV    YPOS,R2            ;R2 CONTAINS REG ADRS OF DELTA (X OR Y POS REG)
020574  012700  032342                   MOV    #BUFFER,R0         ;GET INSTR ADRS POINTER
020600  012720  154024                   MOV    #154024,(R0)+      ;SET UP UNITY VECTOR SCALE
020604  012720  114000                   MOV    #114000,(R0)+      ;STORE POINT INSTR
020610  005020                           CLR    (R0)+              ;X=0
020612  005020                           CLR    (R0)+              ;Y=0
020614  012720  110000                   MOV    #110000,(R0)+      ;STORE LONG VECTOR INSTR
020620  005020                           CLR    (R0)+              ;X TO BE UPDATED
020622  005020                           CLR    (R0)+              ;Y TO BE UPDATED
020624  012710  172000                   MOV    #172000,(R0)       ;STORE STOP INSTR
020630  012700  001777                   MOV    #1777,R0           ;R0 CONTAINS Y VECTOR LENGTH
020634  005001                           CLR    R1                 ;R1 CONTAINS X VECTOR LENGTH
020636  010037  001124                   MOV    R0,$GDDAT          ;CONTAINS EXPECTED DELTA LENGTH
020642  010037  032356           1$:     MOV    R0,BUFFER+14       ;SET UP Y VECTOR LENGTH (1777 TO 0)
020646  010137  032354                   MOV    R1,BUFFER+12       ;SET UP X VECTOR LENGTH (0 TO 1777)
020652  004537  025020                   JSR    R5,EXECUTE         ;GO START DISPLAY
020656  000007                           7                         ;RESUME COUNT

020660  010237  001122                   MOV    R2,$BDADR          ;SET UP REG ADRS WHERE DELTA IS
020664  011237  001126                   MOV    (R2),$BDDAT        ;GREATER OF X OF Y DELTA READ FROM X OR Y POS REG
020670  023737  001124  001126           CMP    $GDDAT,$BDDAT      ;IS IT CORRECT?
020676  001404                           BEQ    2$                 ;BR IF SO
020700  013737  001124  001164           MOV    $GDDAT,$TMP0       ;SET UP LONGEST X OR Y VECTOR LENGTH FOR ER TYPE
020706  104065                           ERROR  65                 ;DELTA LENGTH DOES NOT EQUAL LONGER OF X OR Y
020710  005300                   2$:     DEC    R0                 ;DECREASE Y VECTOR VALUE BY 1
020712  100413                           BMI    TST102             ;;TO NEXT TEST IF 10 BITS OF X & Y HAVE BEEN TESTED
020714  005201                           INC    R1                 ;INCREASE X VECTOR VALUE BY 1
```

```
020716  020001                         CMP     R0,R1            ;LOOK FOR THE LARGEST OF X OR Y
020720  101005                         BHI     3$               ;BR IF Y VECTOR STILL GREATER THAN X
020722  013702  002250                 MOV     XPOS,R2          ;NOW DELTA WILL BE IN XPOS
020726  010137  001124                 MOV     R1,$GDDAT        ;DELTA LENGTH WILL BE THAT OF X VECTOR
020732  000743                         BR      1$               ;TRY NEXT VECTOR COMBINATION
020734  010037  001124         3$:     MOV     R0,$GDDAT        ;DELTA LENGTH WILL BE THAT OF Y VECTOR
020740  000740                         BR      1$               ;TRY NEXT VECTOR COMBINATION

                               ;;*************************************************************
                               ;*TEST 102      CHECK TANGENT WHEN INCREMENTING Y AT 1777 X WITH MAINT SW3 SET
                               ;;*************************************************************
020742  000004                 TST102: SCOPE
020744  012737  000040  001166         MOV     #40,$TIMES       ;;DO 40 ITERATIONS
020752  012737  000102  001206         MOV     #102,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
020760  012737  021046  001110         MOV     #2$,$LPERR       ;SET UP SCOPE LOOP ADRS
020766  013702  002252                 MOV     YPOS,R2          ;R2 CONTAINS REG ADRS OF TANGENT (X OR Y POS REG)
020772  013701  002262                 MOV     YDOFF,R1         ;R1 CONTAINS REG ADRS OF HI ORDER TANGENT
020776  012700  032342                 MOV     #BUFFER,R0       ;GET INSTR ADRS POINTER
021002  012720  154024                 MOV     #154024,(R0)+    ;SET UP UNITY VECTOR SCALE
021006  012720  114000                 MOV     #114000,(R0)+    ;STORE POINT INSTR
021012  005020                         CLR     (R0)+            ;X=0
021014  005020                         CLR     (R0)+            ;Y=0
021016  012720  110000                 MOV     #110000,(R0)+    ;STORE LONG VECTOR INSTR
021022  012720  001777                 MOV     #1777,(R0)+      ;X=1777
021026  005020                         CLR     (R0)+            ;Y TO BE UPDATED
021030  012710  172000                 MOV     #172000,(R0)     ;STORE STOP INSTR
021034  005000                         CLR     R0               ;R0 CONTAINS INCREMENTING Y VECTOR LENGTH (0-1777)
021036  010037  032356         1$:     MOV     R0,BUFFER+14     ;SET UP Y VECTOR LENGTH (0-1777)
021042  004737  024706                 JSR     PC,CALTAN        ;GO FIGURE OUT WHAT TAN SHOULD BE
021046  004537  025020         2$:     JSR     R5,EXECUTE       ;GO START DISPLAY
021052  000007                         7                        ;RESUME COUNT

021054  010237  001122                 MOV     R2,$BDADR        ;SET UP REG ADRS WHERE TANGENT IS
021060  011237  001126                 MOV     (R2),$BDDAT      ;TANGENT IS READ FROM X OR Y POSITION REG
021064  011105                         MOV     (R1),R5          ;READ HI ORDER TANGENT BITS
021066  004737  024662                 JSR     PC,TANCON        ;GO MAKE UP 12 BIT TANGENT
021072  023737  001124  001126         CMP     $GDDAT,$BDDAT    ;IS IT CORRECT?
021100  001403                         BEQ     3$               ;BR IF SO
021102  010037  001164                 MOV     R0,$TMP0         ;SET UP Y VECTOR LENGTH FOR ER TYPE
021106  104066                         ERROR   66               ;INCORRECT TANGENT FOR X & Y VALUE INDICATED
021110  005200                 3$:     INC     R0               ;INCREASE X VECTOR VALUE BY 1
021112  022700  001777                 CMP     #1777,R0         ;R0 UP TO 1777 YET?
021116  101347                         BHI     1$               ;BR IF NOT
021120  103405                         BCS     TST103           ;;NEXT TEST IF R0 = 2000
021122  013702  002250                 MOV     XPOS,R2          ;TANGENT WILL NOW BE IN XPOS
021126  013701  002260                 MOV     XDOFF,R1         ;HI ORDER TANGENT BITS NOW IN XDOFF
021132  000741                         BR      1$               ;ONE MORE TIME

                               ;;*************************************************************
                               ;*TEST 103      CHECK TANGENT WHEN INCREMENTING X AT 1777 Y WITH MAINT SW3 SET
                               ;;*************************************************************
021134  000004                 TST103: SCOPE
021136  012737  000040  001166         MOV     #40,$TIMES       ;;DO 40 ITERATIONS
021144  012737  000103  001206         MOV     #103,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
021152  012737  021240  001110         MOV     #2$,$LPERR       ;SET UP SCOPE LOOP ADRS
021160  013702  002250                 MOV     XPOS,R2          ;R2 CONTAINS REG ADRS OF TANGENT (X OR Y POS REG)
```

```
021164 013701 002260                 MOV     XDOFF,R1          ;R1 CONTAINS REG ADRS OF HI ORDER TANGENT
021170 012700 032342                 MOV     #BUFFER,R0        ;GET INSTR ADRS POINTER
021174 012720 154024                 MOV     #154024,(R0)+     ;SET UP UNITY VECTOR SCALE
021200 012720 114000                 MOV     #114000,(R0)+     ;STORE POINT INSTR
021204 005020                        CLR     (R0)+             ;X=0
021206 005020                        CLR     (R0)+             ;Y=0
021210 012720 110000                 MOV     #110000,(R0)+     ;STORE LONG VECTOR INSTR
021214 005020                        CLR     (R0)+             ;X TO BE UPDATED
021216 012720 001777                 MOV     #1777,(R0)+       ;Y=1777
021222 012710 172000                 MOV     #172000,(R0)      ;STORE STOP INSTR
021226 005000                        CLR     R0                ;R0 CONTAINS INCREMENTING X VECTOR LENGTH (0-1777)
021230 010037 032354         1$:     MOV     R0,BUFFER+12      ;SET UP X VECTOR LENGTH (0-1777)
021234 004737 024706                 JSR     PC,CALTAN         ;GO FIGURE OUT WHAT TAN SHOULD BE
021240 004537 025020         2$:     JSR     R5,EXECUTE        ;GO START DISPLAY
021244 000007                        7                         ;RESUME COUNT

021246 010237 001122                 MOV     R2,$BDADR         ;SET UP REG ADRS WHERE TANGENT IS
021252 011237 001126                 MOV     (R2),$BDDAT       ;TANGENT IS READ FROM X OR Y POSITION REG
021256 011105                        MOV     (R1),R5           ;READ HI ORDER TANGENT BITS
021260 004737 024662                 JSR     PC,TANCON         ;GO MAKE UP 12 BIT TANGENT
021264 023737 001124 001126          CMP     $GDDAT,$BDDAT     ;IS IT CORRECT?
021272 001403                        BEQ     3$                ;BR IF SO
021274 010037 001164                 MOV     R0,$TMP0          ;SET UP X VECTOR LENGTH FOR ER TYPE
021300 104066                        ERROR   66                ;INCORRECT TANGENT FOR X & Y VALUE INDICATED
021302 005200                3$:     INC     R0                ;INCREASE Y VECTOR VALUE BY 1
021304 022700 002000                 CMP     #2000,R0          ;R0 UP TO 2000 YET?
021310 001347                        BNE     1$                ;BR IF NOT

                             ;;*******************************************************************
                             ;*TEST 104      TEST THAT DPC BITS 16 & 17 WILL SET & RESET
                             ;;*******************************************************************
021312 000004                TST104: SCOPE
021314 012737 000040 001166          MOV     #40,$TIMES        ;;DO 40 ITERATIONS
021322 012737 000104 001206          MOV     #104,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
021330 012737 000060 001124          MOV     #60,$GDDAT        ;EXPECT BOTH BITS INITIALLY
021336 012700 006000                 MOV     #6000,R0          ;PUT RELOCATE VALUE IN R0
021342 012737 021350 001110          MOV     #1$,$LPERR        ;SET UP SCOPE LOOP ADRS
021350 012777 010000 160720  1$:     MOV     #MAINT1,@STKPT    ;SET MAINT SW 1
021356 010077 160672                 MOV     R0,@RLO           ;SET UP RELOCATE VALUE
021362 005077 160656                 CLR     @DPC              ;SET UP DPC ADRS & START DISPLAY
021366 013737 002256 001122          MOV     SREG1,$BDADR      ;SET UP REG ADRS 12
021374 017737 160656 001126          MOV     @SREG1,$BDDAT     ;READ REG 12
021402 042737 177717 001126          BIC     #177717,$BDDAT    ;SAVE ONLY DPC BITS 17 & 16
021410 023737 001124 001126          CMP     $GDDAT,$BDDAT     ;ARE THEY CORRECT?
021416 001401                        BEQ     2$                ;BR IF OK
021420 104067                        ERROR   67                ;RELOCATE VALUE FAILED TO SET UP DPC 16 OR 17
021422 162737 000020 001124  2$:     SUB     #20,$GDDAT        ;SET UP NEXT ADRS VALUE AT EXPECTED LOC
021430 162700 002000                 SUB     #2000,R0          ;SET UP NEXT ADRS VALUE AT R0
021434 100345                        BPL     1$                ;BR IF MORE ADRS TO TEST
021436 005037 001110                 CLR     $LPERR            ;NO RESET SCOPE LOOP
021442 005037 001124                 CLR     $GDDAT            ;EXPECT ZERO
021446 012777 006000 160600          MOV     #6000,@RLO        ;SET BOTH BITS
021454 005077 160564                 CLR     @DPC              ;CLR DPC & START - NO NPR
021460 000005                        RESET                     ;CLR DPC 16 & 17
021462 017737 160570 001126          MOV     @SREG1,$BDDAT     ;READ REG 12
021470 042737 177717 001126          BIC     #177717,$BDDAT    ;SAVE ONLY DPC 16 & 17
```

```
021476  001401                                  BEQ     3$              ;BR IF CLEARED
021500  104067                                  ERROR   67              ;RESET FAILED TO CLEAR DPC 16 OR 17
021502  005077  160546                  3$:     CLR     @RLO            ;CLR RELOCATE REG BEFORE ADVANCING

                                        ;;*************************************************************
                                        ;*TEST 105      TEST THAT DPC GETS PUSHED ONTO STACK AT HIGHEST BOUNDRY BELOW 28K
                                        ;;*************************************************************
021506  000004                          TST105: SCOPE
021510  012737  000105  001206                  MOV     #105,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
021516  005737  002216                          TST     CENAB           ;DO WE HAVE MORE THAN 8K?
021522  001477                                  BEQ     TST106          ;;IF NOT - NEXT TEST
021524  012737  021610  001110          1$:     MOV     #3$,$LPERR      ;SET UP SCOPE LOOP ADRS
021532  013737  002272  001122                  MOV     STKVAL,$BDADR   ;SET UP REG 26 ADRS
021540  005002                                  CLR     R2              ;CLR ADRS ROTATIONAL VALUE
021542  013700  002214                          MOV     MEMMAX,R0       ;GET LAST 4K MEM BLK
021546  010001                                  MOV     R0,R1           ;PUT IT INTO R1
021550  010003                                  MOV     R0,R3           ;PUT IT INTO R3 ALSO
021552  062701  020000                          ADD     #20000,R1       ;POINT TO 1ST NON-EXISTENT MEM ADRS IN R1
021556  012704  020040                  2$:     MOV     #20040,R4       ;SET UP TOP OF STACK INITIALLY IN R4
021562  012737  000034  001164                  MOV     #34,$TMP0       ;SET UP STACK LEVEL WHERE JSR'ED DPC IS STORED
021570  010337  001124                          MOV     R3,$GDDAT       ;SAVE EXPECTED JSR RETURN ADRS OF STK
021574  162703  000002                          SUB     #2,R3           ;REDUCE ADRS BY 2 TO ALLOW FOR JSR REL INSTR
021600  011337  002210                          MOV     (R3),LDRSV1     ;SAVE LOCATION
021604  012713  163400                          MOV     #163400,(R3)    ;LOAD UP JSR REL INSTR
021610  010477  160462                  3$:     MOV     R4,@STKPT       ;SET STK PTR
021614  010377  160424                          MOV     R3,@DPC         ;START AT ADRS IN R3
021620  004537  024426                          JSR     R5,DELAY        ;STALL FOR STACKING
021624  000001                                  BIT0                    ;COUNT TO ONE
021626  013777  001164  160442                  MOV     $TMP0,@STKPT    ;SEL STK LEVEL OF STORED DPC
021634  017737  160432  001126                  MOV     @STKVAL,$BDDAT  ;READ STORED JSR RETURN ADRS
021642  023737  001124  001126                  CMP     $GDDAT,$BDDAT   ;IS IT CORRECT?
021650  001401                                  BEQ     4$              ;BR IF SO
021652  104005                                  ERROR   5               ;JSR REL FAILED TO PUSH ON STK THE RETURN ADRS
021654  162704  000004                  4$:     SUB     #4,R4           ;ADVANCE TO NEXT STK LEVEL
021660  162737  000004  001164                  SUB     #4,$TMP0        ;HAVE ALL 8 LEVELS BEEN TESTED?
021666  100350                                  BPL     3$              ;BR IF NOT
021670  013713  002210                          MOV     LDRSV1,(R3)     ;RESTORE MEM LOC
021674  005702                                  TST     R2              ;IS THIS THE 1ST ADRS PASS?
021676  001003                                  BNE     5$              ;BR IF NOT
021700  012702  000002                          MOV     #2,R2           ;SET UP ADRS ROTATIONAL FACTOR IN R2
021704  000401                                  BR      6$              ;THIS TIME DON'T ROTATE
021706  006302                          5$:     ASL     R2              ;SELECT NEXT ADRS BIT TO LEFT
021710  010003                          6$:     MOV     R0,R3           ;GET HIGHEST ADRS BOUNDRY
021712  060203                                  ADD     R2,R3           ;ADD IN NEXT ADRS BIT FOR TEST
021714  103402                                  BCS     TST106          ;;TO NEXT TEST IF ADRS GREATER THAN 16 BITS
021716  020103                                  CMP     R1,R3           ;IS THERE ROOM FOR THIS ADRS BIT?
021720  103316                                  BCC     2$              ;BR IF SO

                                        ;;*************************************************************
                                        ;*TEST 106      TEST THAT DPC GETS RESTORED FROM STACK AT HIGHEST BOUNDRY BELOW 28K
                                        ;;*************************************************************
021722  000004                          TST106: SCOPE
021724  012737  000106  001206                  MOV     #106,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
021732  005737  002216                          TST     CENAB           ;DO WE HAVE MORE THAN 8K?
021736  001516                                  BEQ     TST107          ;;IF NOT - NEXT TEST
021740  012737  022040  001110          1$:     MOV     #3$,$LPERR      ;SET UP SCOPE LOOP ADRS
```

```
021746  005002                            CLR     R2                ;CLR ADRS ROTATIONAL VALUE
021750  013700  002214                    MOV     MEMMAX,R0         ;GET LAST 4K MEM BLK
021754  010001                            MOV     R0,R1             ;PUT IT INTO R1
021756  010003                            MOV     R0,R3             ;PUT IT INTO R3 ALSO
021760  062701  020000                    ADD     #20000,R1         ;POINT TO 1ST NON-EXISTENT MEM ADRS IN R1
021764  012737  166000  032342            MOV     #166000,BUFFER    ;SET UP POP INSTR IN LOWEST 8K
021772  012704  020040          2$:       MOV     #20040,R4         ;SET TOP OF STACK INITIALLY IN R4
021776  012737  000034  001164            MOV     #34,$TMPO         ;SET UP STK LEVEL WHERE JSR'ED DPC IS STORED
022004  010337  001124                    MOV     R3,$GDDAT         ;SAVE EXPECTED JSR RETURN ADRS
022010  162703  000004                    SUB     #4,R3             ;REDUCE ADRS BY 4 TO ALLOW FOR JSR ABS INSTR
022014  011337  002210                    MOV     (R3),LDRSV1       ;SAVE LOCATION
022020  016337  000002  002212            MOV     2(R3),LDRSV2      ;SAVE LOCATION
022026  012713  162000                    MOV     #162000,(R3)      ;LOAD UP JSR ABS INSTR
022032  012763  032342  000002            MOV     #BUFFER,2(R3)     ;LOAD UP ABS ADRS TO POP INSTR IN LOWEST 8K
022040  010477  160232          3$:       MOV     R4,@STKPT         ;SET UP CURRENT STACK LEVEL
022044  010377  160174                    MOV     R3,@DPC           ;START AT ADRS IN R3
022050  013737  002244  001122            MOV     DPC,$BDADR        ;SET UP REG ADRS 00
022056  005277  160162                    INC     @DPC              ;PICK UP ABS ADRS (POP INSTR ADRS)
022062  004537  024426                    JSR     R5,DELAY          ;DELAY FOR STACKING
022066  000001                            BIT0                      ;COUNT TO ONE
022070  005277  160150                    INC     @DPC              ;PICK UP POP INSTR
022074  004537  024426                    JSR     R5,DELAY          ;STALL FOR POPPING
022100  000001                            BIT0                      ;COUNT TO ONE
022102  017737  160136  001126            MOV     @DPC,$BDDAT       ;GET RETURN ADRS FOR DPC
022110  023737  001124  001126            CMP     $GDDAT,$BDDAT     ;IS IT CORRECT?
022116  001401                            BEQ     4$                ;BR IF SO
022120  104035                            ERROR   35                ;DPC FAILED TO RESTORE AT ADRS & STK LEVEL INDICATED
022122  162704  000004          4$:       SUB     #4,R4             ;SET UP NEXT STK SELECTION LEVEL
022126  162737  000004  001164            SUB     #4,$TMPO          ;HAVE WE GONE THRU 8 LEVELS THIS ADRS
022134  100341                            BPL     3$                ;BR IF NOT
022136  013723  002210                    MOV     LDRSV1,(R3)+      ;RESTORE MEM LOC
022142  013713  002212                    MOV     LDRSV2,(R3)       ;RESTORE MEM LOC
022146  005702                            TST     R2                ;IS THIS THE 1ST ADRS PASS?
022150  001003                            BNE     5$                ;BR IF NOT
022152  012702  000002                    MOV     #2,R2             ;SET UP ADRS ROTATIONAL FACTOR IN R2
022156  000401                            BR      6$                ;THIS TIME DON'T ROTATE
022160  006302                  5$:       ASL     R2                ;SELECT NEXT ADRS BIT TO LEFT
022162  010003                  6$:       MOV     R0,R3             ;GET HIGHEST 4K BOUNDRY ADRS
022164  060203                            ADD     R2,R3             ;ADD IN NEXT ADRS BIT FOR TEST
022166  103402                            BCS     TST107            ;;TO NEXT TEST IF ADRS GREATER THAN 16 BITS
022170  020103                            CMP     R1,R3             ;IS THERE ROOM FOR THIS ADRS BIT?
022172  103277                            BCC     2$                ;BR AND EXERCISE THIS BIT IF SO

                                ;;*****************************************************************
                                ;*TEST 107       TEST THAT THE VS60 CAN OPERATE ABOVE 28K
                                ;;*****************************************************************
022174  000004                  TST107: SCOPE
022176  012737  000040  001166            MOV     #40,$TIMES        ;;DO 40 ITERATIONS
022204  012737  000107  001206            MOV     #107,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

                                ;THIS TEST USES THE GRAPHPLOT INCREMENT PART OF THE 'LOAD STATUS B'
                                ;INSTR TO HOLD THE CURRENT 4K MEMORY BLOCK ADRS BEING ADDRESSED
                                ;ABOVE 28K.  THE 6 BITS OF THE GRAPHPLOT HAVE THE FOLLOWING RELATIONSHIP:
                                ;16=32K, 20=36K, 22=40K, 24=44K, ETC. UP TO 76=128K.  THE VS60 PREFORMS
                                ;THE FOLLOWING INSTRS IN EA 4K BLK ABOVE 28K - JSR,POP & 'LOAD STATUS
                                ;B' (LD GRAPHPLOT INC).  AN ERROR IS REPORTED IF THE GRAPHPLOT VALUE
```

```
                                           ;RECEIVED IS DIFFERENT THAN THE VALUE WRITTEN TO THE 4K BLK UNDER TEST.

022212  005737  002220                TST     KTENAB          ;DO WE HAVE UPPER CORE?
022216  001547                        BEQ     TST110          ;;IF NOT GO TO NEXT TEST
022220  100437                1$:     BMI     3$              ;BR IF MEM ALREADY LOADED WITH DISPLAY CODE
022222  052737  100000  002220        BIS     #100000,KTENAB  ;DO IT 1ST TIME ONLY
022230  012777  001600  157776        MOV     #1600,@KIPAR2   ;START AT 28K
022236  012701  174116                MOV     #174116,R1      ;GRAPHPLOT VALUE WILL CONTAIN 4K MEM BLK NO
022242  012700  040000        2$:     MOV     #40000,R0       ;USE ADRS ZERO & KIPAR2
022246  052737  001400  177572        BIS     #1400,@#KTSR0   ;ENABLE KT MAINT MODE
022254  012720  163001                MOV     #163001,(R0)+   ;SET UP JSR REL INSTR
022260  010120                        MOV     R1,(R0)+        ;SET UP LOAD STATUS B INSTR
022262  012710  166000                MOV     #166000,(R0)    ;SET UP POP INSTR
022266  042737  001400  177572        BIC     #1400,@#KTSR0   ;DISABLE KT
022274  027737  157734  002222        CMP     @KIPAR2,KTMAX   ;HAS INSTR CODE BEEN LOADED THRU CODE
022302  001406                        BEQ     3$              ;BR IF SO
022304  062777  000200  157722        ADD     #200,@KIPAR2    ;ADVANCE TO NEXT 4K BLK
022312  062701  000002                ADD     #2,R1           ;RECORED MEM BLK IN GRAPHPLOT INC INSTR
022316  000751                        BR      2$              ;GO SET UP INSTRS IN NEXT 4K BLK OF CORE
022320  012777  001600  157706 3$:    MOV     #1600,@KIPAR2   ;SET UP 28K ADRS
022326  005077  157722        4$:     CLR     @RL0            ;CLR RELOCATE REG
022332  017700  157676                MOV     @KIPAR2,R0      ;GET CURRENT 4K BLK ADRS
022336  004737  024556                JSR     PC,DPCONV       ;GO CONVERT 4K BLK ADRS INTO 18 BIT ADRS
022342  010037  001124                MOV     R0,$GDDAT       ;SAVE 16 BIT ADRS
022346  010137  002224                MOV     R1,G1716        ;SAVE ADRS BITS 17 & 16
022352  006201                        ASR     R1              ;NOW SET UP RELOCATION REG
022354  103003                        BCC     5$              ;BR IF ADRS BIT 16 IS NOT SET
022356  052777  002000  157670        BIS     #2000,@RL0      ;SET UP RELOCATION REG BIT 10
022364  006201                5$:     ASR     R1              ;LOOK AT ADRS BIT 17
022366  103003                        BCC     6$              ;BR IF ADRS BIT 17 IS NOT SET
022370  052777  004000  157656        BIS     #4000,@RL0      ;SET UP RELOCATION REG BIT 11
022376  012777  020040  157672 6$:    MOV     #20040,@STKPT   ;SET TOP OF STACK
022404  013777  001124  157632        MOV     $GDDAT,@DPC     ;START
022412  004537  024426                JSR     R5,DELAY        ;GO STALL FOR STACKING
022416  000001                        BIT0                    ;COUNT TO 1
022420  005277  157620                INC     @DPC            ;RESUME
022424  004537  024426                JSR     R5,DELAY        ;GO STALL FOR POPPING
022430  000001                        BIT0                    ;COUNT TO 1
022432  005277  157606                INC     @DPC            ;RESUME
022436  012737  022376  001110        MOV     #6$,$LPERR      ;SET UP SCOPE LOOP ADRS
022444  017700  157600                MOV     @XPOS,R0                ;GET ADRS BLK FROM GRAPHPLOT INC BITS
022450  042700  001777                BIC     #1777,R0        ;SAVE ONLY ADRS BLK INFORMATION READ
022454  005001                        CLR     R1              ;CLR HI ORDER ADRS
022456  004737  024570                JSR     PC,DPCON1       ;GO CONVERT BLK ADRS TO A 18 BIT ADRS
022462  010037  001126                MOV     R0,$BDDAT       ;SAVE 16 BIT ADRS READ
022466  010137  002226                MOV     R1,B1716        ;SAVE ADRS BIT 17 & 16
022472  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;IS THE 16 BIT ADRS CORRECT?
022500  001004                        BNE     7$              ;BR IF NOT
022502  023737  002224  002226        CMP     G1716,B1716     ;ADRS BIT 17 & 16 CORRECT?
022510  001401                        BEQ     8$              ;BR IF SO
022512  104070                7$:     ERROR   70              ;VS60 FAILED TO EXECUTE INSTRS AT ADRS INDICATED
022514  062777  000200  157512 8$:    ADD     #200,@KIPAR2    ;SET NEXT 4K BLK
022522  027737  157506  002222        CMP     @KIPAR2,KTMAX   ;HAS INSTR CODE BEEN EXECUTED IN ALL 4K AVAIL BLKS?
022530  101676                        BLOS    4$              ;BR IF NOT
022532  005077  157516                CLR     @RL0            ;INSURE RELOCATE REG 0 BEFORE ADVANCING
```

```
                                          ;;*************************************************************
                                          ;*TEST 110          SILO TEST 1 - ALL STATUS READABLE FROM BUS DIRECTLY
                                          ;;*************************************************************
022536  000004                            TST110: SCOPE
022540  012737  000040  001166                    MOV     #40,$TIMES         ;;DO 40 ITERATIONS
022546  012737  000110  001206                    MOV     #110,$TESTN        ;;SET TEST NUMBER IN APT MAIL BOX
022554  012700  032342                            MOV     #BUFFER,R0         ;GET ADRS WHERE INSTRS GO
022560  012720  152525                            MOV     #152525,(R0)+      ;NAME OF 2525
022564  012720  155661                            MOV     #155661,(R0)+      ;ROTATE,CHAR SC=1,VEC SC=1
022570  012720  164374                            MOV     #164374,(R0)+      ;CONS0=INT,L.P. INTR EN,L.P. SW INTR EN
022574  012720  164750                            MOV     #164750,(R0)+      ;CONS1=INT,-L.P. INTR EN,-L.P. SW INTR EN
022600  012720  170063                            MOV     #170063,(R0)+      ;ITALICS,MENU
022604  012720  175200                            MOV     #175200,(R0)+      ;COLOR=1
022610  012720  112235                            MOV     #112235,(R0)+      ;LONG VEC,INT=1,BLINK,LINE=1,MODE=2
022614  005020                                    CLR     (R0)+              ;X=0
022616  005020                                    CLR     (R0)+              ;Y=0
022620  012720  164000                            MOV     #164000,(R0)+      ;NOP
022624  012720  151252                            MOV     #151252,(R0)+      ;NAME=1252
022630  012720  155322                            MOV     #155322,(R0)+      ;-ROTATE,CHAR SC=2,VEC SC=2
022634  012720  164270                            MOV     #164270,(R0)+      ;CONS0=-INT,L.P. INTR EN,-L.P. SW INTR EN
022640  012720  164674                            MOV     #164674,(R0)+      ;CONS1=-INT,L.P. INTR EN,L.P. SW INTR EN
022644  012720  170042                            MOV     #170042,(R0)+      ;-ITALICS,-MENU
022650  012720  175400                            MOV     #175400,(R0)+      ;COLOR=2
022654  012720  146426                            MOV     #146426,(R0)+      ;ABS VEC,INT=2,-BLINK,LINE=2,MODE=11
022660  005020                                    CLR     (R0)+              ;X=0
022662  005020                                    CLR     (R0)+              ;Y=0

022664  012720  164000                            MOV     #164000,(R0)+      ;NOP
022670  012720  153070                            MOV     #153070,(R0)+      ;NAME=3070
022674  012720  155664                            MOV     #155664,(R0)+      ;ROTATE,CHAR SC=1,VEC SC=4
022700  012720  164374                            MOV     #164374,(R0)+      ;CONS0=INT,L.P. INTR EN,L.P. SW INTR EN
022704  012720  164750                            MOV     #164750,(R0)+      ;CONS1=INT,-L.P. INTR EN,-L.P. SW INTR EN
022710  012720  170063                            MOV     #170063,(R0)+      ;ITALICS,MENU
022714  012720  175200                            MOV     #175200,(R0)+      ;COLOR=1
022720  012720  113035                            MOV     #113035,(R0)+      ;LONG VEC,INT=4,BLINK,LINE=1,MODE=2
022724  005020                                    CLR     (R0)+              ;X=0
022726  005020                                    CLR     (R0)+              ;Y=0
022730  012720  164000                            MOV     #164000,(R0)+      ;NOP
022734  012720  150707                            MOV     #150707,(R0)+      ;NAME=0707
022740  012720  155330                            MOV     #155330,(R0)+      ;-ROTATE,CHAR SC=2,VEC SC=10
022744  012720  164270                            MOV     #164270,(R0)+      ;CONS0=-INT,L.P. INTR EN,-L.P. SW INTR EN
022750  012720  164674                            MOV     #164674,(R0)+      ;CONS1=-INT,L.P. INTR EN,L.P. SW INTR EN
022754  012720  175400                            MOV     #175400,(R0)+      ;COLOR=2
022760  012720  170042                            MOV     #170042,(R0)+      ;-ITALICS,-MENU
022764  012720  146026                            MOV     #146026,(R0)+      ;ABS VEC,INT=0,-BLINK,LINE=2,MODE=11
022770  005020                                    CLR     (R0)+              ;X=0
022772  005020                                    CLR     (R0)+              ;Y=0
022774  012720  161730                            MOV     #161730,(R0)+      ;JMP REL TO BUFFER START

023000  013700  001252                            MOV     $VECT1,R0          ;GET BASIC VECTOR ADRS
023004  012760  023166  000004                    MOV     #3$,4(R0)          ;SET UP INTR RETURN ADRS
023012  012760  000340  000006                    MOV     #340,6(R0)         ;SET PRIORITY TO HIGHEST LEVEL ON INTR
023020  012701  023416                            MOV     #BSTART,R1         ;R1 CONTAINS ADRS PTR OF DISPLAY STARTS
023024  012702  023330                            MOV     #SILOR,R2          ;R2 CONTAINS ADRS PTR OF EXPECTED SILO RESULTS
023030  012703  023430                            MOV     #RCNT,R3           ;R3 CONTAINS ADRS PTR OF # OF RESUMES FOR EA SILO LEVEL
023034  012704  023402                            MOV     #BADR,R4           ;R4 CONTAINS ADRS PTR WHERE REG DATA IS EXPECTED
```

```
023040  012705  023330                  MOV     #SILOR,R5           ;R5 WILL UPDATE R2 ON NEXT BUF START ADRS
023044  005037  001164                  CLR     $TMPO               ;$TMPO CONTAINS SILO LEVEL UNDER TEST
023050  012777  020040  157220  1$:     MOV     #20040,@STKPT       ;TEST DONE WITH MAINT SW 2
023056  012737  000340  177776          MOV     #340,PSW            ;WANT PRIORITY LEVEL AT TOP
023064  011177  157154                  MOV     (R1),@DPC           ;START
023070  011300                          MOV     (R3),R0             ;SET UP RESUME CNTR
023072  005277  157146          2$:     INC     @DPC                ;RESUME
023076  012737  023050  001110          MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
023104  005300                          DEC     R0                  ;COUNT RESUME
023106  001371                          BNE     2$                  ;BR IF CNT NOT TO ZERO
023110  052777  040000  157160          BIS     #40000,@STKPT       ;SET MAINT SW 3
023116  052777  040000  157142          BIS     #40000,@CONS        ;SET L.P. FLAG 00
023124  005037  177776                  CLR     PSW                 ;ALLOW INTR
023130  021616                          CMP     (SP),(SP)           ;LET CPU GRANT REQ
023132  021616                          CMP     (SP),(SP)           ;
023134  021616                          CMP     (SP),(SP)           ;
023136  021616                          CMP     (SP),(SP)           ;
023140  013737  002266  001122          MOV     CONS,$BDADR         ;SET UP REG ADRS 22
023146  012737  144000  001124          MOV     #144000,$GDDAT      ;EXPECT L. P. FLAG
023154  017737  157106  001126          MOV     @CONS,$BDDAT        ;READ REG 22
023162  104001                          ERROR   1                   ;L.P. FLAG INTR FAILED USING MAINT MODE 3
023164  000455                          BR      7$                  ;NEXT TEST ON INTR FAILURE
023166  022626                  3$:     CMP     (SP)+,(SP)+         ;FIX STACK SINCE NO RTI ON INTR
023170  017437  000000  001122  4$:     MOV     @0(R4),$BDADR       ;GET BUS ADRS WHERE DATA IS
023176  017737  155720  001126          MOV     @$BDADR,$BDDAT      ;GET DATA FROM REG
023204  011237  001124                  MOV     (R2),$GDDAT         ;GET EXPECTED DATA FROM SILOR TABLE
023210  023737  001124  001126          CMP     $GDDAT,$BDDAT       ;CORRECT?
023216  001401                          BEQ     5$                  ;BR IF SO
023220  104002                          ERROR   2                   ;SILO FAILURE AT LEVEL INDICATED
023222  062702  000002          5$:     ADD     #2,R2               ;BUMP ADRS PTR OF EXPECTED SILO RESULTS
023226  062704  000002                  ADD     #2,R4               ;BUMP ADRS PTR OF REG ADRS WHERE DATA IS READ
023232  005714                          TST     (R4)                ;DONE CHECKS?
023234  001355                          BNE     4$                  ;BR IF NOT
023236  012704  023402                  MOV     #BADR,R4            ;RESET REG ADRS PTR
023242  005712                          TST     (R2)                ;AT END OF RESULT TABLE?
023244  001002                          BNE     6$                  ;BR IF NOT
023246  012702  023330                  MOV     #SILOR,R2           ;RESET ADRS PTR OF EXPECTED RESULTS
023252  062703  000002          6$:     ADD     #2,R3               ;POINT TO NEXT RESUME CNT FOR NEXT SILO LEVEL
023256  005237  001164                  INC     $TMPO               ;ADVANCE TO NEXT SILO LEVEL
023262  022737  000004  001164          CMP     #4,$TMPO            ;ALL 4 LEVELS TESTED AT THIS STARTING ADRS?
023270  001267                          BNE     1$                  ;BR IF NOT - REPEAT TEST CHECKING NEXT SILO LEVEL
023272  005037  001164                  CLR     $TMPO               ;RESET SILO LEVEL FOR TESTING WITH DIFF DATA
023276  012703  023430                  MOV     #RCNT,R3            ;RESET RESUME CNT ADRS PTR
023302  062705  000012                  ADD     #12,R5              ;NEW RESULT ADRS FOR NEXT BUF START ADRS
023306  010502                          MOV     R5,R2               ;UPDATE RESULT PTR
023310  062701  000002                  ADD     #2,R1               ;BUMP BUFFER STARTING ADRS PTR - THIS CAUSES
                                                                    ;DIFFERENT DATA PATTERNS THRU SILO
023314  005711                          TST     (R1)                ;ALL 4 BUFFER ADRS STARTS BEEN DONE?
023316  001254                          BNE     1$                  ;BR IF NOT
023320  004737  024626          7$:     JSR     PC,RSTVEC           ;GO RESTORE VECTOR WITH HALT
023324  000005                          RESET                       ;CLR VS60 BEFORE ADVANCING
023326  000445                          BR      TST111              ;;TO NEXT TEST

023330  032364                  SILOR:  BUFFER+22       ;DPC RETURN ADRS
023332  010631                          10631   ;MODE=2,INT=1,L.P. FLAG,ITALICS,BLINK,LINE=1
023334  002501                          2501    ;ROTATE,CHAR SC=1,MENU,VCT CS=1
```

```
023336  147004                          147004  ;CONSO=INT,L.P. FLG,L.P. INTR EN,L.P. SW EN,CONS1=INT,COLOR=1
023340  002525                          2525    ;NAME=2525
023342  032410                          BUFFER+46       ;DPC RETURN ADRS
023344  045202                          45202   ;MODE=11,INT=2,L.P. FLG,LINE=2
023346  001002                          1002    ;CHAR SC=2,VCT SC=2
023350  044070                          44070   ;CONSO=L.P.FLG,L.P. INTR EN,CONS1=L.P. FLG INTR EN,L.P. SW EN,COLOR=2
023352  001252                          1252    ;NAME=1252
023354  032434                          BUFFER+72       ;DPC RETURN ADRS
023356  012231                          12231   ;MODE=2,INT=3,L.P. FLG,ITALICS,BLINK,LINE=1
023360  002504                          2504    ;ROTATE,CHAR SC=1,MENU,VEC SC=4
023362  147004                          147004  ;CONSO=INT,L.P. FLG,L.P. INTR EN,L.P. SW EN,CONS1=INT,COLOR=1
023364  003070                          3070    ;NAME=3070
023366  032460                          BUFFER+116      ;DPC RETURN ADRS
023370  044202                          44202   ;MODE=11,INT=0,L.P. FLG,LINE=2
023372  001010                          1010    ;CHAR SC=2,VCT SC=4
023374  044070                          44070   ;CONSO=L.P. FLG,L.P. INTR EN,CONS1=L.P. FLG FLG EN,L.P. SE EN,COLOR=2
023376  000707                          707     ;NAME=707
023400  000000                          0       ;TERMINATOR

023402  002244                  BADR:   DPC     ;THIS TABLE CONTAINS REGS ADRS WHERE DATA IS EXPECTED
023404  002246                          SREG0
023406  002256                          SREG1
023410  002266                          CONS
023412  002270                          DNAME
023414  000000                          0

023416  032342                  BSTART: BUFFER  ;THIS TABLE CONTAINS DIFFERENT STARTING ADRS FOR TESTING
                                                ;EACH SILO LEVEL
023420  032366                          BUFFER+24
023422  032412                          BUFFER+50
023424  032436                          BUFFER+74
023426  000000                          0

023430  000017                  RCNT:   17      ;THIS TABLE CONTROLS THE AMOUNT OF RESUMES NEEDED TO
                                                ;CYCLE THRU ALL FOUR LEVELS OF SILO
023432  000031                          31
023434  000043                          43
023436  000055                          55
023440  000000                          0


                                ;;*************************************************************
                                ;*TEST 111      SILO TEST 2 - ALL STATUS VERIFIED ON STACK
                                ;;*************************************************************
023442  000004                  TST111: SCOPE
023444  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
023452  012737  000111  001206          MOV     #111,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
023460  012700  032342                  MOV     #BUFFER,R0      ;GET ADRS WHERE INSTR'S GO
023464  012720  171600                  MOV     #171600,(R0)+   ;STOP INTR EN,-L.P. HIT EN
023470  012720  176357                  MOV     #176357,(R0)+   ;QUE CNTRL,-EDGE FLG INTR EN,QUE,CHAR STRING ESC
023474  012720  164770                  MOV     #164770,(R0)+   ;CONS1=INT,L.P. INTR EN
023500  012720  102200                  MOV     #102200,(R0)+   ;CHAR INSTR,INT=1
023504  012720  007015                  MOV     #7015,(R0)+     ;SHIFT OUT & 'CR'
023510  012720  164000                  MOV     #164000,(R0)+   ;NOP OR JSR REL +1 WHEN CHECKING SILO
023514  012720  164000                  MOV     #164000,(R0)+   ;NOP
023520  012720  171300                  MOV     #171300,(R0)+   ;-STOP INTR EN,L.P. HIT EN
023524  012720  176272                  MOV     #176272,(R0)+   ;-QUE CNTRL,EDGE FLG INTR EN,-QUE,-CHAR STRING ESC
```

```
023530  012720  164770                  MOV     #164770,(R0)+           ;CONS1=INT,L.P. INTR EN
023534  012720  102400                  MOV     #102400,(R0)+           ;CHAR INSTR,INT=2
023540  012720  006417                  MOV     #6417,(R0)+             ;SHIFT OUT OFF & 'CR'
023544  012720  164000                  MOV     #164000,(R0)+           ;NOP OR JSR REL +1 WHEN CHECKING SILO
023550  012720  164000                  MOV     #164000,(R0)+           ;NOP
023554  012720  171600                  MOV     #171600,(R0)+           ;STOP INTR EN,-L.P. HIT EN
023560  012720  176357                  MOV     #176357,(R0)+           ;QUE CNTRL,-EDGE FLG INTR EN,QUE,CHAR STRING ESC
023564  012720  164770                  MOV     #164770,(R0)+           ;CONS1=INT,L.P. INTR EN
023570  012720  102600                  MOV     #102600,(R0)+           ;CHAR INSTR,INT=3
023574  012720  007015                  MOV     #7015,(R0)+             ;SHIFT OUT & 'CR'
023600  012720  164000                  MOV     #164000,(R0)+           ;NOP OR JSR REL +1 WHEN CHECKING SILO
023604  012720  164000                  MOV     #164000,(R0)+           ;NOP
023610  012720  171300                  MOV     #171300,(R0)+           ;-STOP INTR EN,L.P. HIT EN
023614  012720  176272                  MOV     #176272,(R0)+           ;-QUE CNTRL,EDGE FLG INTR EN,-QUE,-CHAR STRING ESC
023620  012720  164770                  MOV     #164770,(R0)+           ;CONS1=INT,L.P INTR EN
023624  012720  103000                  MOV     #103000,(R0)+           ;CHAR INSTR,INT=4
023630  012720  006417                  MOV     #6417,(R0)+             ;SHIFT OUT OFF & 'CR'
023634  012720  164000                  MOV     #164000,(R0)+           ;NOP OR JSR REL +1 WHEN CHECKING SILO
023640  012720  161744                  MOV     #161744,(R0)+           ;JMP REL TO BUFFER START
023644  013700  001252                  MOV     $VECT1,R0               ;GET BASIC VECTOR ADRS
023650  012760  024030  000004          MOV     #3$,4(R0)               ;SET UP INTR RETURN ADRS
023656  012760  000340  000006          MOV     #340,6(R0)              ;HIGHEST PRIORITY ON INTR
023664  012737  023712  001110          MOV     #1$,$LPERR              ;SET UP SCOPE LOOP ADRS
023672  012701  024236                  MOV     #BSTRT,R1               ;R1 CONTAINS ADRS PTR OF DISPLAY STARTS
023676  012702  024230                  MOV     #SILOR1,R2              ;R2 CONTAINS ADRS PTR OF EXPECTED SILO DATA
023702  012703  024250                  MOV     #RCNT1,R3               ;R3 CONTAINS ADRS PTR OF # OF RESUMES FOR EA SILO LEVEL
023706  005037  001164                  CLR     $TMP0                   ;$TMP0 CONTAINS SILO LEVEL UNDER TEST
023712  012777  020040  156356  1$:     MOV     #20040,@STKPT           ;SET MAINT SW 2 & TOP OF STACK
023720  012737  000340  177776          MOV     #340,PSW                ;WANT PRIORITY AT TOP
023726  011177  156312                  MOV     (R1),@DPC               ;START
023732  011300                          MOV     (R3),R0                 ;SET UP RESUME CNTR
023734  005277  156304          2$:     INC     @DPC                    ;RESUME
023740  004537  024426                  JSR     R5,DELAY                ;STALL
023744  000004                          BIT2                            ;COUNT TO 4
023746  005300                          DEC     R0                      ;COUNT RESUME
023750  001371                          BNE     2$                      ;BR IF COUNT NOT TO ZERO
023752  052777  040000  156316          BIS     #40000,@STKPT           ;SET MAINT SW 3
023760  052777  000400  156300          BIS     #400,@CONS              ;SET L.P. FLAG 01
023766  005037  177776                  CLR     PSW                     ;ALLOW INTR
023772  021616                          CMP     (SP),(SP)               ;LET CPU GRANT REQ
023774  021616                          CMP     (SP),(SP)               ;
023776  021616                          CMP     (SP),(SP)               ;
024000  021616                          CMP     (SP),(SP)               ;
024002  013737  002266  001122          MOV     CONS,$BDADR             ;SET UP ADRS REG 22
024010  012737  101440  001124          MOV     #101440,$GDDAT          ;EXPECT L.P. FLAG 01,INTS,L.P. EN
024016  017737  156244  001126          MOV     @CONS,$BDDAT            ;READ REG 22
024024  104001                          ERROR   1                       ;L.P. FLAG 01 FAILED TO INTR USING MAINT MODE 3
024026  000474                          BR      7$                      ;NEXT TEST ON INTR FAILURE
024030  022626                  3$:     CMP     (SP)+,(SP)+             ;FIX STACK SINCE NO RTI
024032  017737  156206  024226          MOV     @DPC,DPCSAV             ;GET DPC ADRS
024040  012777  163000  000160          MOV     #163000,@DPCSAV         ;SET UP JSR REL AT SILO RESTORED DPC
024046  013777  024226  156170          MOV     DPCSAV,@DPC             ;SHOULD DO A JSR NOW(RESTORED SILO DATA TO STACK)
024054  004537  024426                  JSR     R5,DELAY                ;STALL FOR STACKING
024060  000001                          BIT0                            ;COUNT TO 1
024062  012777  164000  000136          MOV     #164000,@DPCSAV         ;RESTORE NOP INSTR FOR NEXT STARTS
024070  013737  002272  001122          MOV     STKVAL,$BDADR           ;SET UP REG ADRS 26
```

```
024076  052777  000037  156172          BIS     #37,@STKPT       ;DATA IS IN BYTE 3 OF STACK LEVEL 1
024104  017737  156162  001126          MOV     @STKVAL,$BDDAT   ;READ SAVE STATUS FROM STACK
024112  011237  001124                  MOV     (R2),$GDDAT      ;GET EXPECTED STATUS
024116  023737  001124  001126          CMP     $GDDAT,$BDDAT    ;CORRECT?
024124  001401                          BEQ     4$               ;BR IF SO
024126  104002                          ERROR   2                ;SILO FAILED AT LEVEL INDICATED - DATA VERIFIED FROM STK
024130  062702  000002          4$:     ADD     #2,R2            ;BUMP ADRS PTR OF EXPECTED SILO RESULTS
024134  005712                          TST     (R2)             ;AT END OR RESULT TABLE?
024136  001002                          BNE     5$               ;BR IF NOT
024140  012702  024230                  MOV     #SILOR1,R2       ;RESET ADRS PTR OF EXPECTED RESULTS
024144  062703  000002          5$:     ADD     #2,R3            ;POINT TO NEXT RESUME CNT FOR NEXT SILO LEVEL
024150  005237  001164                  INC     $TMPO            ;ADVANCE TO NEXT SILO LEVEL
024154  022737  000004  001164          CMP     #4,$TMPO         ;ALL 4 LEVELS TESTED AT THIS STARTING ADRS?
024162  001253                          BNE     1$               ;BR IF NOT - REPEAT TEST CHECKING NEXT SILO LEVEL
024164  005037  001164                  CLR     $TMPO            ;RESET SILO LEVEL FOR TESTING WITH DIFF DAT
024170  012703  024250                  MOV     #RCNT1,R3        ;RESET RESUME CNT ADRS PTR
024174  062702  000002                  ADD     #2,R2            ;NEW RESULT ADRS FOR NEXT BUF STARTING ADRS
024200  005712                          TST     (R2)             ;END OF TABLE?
024202  001002                          BNE     6$               ;BR IF NOT
024204  012702  024230                  MOV     #SILOR1,R2       ;RESET RESULT ADRS
024210  062701  000002          6$:     ADD     #2,R1            ;BUMP BUFFER STARTING ADRS PTR - THIS
                                                                 ;CAUSES DIFFERNT DATA PATTERNS THRU SILO
024214  005711                          TST     (R1)             ;ALL 4 BUFFER ADRS STARTS BEEN DONE?
024216  001235                          BNE     1$               ;BR IF NOT
024220  004737  024626          7$:     JSR     PC,RSTVEC        ;GO RESTORE VECTOR WITH HALT
024224  000416                          BR      TST112           ;;GO TO NEXT TEST

024226  000000                  DPCSAV: 0               ;THIS LOCATION SAVES THE DPC THAT WAS RESTORED FROM THE SILO

024230  053465                  SILOR1: 53465           ;SHIFT OUT,CHAR STRING ESC,-EDGE FLG INTR EN,QUE,QUE CNTRL
                                                        ;L.P. INTR EN 01,INT 00&01,-L.P. HIT DIS,STOP INTR EN
024232  005066                          5066            ;-SHIFT OUT,-CHAR STRING ESC,EDGE FLG INTR EN,-QUE,-QUE CNTRL,
                                                        ;L.P. INTR EN 01,INT 00&01,L.P. HIT DIS,-STOP INTR EN
024234  000000                          0               ;TERMINATOR

024236  032342                  BSTRT:  BUFFER          ;THIS TABLE CONTAINS DIFFERENT STARTING ADRS FOR TESTING
                                                        ;EACH SILO LEVEL
024240  032360                          BUFFER+16
024242  032376                          BUFFER+34
024244  032414                          BUFFER+52
024246  000000                          0

024250  000011                  RCNT1:  11              ;THIS TABLE CONTROLS THE  AMOUNT OF RESUMES NEEDED TO
024252  000020                          20              ;CYCLE THRU ALL FOUR LEVELS OF SILO
024254  000027                          27
024256  000036                          36
024260  000000                          0

                                ;;*****************************************************************
                                ;*TEST 112      TEST FOR HALT AT END OF DIAGNOSTIC (SW12 SET)
                                ;;*****************************************************************
024262  000004                  TST112: SCOPE
024264  012737  000112  001206          MOV     #112,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
024272  032777  010000  154640          BIT     #BIT12,@SWR      ;IS SR12 SET?
024300  001401                          BEQ     $EOP             ;BR IF NOT
024302  000000                          HALT                     ;COMPLETED PASS - SW12 SAYS HALT
```

```
                                        .SBTTL  END OF PASS ROUTINE

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

024304                                  $EOP:
024304  000004                                  SCOPE
024306  005037  001102                          CLR     $TSTNM          ;;ZERO THE TEST NUMBER
024312  005037  001166                          CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
024316  005237  001210                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
024322  042737  100000  001210                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
024330  005327                                  DEC     (PC)+           ;;LOOP?
024332  000001                          $EOPCT: .WORD   1
024334  003022                                  BGT     $DOAGN          ;;YES
024336  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
024340  000001                          $ENDCT: .WORD   1
024342  024332                                  $EOPCT
024344  104400  024411                          TYPE    ,$ENDMG         ;;TYPE "END PASS #"
024350  013746  001210                          MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
024354  104404                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
024356  104400  024406                          TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
024362  013700  000042                  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
024366  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
024370  000005                                  RESET                   ;;CLEAR THE WORLD
024372  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
024374  000240                                  NOP                     ;;SAVE ROOM
024376  000240                                  NOP                     ;;FOR
024400  000240                                  NOP                     ;;ACT11
024402                                  $DOAGN:
024402  000137                                  JMP     @(PC)+          ;;RETURN
024404  003244                          $RTNAD: .WORD   RSTART
024406     377     377     000          $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
024411     015  042412  042116          $ENDMG: .ASCIZ  <15><12>/END PASS #/
024416  050040  051501  020123
024424  000043
```

```
                                       ;;*************************************************************
                                       ;THIS ROUTINE WILL STALL FOR THE NO OF COUNTS IN THE ADRS OF R5
                                       ;;*************************************************************
024426  012537  002206                 DELAY:  MOV     (R5)+,DLYCNT    ;GET DELAY COUNT
024432  005337  002206                 1$:     DEC     DLYCNT          ;START COUNTING
024436  001375                                 BNE     1$              ;BR IF MORE COUNTS
024440  000205                                 RTS     R5              ;EXIT DELAY ROUTINE

                                       ;;*************************************************************
                                       ;THIS ROUTINE SETS UP JSR, POP & TEST INSTR FOR POP-RESTORE TESTS
                                       ;;*************************************************************
024442  012777  020040  155626         POPSET: MOV     #20040,@STKPT   ;SET TOP OF STK
024450  012737  163000  032344                 MOV     #163000,BUFFER+2        ;SET UP JSR REL INSTR
024456  012737  166000  032350                 MOV     #166000,BUFFER+6        ;SET UP POP-RESTORE INSTR
024464  012437  032346                         MOV     (R4)+,BUFFER+4  ;SET UP INSTRUCTION UNDER TEST
024470  000204                                 RTS     R4              ;RETURN

                                       ;;*************************************************************
                                       ;THIS ROUTINE CONTROLS JSR, POP & TEST INSTR SEQUENCE FOR POP-RESTORE TESTS
                                       ;;*************************************************************
024472  012777  020040  155576         DO:     MOV     #20040,@STKPT   ;SET TOP OF STK
024500  012777  032342  155536                 MOV     #BUFFER,@DPC    ;START -SET BITS UNDER TEST
024506  052777  020000  155562                 BIS     #20000,@STKPT   ;FUMBLE
024514  005277  155524                         INC     @DPC            ;ADVANCE TO JSR INSTR
024520  004537  024426                         JSR     R5,DELAY        ;STALL FOR STACKING
024524  000004                                 BIT2                    ;COUNT TO 4
024526  012777  032346  155510                 MOV     #BUFFER+4,@DPC  ;START -RESET BITS UNDER TEST
024534  052777  020000  155534                 BIS     #20000,@STKPT   ;FUMBLE
024542  005277  155476                         INC     @DPC            ;ADVANCE TO POP INSTR
024546  004537  024426                         JSR     R5,DELAY        ;STALL FOR RESTORING
024552  000004                                 BIT2                    ;COUNT TO 4
024554  000207                                 RTS     PC              ;RETURN

                                       ;;*************************************************************
                                       ;THIS ROUTINE CONVERTS A KT11 PAGE ADRS TO A 18 BIT ADRS
                                       ;;*************************************************************
024556  005001                         DPCONV: CLR     R1              ;CLR HI ORDER ADRS
024560  006300                                 ASL     R0              ;START SHIFTING LEFT
024562  006300                                 ASL     R0              ;
024564  006300                                 ASL     R0              ;
024566  006300                                 ASL     R0              ;
024570  100002                         DPCON1: BPL     1$              ;BR IF ADRS BIT 17 IS CLR
024572  012701  000002                         MOV     #2,R1           ;SET ADRS BIT 17 INDICATION IN BIT 1 - R1
024576  006300                         1$:     ASL     R0              ;LOOK AT ADRS BIT 16
024600  100001                                 BPL     2$              ;BR IF ADRS BIT 16 IS CLR
024602  005201                                 INC     R1              ;SET ADRS BIT 16 INDICATION IN BIT 0 - R1
024604  006300                         2$:     ASL     R0              ;COMPLETE LEFT JUSTIFICATION
024606  000207                                 RTS     PC              ;RETURN
```

```
                                        ;;**********************************************************
                                                ;THIS CODE REPORTS ANY KT11 ERROR AND HALTS
                                        ;;**********************************************************
024610  042737  001400  177572  KTSER:  BIC     #1400,@#KTSR0   ;DISABLE KT11
024616  104400  027457                  TYPE    ,MSG2           ;GO REPORT KT11 THAT A ERROR TRAP OCCURRED
024622  000000                  1$:     HALT                    ;KT11 ERROR TRAP - RUN KT11 DIAG OR
024624  000776                          BR      1$              ;RESTART

                                        ;;**********************************************************
                                                ;THIS CODE RESETS ALL VS60 VECTORS WITH HALTS
                                        ;;**********************************************************
024626  012737  000340  177776  RSTVEC: MOV     #340,PSW        ;RESET PRIORITY TO HIGHEST LEVEL
024634  012701  000010                  MOV     #10,R1          ;SET UP VECTOR LOCATION COUNT
024640  013700  001252                  MOV     $VECT1,R0       ;GET 1ST VS60 VECTOR ADRS
024644  010010                  1$:     MOV     R0,(R0)         ;SET UP ADRS TO HALT
024646  062720  000002                  ADD     #2,(R0)+        ;POINT IT TO HALT
024652  005020                          CLR     (R0)+           ;SET UP HALT
024654  005301                          DEC     R1              ;COUNT THIS VECTOR RESTORE
024656  001372                          BNE     1$              ;BR IF MORE TO RESTORE
024660  000207                          RTS     PC              ;RETURN IF ALL DONE

                                        ;;**********************************************************
                                                ;THIS ROUTINE MAKES UP A 12 BIT TANGENT FROM THE X OR Y POS
                                                ;REGS AND THE X OR Y DYNAMIC OFFSET REGS (HI ORDER 2 BITS) AND
                                                ;LEAVES THE RESULT IN $BDDAT
                                        ;;**********************************************************
024662  042737  176000  001126  TANCON: BIC     #176000,$BDDAT  ;SAVE LOW ORDER BITS ONLY
024670  042705  147777                  BIC     #147777,R5      ;ONLY WANT BITS 10 & 11
024674  006205                          ASR     R5              ;ROTATE 2 RIGHT
024676  006205                          ASR     R5              ;
024700  060537  001126                  ADD     R5,$BDDAT       ;MAKE UP 12 BIT TANGENT
024704  000207                          RTS     PC              ;RETURN
```

```
                                        ;;***************************************************************
                                                ;THIS ROUTINE UPDATES THE TANGENT IN & GDDAT FOR EVERY INCREASE
                                                ; IN VECTOR OF X OR Y
                                        ;;***************************************************************
024706  001010                          CALTAN: BNE     1$                      ;BR AND SET UP TANGENT IF VECTOR NON-ZERO
024710  012703  000400                          MOV     #400,R3                 ;R3 COUNTS EVERY 256+4 ADDS TO TANGENT
024714  012737  000001  001124                  MOV     #1,$GDDAT               ;TANGENT WILL BE 1 WHEN VECTOR LENGTH 0
024722  005037  025016                          CLR     TANGNT                  ;TANGENT HOLDING LOC
024726  000432                                  BR      3$                      ;RETURN WITH ZERO TANGENT
024730  013737  025016  001124          1$:     MOV     TANGNT,$GDDAT           ;GET LAST VALUE
024736  062737  000005  001124                  ADD     #5,$GDDAT               ;ADD 4 AND 1 FOR FRACTIONAL ROUNDING IN HARDWARE
024744  062737  000004  025016                  ADD     #4,TANGNT               ;ADD 4 ON EVERY COUNT OF VECTOR LENGTH
024752  005303                                  DEC     R3                      ;ROUND UP EVERY 11.25 DEGREES
024754  001010                                  BNE     2$                      ;BR IF NOT REQUIRED
024756  062737  000001  001124                  ADD     #1,$GDDAT               ;ROUND UP EVERY 256
024764  062737  000001  025016                  ADD     #1,TANGNT               ;SAME
024772  012703  000400                          MOV     #400,R3                 ;RESET 256 COUNT
024776  032737  010000  001124          2$:     BIT     #10000,$GDDAT           ;TEST TANGENT FOR GREATER THAN 12 BITS
025004  001403                                  BEQ     3$                      ;BR IF NOT
025006  012737  007777  001124                  MOV     #7777,$GDDAT            ;CAN'T GET GREATER THAN 7777
025014  000207                          3$:     RTS     PC                      ;RETURN WITH TANGENT IN $GDDAT

025016  000000                          TANGNT: 0                               ;LOC MAINTAINS TANGENT MINUS 1

                                        ;;***************************************************************
                                                ;THIS ROUTINE STARTS THE DISPLAY WITH MAINT. SW 2, THEN SETS
                                                ;MAINT. SW 3, AND THEN ISSUES A # OF RESUMES SPECIFIED
                                                ;BY R5 AT WHICH TIME IT WILL EXIT
                                        ;;***************************************************************
025020                                  EXECUTE:
025020  012537  025070                          MOV     (R5)+,CNTR              ;SET UP THE RESUME COUNTER
025024  012777  020040  155244                  MOV     #20040,@STKPT           ;SET MAINT SW 2 AND TOP OF STACK
025032  012777  032342  155204                  MOV     #BUFFER,@DPC            ;START DISPLAY
025040  052777  040000  155230          1$:     BIS     #40000,@STKPT           ;SET MAINT SW 3
025046  052777  040000  155222                  BIS     #40000,@STKPT           ;AGAIN
025054  005277  155164                          INC     @DPC                    ;RESUME
025060  005337  025070                          DEC     CNTR                    ;COUNT RESUME
025064  001365                                  BNE     1$                      ;BR IF MORE RESUMES TO FOLLOW
025066  000205                                  RTS     R5                      ;EXIT

025070  000000                          CNTR:   0                               ;CNTR USED ABOVE
```

```
                                        .SBTTL  SCOPE HANDLER ROUTINE

                                        ;;*************************************************************
                                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                        ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW14=1         LOOP ON TEST
                                        ;*SW11=1         INHIBIT ITERATIONS
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*SW08=1         LOOP ON TEST IN SWR<7:0>
                                        ;*CALL
                                        ;*       SCOPE           ;;SCOPE=IOT

 025072                                 $SCOPE:
 025072  032777  040000  154040         1$:     BIT     #BIT14,@SWR             ;;LOOP ON PRESENT TEST?
 025100  001114                                 BNE     $OVER                   ;;YES IF SW14=1
                                        ;#####START OF CODE FOR THE XOR TESTER#####
 025102  000416                         $XTSTR: BR      6$                      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
 025104  013746  000004                         MOV     @#ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 025110  012737  025130  000004                 MOV     #5$,@#ERRVEC            ;;SET FOR TIMEOUT
 025116  005737  177060                         TST     @#177060                ;;TIME OUT ON XOR?
 025122  012637  000004                         MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
 025126  000463                                 BR      $SVLAD                  ;;GO TO THE NEXT TEST
 025130  022626                         5$:     CMP     (SP)+,(SP)+             ;;CLEAR THE STACK AFTER A TIME OUT
 025132  012637  000004                         MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
 025136  000423                                 BR      7$                      ;;LOOP ON THE PRESENT TEST
 025140                                 6$:;#####END OF CODE FOR THE XOR TESTER#####
 025140  032777  000400  153772                 BIT     #BIT08,@SWR             ;;LOOP ON SPEC. TEST?
 025146  001404                                 BEQ     2$                      ;;BR IF NO
 025150  127737  153764  001102                 CMPB    @SWR,$TSTNM             ;;ON THE RIGHT TEST?    SWR<7:0>
 025156  001465                                 BEQ     $OVER                   ;;BR IF YES
 025160  105737  001103                 2$:     TSTB    $ERFLG                  ;;HAS AN ERROR OCCURRED?
 025164  001421                                 BEQ     3$                      ;;BR IF NO
 025166  123737  001115  001103                 CMPB    $ERMAX,$ERFLG           ;;MAX. ERRORS FOR THIS TEST OCCURRED?
 025174  101015                                 BHI     3$                      ;;BR IF NO
 025176  032777  001000  153734                 BIT     #BIT09,@SWR             ;;LOOP ON ERROR?
 025204  001404                                 BEQ     4$                      ;;BR IF NO
 025206  013737  001110  001106         7$:     MOV     $LPERR,$LPADR           ;;SET LOOP ADDRESS TO LAST SCOPE
 025214  000446                                 BR      $OVER
 025216  105037  001103                 4$:     CLRB    $ERFLG                  ;;ZERO THE ERROR FLAG
 025222  005037  001166                         CLR     $TIMES                  ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
 025226  000415                                 BR      1$                      ;;ESCAPE TO THE NEXT TEST
 025230  032777  004000  153702         3$:     BIT     #BIT11,@SWR             ;;INHIBIT ITERATIONS?
 025236  001011                                 BNE     1$                      ;;BR IF YES
 025240  005737  001210                         TST     $PASS                   ;;IF FIRST PASS OF PROGRAM
 025244  001406                                 BEQ     1$                      ;;      INHIBIT ITERATIONS
 025246  005237  001104                         INC     $ICNT                   ;;INCREMENT ITERATION COUNT
 025252  023737  001166  001104                 CMP     $TIMES,$ICNT            ;;CHECK THE NUMBER OF ITERATIONS MADE
 025260  002024                                 BGE     $OVER                   ;;BR IF MORE ITERATION REQUIRED
 025262  012737  000001  001104         1$:     MOV     #1,$ICNT                ;;REINITIALIZE THE ITERATION COUNTER
 025270  013737  025346  001166                 MOV     $MXCNT,$TIMES           ;;SET NUMBER OF ITERATIONS TO DO
 025276  105237  001102                 $SVLAD: INCB    $TSTNM                  ;;COUNT TEST NUMBERS
 025302  113737  001102  001206                 MOVB    $TSTNM,$TESTN           ;;SET TEST NUMBER IN APT MAILBOX
 025310  011637  001106                         MOV     (SP),$LPADR             ;;SAVE SCOPE LOOP ADDRESS
```

```
 3804  025314  011637  001110                    MOV     (SP),$LPERR        ;;SAVE ERROR LOOP ADDRESS
 3805  025320  005037  001170                    CLR     $ESCAPE            ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 3806  025324  112737  000001  001115            MOVB    #1,$ERMAX          ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 3807  025332  013777  001102  153602  $OVER:    MOV     $TSTNM,@DISPLAY    ;;DISPLAY TEST NUMBER
 3808  025340  013716  001106            MOV     $LPADR,(SP)        ;;FUDGE RETURN ADDRESS
 3809  025344  000002                            RTI                        ;;FIXES PS
 3810  025346  003720                    $MXCNT: 2000.                      ;;MAX. NUMBER OF ITERATIONS
```

```
                                        .SBTTL  APT COMMUNICATIONS ROUTINE

                                        ;;*************************************************************
025350  112737  000001  025614  $ATY1:  MOVB    #1,$FFLG                ;;TO REPORT FATAL ERROR
025356  112737  000001  025612  $ATY3:  MOVB    #1,$MFLG                ;;TO TYPE A MESSAGE
025364  000403                          BR      $ATYC
025366  112737  000001  025614  $ATY4:  MOVB    #1,$FFLG                ;;TO ONLY REPORT FATAL ERROR
025374                          $ATYC:
025374  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
025376  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
025400  105737  025612                  TSTB    $MFLG                   ;;SHOULD TYPE A MESSAGE?
025404  001450                          BEQ     5$                      ;;IF NOT:  BR
025406  122737  000001  001222          CMPB    #APTENV,$ENV            ;;OPERATING UNDER APT?
025414  001031                          BNE     3$                      ;;IF NOT:  BR
025416  132737  000100  001223          BITB    #APTSPOOL,$ENVM         ;;SHOULD SPOOL MESSAGES?
025424  001425                          BEQ     3$                      ;;IF NOT:  BR
025426  017600  000004                  MOV     @4(SP),R0               ;;GET MESSAGE ADDR.
025432  062766  000002  000004          ADD     #2,4(SP)                        ;;BUMP RETURN ADDR.
025440  005737  001202          1$:     TST     $MSGTYPE                ;;SEE IF DONE W/ LAST XMISSION?
025444  001375                          BNE     1$                      ;;IF NOT:  WAIT
025446  010037  001216                  MOV     R0,$MSGAD               ;;PUT ADDR IN MAILBOX
025452  105720                  2$:     TSTB    (R0)+                   ;;FIND END OF MESSAGE
025454  001376                          BNE     2$
025456  163700  001216                  SUB     $MSGAD,R0               ;;SUB START OF MESSAGE
025462  006200                          ASR     R0                      ;;GET MESSAGE LNGTH IN WORDS
025464  010037  001220                  MOV     R0,$MSGLGT              ;;PUT LENGTH IN MAILBOX
025470  012737  000004  001202          MOV     #4,$MSGTYPE             ;;TELL APT TO TAKE MSG.
025476  000413                          BR      5$
025500  017637  000004  025524  3$:     MOV     @4(SP),4$               ;;PUT MSG ADDR IN JSR LINKAGE
025506  062766  000002  000004          ADD     #2,4(SP)                        ;;BUMP RETURN ADDRESS
025514  013746  177776                  MOV     177776,-(SP)            ;;PUSH 177776 ON STACK
025520  004737  026154                  JSR     PC,$TYPE                ;;CALL TYPE MACRO
025524  000000                  4$:     .WORD   0
025526                          5$:
025526  105737  025614          10$:    TSTB    $FFLG                   ;;SHOULD REPORT FATAL ERROR?
025532  001416                          BEQ     12$                     ;;IF NOT:  BR
025534  005737  001222                  TST     $ENV                    ;;RUNNING UNDER APT?
025540  001413                          BEQ     12$                     ;;IF NOT:  BR
025542  005737  001202          11$:    TST     $MSGTYPE                ;;FINISHED LAST MESSAGE?
025546  001375                          BNE     11$                     ;;IF NOT:  WAIT
025550  017637  000004  001204          MOV     @4(SP),$FATAL           ;;GET ERROR #
025556  062766  000002  000004          ADD     #2,4(SP)                        ;;BUMP RETURN ADDR.
025564  005237  001202                  INC     $MSGTYPE                ;;TELL APT TO TAKE ERROR
025570  105037  025614          12$:    CLRB    $FFLG                   ;;CLEAR FATAL FLAG
025574  105037  025613                  CLRB    $LFLG                   ;;CLEAR LOG FLAG
025600  105037  025612                  CLRB    $MFLG                   ;;CLEAR MESSAGE FLAG
025604  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
025606  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
025610  000207                          RTS     PC                      ;;RETURN
025612     000                  $MFLG:  .BYTE   0                       ;;MESSG. FLAG
025613     000                  $LFLG:  .BYTE   0                       ;;LOG FLAG
025614     000                  $FFLG:  .BYTE   0                       ;;FATAL FLAG
        025616                          .EVEN
        000200                  APTSIZE=200
        000001                  APTENV=001
        000100                  APTSPOOL=100
```

```
                000040                          APTCSUP=040
                                                .SBTTL  ERROR HANDLER ROUTINE

                                                ;;************************************************************
                                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                                ;*AND GO TO $ERRTYP ON ERROR
                                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                                ;*SW15=1         HALT ON ERROR
                                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                                ;*SW10=1         BELL ON ERROR
                                                ;*SW09=1         LOOP ON ERROR
                                                ;*CALL
                                                ;*       ERROR   N         ;;ERROR=EMT AND N=ERROR ITEM NUMBER

        025616                                  $ERROR:
        025616  113737  001102  002204                  MOVB    $TSTNM,TSTNUM
        025624  105237  001103                  7$:     INCB    $ERFLG                  ;;SET THE ERROR FLAG
        025630  001775                                  BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
        025632  013777  001102  153302                  MOV     $TSTNM,@DISPLAY         ;;DISPLAY TEST NUMBER AND ERROR FLAG
        025640  032777  002000  153272                  BIT     #BIT10,@SWR             ;;BELL ON ERROR?
        025646  001402                                  BEQ     1$                      ;;NO - SKIP
        025650  104400  001172                          TYPE    ,$BELL                  ;;RING BELL
        025654  005237  001112                  1$:     INC     $ERTTL                  ;;COUNT THE NUMBER OF ERRORS
        025660  011637  001116                          MOV     (SP),$ERRPC             ;;GET ADDRESS OF ERROR INSTRUCTION
        025664  162737  000002  001116                  SUB     #2,$ERRPC
        025672  117737  153220  001114                  MOVB    @$ERRPC,$ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
        025700  032777  020000  153232                  BIT     #BIT13,@SWR             ;;SKIP TYPEOUT IF SET
        025706  001004                                  BNE     20$                     ;;SKIP TYPEOUTS
        025710  004737  026020                          JSR     PC,$ERRTYP              ;;GO TO USER ERROR ROUTINE
        025714  104400  001177                          TYPE    ,$CRLF
        025720                                  20$:
        025720  122737  000001  001222                  CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
        025726  001007                                  BNE     2$                      ;;NO,SKIP APT ERROR REPORT
        025730  113737  001114  025742                  MOVB    $ITEMB,21$              ;;SET ITEM NUMBER AS ERROR NUMBER
        025736  004737  025366                          JSR     PC,$ATY4                ;;REPORT FATAL ERROR TO APT
        025742     000                          21$:    .BYTE   0
        025743     000                                  .BYTE   0
        025744  000777                          22$:    BR      22$                     ;;APT ERROR LOOP
        025746  005777  153166                  2$:     TST     @SWR                    ;;HALT ON ERROR
        025752  100001                                  BPL     3$                      ;;SKIP IF CONTINUE
        025754  000000                                  HALT                            ;;HALT ON ERROR!
        025756  032777  001000  153154          3$:     BIT     #BIT09,@SWR             ;;LOOP ON ERROR SWITCH SET?
        025764  001402                                  BEQ     4$                      ;;BR IF NO
        025766  013716  001110                          MOV     $LPERR,(SP)             ;;FUDGE RETURN FOR LOOPING
        025772  005737  001170                  4$:     TST     $ESCAPE                 ;;CHECK FOR AN ESCAPE ADDRESS
        025776  001402                                  BEQ     5$                      ;;BR IF NONE
        026000  013716  001170                          MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
        026004                                  5$:
        026004  022737  024372  000042                  CMP     #$ENDAD,@#42            ;;ACT-11 AUTO-ACCEPT?
        026012  001001                                  BNE     6$                      ;;BRANCH IF NO
        026014  000000                                  HALT                            ;;YES
        026016                                  6$:
        026016  000002                                  RTI                             ;;RETURN
```

```
                                     .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

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

026020                               $ERRTYP:
026020  104400  001177                       TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
026024  010046                               MOV     R0,-(SP)                ;;SAVE R0
026026  005000                               CLR     R0                      ;;PICKUP THE ITEM INDEX
026030  153700  001114                       BISB    @#$ITEMB,R0
026034  001004                               BNE     1$                      ;;IF ITEM NUMBER IS ZERO, JUST
                                                                             ;;TYPE THE PC OF THE ERROR
026036  013746  001116                       MOV     $ERRPC,-(SP)            ;;SAVE $ERRPC FOR TYPEOUT
                                                                             ;;ERROR ADDRESS
026042  104401                               TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
026044  000426                               BR      6$                      ;;GET OUT
026046  005300                       1$:     DEC     R0                      ;;ADJUST THE INDEX SO THAT IT WILL
026050  006300                               ASL     R0                      ;;       WORK FOR THE ERROR TABLE
026052  006300                               ASL     R0
026054  006300                               ASL     R0
026056  062700  001264                       ADD     #$ERRTB,R0              ;;FORM TABLE POINTER
026062  012037  026072                       MOV     (R0)+,2$                ;;PICKUP "ERROR MESSAGE" POINTER
026066  001404                               BEQ     3$                      ;;SKIP TYPEOUT IF NO POINTER
026070  104400                               TYPE                            ;;TYPE THE "ERROR MESSAGE"
026072  000000                       2$:     .WORD   0                       ;;"ERROR MESSAGE" POINTER GOES HERE
026074  104400  001177                       TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
026100  012037  026110               3$:     MOV     (R0)+,4$                ;;PICKUP "DATA HEADER" POINTER
026104  001404                               BEQ     5$                      ;;SKIP TYPEOUT IF 0
026106  104400                               TYPE                            ;;TYPE THE "DATA HEADER"
026110  000000                       4$:     .WORD   0                       ;;"DATA HEADER" POINTER GOES HERE
026112  104400  001177                       TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
026116  011000                       5$:     MOV     (R0),R0                 ;;PICKUP "DATA TABLE" POINTER
026120  001004                               BNE     7$                      ;;GO TYPE THE DATA
026122  012600                       6$:     MOV     (SP)+,R0                ;;RESTORE R0
026124  104400  001177                       TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
026130  000207                               RTS     PC                      ;;RETURN
026132                               7$:
026132  013046                               MOV     @(R0)+,-(SP)            ;;SAVE @(R0)+ FOR TYPEOUT
026134  104401                               TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
026136  005710                               TST     (R0)                    ;;IS THERE ANOTHER NUMBER?
026140  001770                               BEQ     6$                      ;;BR IF NO
026142  104400  026150                       TYPE    ,8$                     ;;TYPE TWO(2) SPACES
026146  000771                               BR      7$                      ;;LOOP
026150  020040     000               8$:     .ASCIZ  /  /                    ;;TWO(2) SPACES
        026154                               .EVEN
```

```
                                        .SBTTL  TYPE ROUTINE

                                        ;;*************************************************************
                                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                        ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                        ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                        ;*
                                        ;*CALL:
                                        ;*1) USING A TRAP INSTRUCTION
                                        ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                        ;*OR
                                        ;*      TYPE
                                        ;*      MESADR
                                        ;*

026154  105737  001157          $TYPE:  TSTB    $TPFLG                  ;;IS THERE A TERMINAL?
026160  100002                          BPL     1$                      ;;BR IF YES
026162  000000                          HALT                            ;;HALT HERE IF NO TERMINAL
026164  000430                          BR      3$                      ;;LEAVE
026166  010046                  1$:     MOV     R0,-(SP)                ;;SAVE R0
026170  017600  000002                  MOV     @2(SP),R0               ;;GET ADDRESS OF ASCIZ STRING
026174  122737  000001  001222          CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
026202  001011                          BNE     62$                     ;;NO,GO CHECK FOR APT CONSOLE
026204  132737  000100  001223          BITB    #APTSPOOL,$ENVM         ;;SPOOL MESSAGE TO APT
026212  001405                          BEQ     62$                     ;;NO,GO CHECK FOR CONSOLE
026214  010037  026224                  MOV     R0,61$                  ;;SETUP MESSAGE ADDRESS FOR APT
026220  004737  025356                  JSR     PC,$ATY3                ;;SPOOL MESSAGE TO APT
026224  000000                  61$:    .WORD   0                       ;;MESSAGE ADDRESS
026226  132737  000040  001223  62$:    BITB    #APTCSUP,$ENVM          ;;APT CONSOLE SUPPRESSED
026234  001003                          BNE     60$                     ;;YES,SKIP TYPE OUT
026236  112046                  2$:     MOVB    (R0)+,-(SP)             ;;PUSH CHARACTER TO BE TYPED ONTO STACK
026240  001005                          BNE     4$                      ;;BR IF IT ISN'T THE TERMINATOR
026242  005726                          TST     (SP)+                   ;;IF TERMINATOR POP IT OFF THE STACK
026244  012600                  60$:    MOV     (SP)+,R0                ;;RESTORE R0
026246  062716  000002          3$:     ADD     #2,(SP)                 ;;ADJUST RETURN PC
026252  000002                          RTI                             ;;RETURN
026254  122716  000011          4$:     CMPB    #HT,(SP)                ;;BRANCH IF <HT>
026260  001430                          BEQ     8$
026262  122716  000200                  CMPB    #CRLF,(SP)              ;;BRANCH IF NOT <CRLF>
026266  001006                          BNE     5$
026270  005726                          TST     (SP)+                   ;;POP  <CR><LF> EQUIV
026272  104400                          TYPE                            ;;TYPE A CR AND LF
026274  001177                          $CRLF
026276  105037  026432                  CLRB    $CHARCNT                ;;CLEAR CHARACTER COUNT
026302  000755                          BR      2$                      ;;GET NEXT CHARACTER
026304  004737  026366          5$:     JSR     PC,$TYPEC               ;;GO TYPE THIS CHARACTER
026310  123726  001156          6$:     CMPB    $FILLC,(SP)+            ;;IS IT TIME FOR FILLER CHARS.?
026314  001350                          BNE     2$                      ;;IF NO GO GET NEXT CHAR.
026316  013746  001154                  MOV     $NULL,-(SP)             ;;GET # OF FILLER CHARS. NEEDED
                                                                        ;;AND THE NULL CHAR.
026322  105366  000001          7$:     DECB    1(SP)                   ;;DOES A NULL NEED TO BE TYPED?
026326  002770                          BLT     6$                      ;;BR IF NO--GO POP THE NULL OFF OF STACK
026330  004737  026366                  JSR     PC,$TYPEC               ;;GO TYPE A NULL
026334  105337  026432                  DECB    $CHARCNT                ;;DO NOT COUNT AS A COUNT
```

```
026340  000770                          BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

026342  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
026346  004737  026366          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
026352  132737  000007  026432          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
026360  001372                          BNE     9$              ;;TAB STOP
026362  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
026364  000724                          BR      2$              ;;GET NEXT CHARACTER
026366  105777  152556          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
026372  100375                          BPL     $TYPEC
026374  116677  000002  152550          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
026402  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
026410  001003                          BNE     1$              ;;BRANCH IF NO
026412  105037  026432                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
026416  000406                          BR      $TYPEX          ;;EXIT
026420  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
026426  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
026430  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
026432  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
026434  000207                  $TYPEX: RTS     PC
```

```
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                        ;;**************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)               ;;NUMBER TO BE TYPED
                                        ;*      TYPOS                           ;;CALL FOR TYPEOUT
                                        ;*      .BYTE   N                       ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                        ;*      .BYTE   M                       ;;M=1 OR 0
                                        ;*                                              ;;1=TYPE LEADING ZEROS
                                        ;*                                              ;;0=SUPPRESS LEADING ZEROS
                                        ;*
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)               ;;NUMBER TO BE TYPED
                                        ;*      TYPON                           ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)               ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                           ;;CALL FOR TYPEOUT

026436  017646  000000                  $TYPOS: MOV     @(SP),-(SP)             ;;PICKUP THE MODE
026442  116637  000001  026661                  MOVB    1(SP),$0FILL            ;;LOAD ZERO FILL SWITCH
026450  112637  026663                          MOVB    (SP)+,$0MODE+1          ;;NUMBER OF DIGITS TO TYPE
026454  062716  000002                          ADD     #2,(SP)                 ;;ADJUST RETURN ADDRESS
026460  000406                                  BR      $TYPON
026462  112737  000001  026661          $TYPOC: MOVB    #1,$0FILL               ;;SET THE ZERO FILL SWITCH
026470  112737  000006  026663                  MOVB    #6,$0MODE+1             ;;SET FOR SIX(6) DIGITS
026476  112737  000005  026660          $TYPON: MOVB    #5,$0CNT                ;;SET THE ITERATION COUNT
026504  010346                                  MOV     R3,-(SP)                ;;SAVE R3
026506  010446                                  MOV     R4,-(SP)                ;;SAVE R4
026510  010546                                  MOV     R5,-(SP)                ;;SAVE R5
026512  113704  026663                          MOVB    $0MODE+1,R4             ;;GET THE NUMBER OF DIGITS TO TYPE
026516  005404                                  NEG     R4
026520  062704  000006                          ADD     #6,R4                   ;;SUBTRACT IT FOR MAX. ALLOWED
026524  110437  026662                          MOVB    R4,$0MODE               ;;SAVE IT FOR USE
026530  113704  026661                          MOVB    $0FILL,R4               ;;GET THE ZERO FILL SWITCH
026534  016605  000012                          MOV     12(SP),R5               ;;PICKUP THE INPUT NUMBER
026540  005003                                  CLR     R3                      ;;CLEAR THE OUTPUT WORD
026542  006105                          1$:     ROL     R5                      ;;ROTATE MSB INTO "C"
026544  000404                                  BR      3$                      ;;GO DO MSB
026546  006105                          2$:     ROL     R5                      ;;FORM THIS DIGIT
026550  006105                                  ROL     R5
026552  006105                                  ROL     R5
026554  010503                                  MOV     R5,R3
026556  006103                          3$:     ROL     R3                      ;;GET LSB OF THIS DIGIT
026560  105337  026662                          DECB    $0MODE                  ;;TYPE THIS DIGIT?
026564  100016                                  BPL     7$                      ;;BR IF NO
026566  042703  177770                          BIC     #177770,R3              ;;GET RID OF JUNK
026572  001002                                  BNE     4$                      ;;TEST FOR 0
026574  005704                                  TST     R4                      ;;SUPPRESS THIS 0?
026576  001403                                  BEQ     5$                      ;;BR IF YES
```

```
026600  005204                  4$:      INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
026602  052703  000060                   BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
026606  052703  000040          5$:      BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
026612  110337  026656                   MOVB    R3,8$           ;;SAVE FOR TYPING
026616  104400  026656                   TYPE    ,8$             ;;GO TYPE THIS DIGIT
026622  105337  026660          7$:      DECB    $OCNT           ;;COUNT BY 1
026626  003347                           BGT     2$              ;;BR IF MORE TO DO
026630  002402                           BLT     6$              ;;BR IF DONE
026632  005204                           INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
026634  000744                           BR      2$              ;;GO DO THE LAST DIGIT
026636  012605                  6$:      MOV     (SP)+,R5        ;;RESTORE R5
026640  012604                           MOV     (SP)+,R4        ;;RESTORE R4
026642  012603                           MOV     (SP)+,R3        ;;RESTORE R3
026644  016666  000002  000004           MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
026652  012616                           MOV     (SP)+,(SP)
026654  000002                           RTI                     ;;RETURN
026656     000                  8$:      .BYTE   0               ;;STORAGE FOR ASCII DIGIT
026657     000                           .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
026660     000                  $OCNT:   .BYTE   0               ;;OCTAL DIGIT COUNTER
026661     000                  $OFILL:  .BYTE   0               ;;ZERO FILL SWITCH
026662  000000                  $OMODE:  .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                        .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

 026664                                 $TYPDS:
 026664 010046                                  MOV     R0,-(SP)            ;;PUSH R0 ON STACK
 026666 010146                                  MOV     R1,-(SP)            ;;PUSH R1 ON STACK
 026670 010246                                  MOV     R2,-(SP)            ;;PUSH R2 ON STACK
 026672 010346                                  MOV     R3,-(SP)            ;;PUSH R3 ON STACK
 026674 010546                                  MOV     R5,-(SP)            ;;PUSH R5 ON STACK
 026676 012746 020200                           MOV     #20200,-(SP)        ;;SET BLANK SWITCH AND SIGN
 026702 016605 000020                           MOV     20(SP),R5           ;;GET THE INPUT NUMBER
 026706 100004                                  BPL     1$                  ;;BR IF INPUT IS POS.
 026710 005405                                  NEG     R5                  ;;MAKE THE BINARY NUMBER POS.
 026712 112766 000055 000001                    MOVB    #'-,1(SP)           ;;MAKE THE ASCII NUMBER NEG.
 026720 005000                          1$:     CLR     R0                  ;;ZERO THE CONSTANTS INDEX
 026722 012703 027100                           MOV     #$DBLK,R3           ;;SETUP THE OUTPUT POINTER
 026726 112723 000040                           MOVB    #' ,(R3)+           ;;SET THE FIRST CHARACTER TO A BLANK
 026732 005002                          2$:     CLR     R2                  ;;CLEAR THE BCD NUMBER
 026734 016001 027070                           MOV     $DTBL(R0),R1        ;;GET THE CONSTANT
 026740 160105                          3$:     SUB     R1,R5               ;;FORM THIS BCD DIGIT
 026742 002402                                  BLT     4$                  ;;BR IF DONE
 026744 005202                                  INC     R2                  ;;INCREASE THE BCD DIGIT BY 1
 026746 000774                                  BR      3$
 026750 060105                          4$:     ADD     R1,R5               ;;ADD BACK THE CONSTANT
 026752 005702                                  TST     R2                  ;;CHECK IF BCD DIGIT=0
 026754 001002                                  BNE     5$                  ;;FALL THROUGH IF 0
 026756 105716                                  TSTB    (SP)                ;;STILL DOING LEADING 0'S?
 026760 100407                                  BMI     7$                  ;;BR IF YES
 026762 106316                          5$:     ASLB    (SP)                ;;MSD?
 026764 103003                                  BCC     6$                  ;;BR IF NO
 026766 116663 000001 177777                    MOVB    1(SP),-1(R3)        ;;YES--SET THE SIGN
 026774 052702 000060                   6$:     BIS     #'0,R2              ;;MAKE THE BCD DIGIT ASCII
 027000 052702 000040                   7$:     BIS     #' ,R2              ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 027004 110223                                  MOVB    R2,(R3)+            ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 027006 005720                                  TST     (R0)+               ;;JUST INCREMENTING
 027010 020027 000010                           CMP     R0,#10              ;;CHECK THE TABLE INDEX
 027014 002746                                  BLT     2$                  ;;GO DO THE NEXT DIGIT
 027016 003002                                  BGT     8$                  ;;GO TO EXIT
 027020 010502                                  MOV     R5,R2               ;;GET THE LSD
 027022 000764                                  BR      6$                  ;;GO CHANGE TO ASCII
 027024 105726                          8$:     TSTB    (SP)+               ;;WAS THE LSD THE FIRST NON-ZERO?
 027026 100003                                  BPL     9$                  ;;BR IF NO
 027030 116663 177777 177776                    MOVB    -1(SP),-2(R3)       ;;YES--SET THE SIGN FOR TYPING
 027036 105013                          9$:     CLRB    (R3)                ;;SET THE TERMINATOR
 027040 012605                                  MOV     (SP)+,R5            ;;POP STACK INTO R5
 027042 012603                                  MOV     (SP)+,R3            ;;POP STACK INTO R3
 027044 012602                                  MOV     (SP)+,R2            ;;POP STACK INTO R2
```

```
 4180  027046  012601                            MOV     (SP)+,R1          ;;POP STACK INTO R1
 4181  027050  012600                            MOV     (SP)+,R0          ;;POP STACK INTO R0
 4182  027052  104400  027100                    TYPE    ,$DBLK            ;;NOW TYPE THE NUMBER
 4183  027056  016666  000002  000004            MOV     2(SP),4(SP)       ;;ADJUST THE STACK
 4184  027064  012616                            MOV     (SP)+,(SP)
 4185  027066  000002                            RTI                       ;;RETURN TO USER
 4186  027070  023420                  $DTBL:    10000.
 4187  027072  001750                            1000.
 4188  027074  000144                            100.
 4189  027076  000012                            10.
 4190  027100  000004                  $DBLK:    .BLKW   4
```

```
                                        .SBTTL  TRAP DECODER

                                        ;;*************************************************************
                                        ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
                                        ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                        ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                        ;*GO TO THAT ROUTINE.

027110  010046                          $TRAP:  MOV     R0,-(SP)                ;;SAVE R0
027112  016600  000002                          MOV     2(SP),R0                ;;GET TRAP ADDRESS
027116  005740                                  TST     -(R0)                   ;;BACKUP BY 2
027120  111000                                  MOVB    (R0),R0                 ;;GET RIGHT BYTE OF TRAP
027122  006300                                  ASL     R0                      ;;POSITION FOR INDEXING
027124  016000  027132                          MOV     $TRPAD(R0),R0           ;;INDEX TO TABLE
027130  000200                                  RTS     R0                      ;;GO TO ROUTINE

                                        .SBTTL  TRAP TABLE

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

                                        ;       ROUTINE
                                        ;       -------
027132                                  $TRPAD:
027132  026154                                  $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
027134  026462                                  $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
027136  026436                                  $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
027140  026476                                  $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
027142  026664                                  $TYPDS  ;;CALL=TYPDS    TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
027144  012737  027310  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC        ;;SET FOR FAST UP
027152  012737  000340  000026          MOV     #340,@#PWRVEC+2         ;;PRIO:7
027160  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
027162  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
027164  010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
027166  010346                          MOV     R3,-(SP)                ;;PUSH R3 ON STACK
027170  010446                          MOV     R4,-(SP)                ;;PUSH R4 ON STACK
027172  010546                          MOV     R5,-(SP)                ;;PUSH R5 ON STACK
027174  017746  151740                  MOV     @SWR,-(SP)              ;;PUSH @SWR ON STACK
027200  010637  027314                  MOV     SP,$SAVR6               ;;SAVE SP
027204  012737  027216  000024          MOV     #$PWRUP,@#PWRVEC        ;;SET UP VECTOR
027212  000000                          HALT
027214  000776                          BR      .-2                     ;;HANG UP

                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
027216  012737  027310  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC        ;;SET FOR FAST DOWN
027224  013706  027314                  MOV     $SAVR6,SP               ;;GET SP
027230  005037  027314                  CLR     $SAVR6                  ;;WAIT LOOP FOR THE TTY
027234  005237  027314          1$:     INC     $SAVR6                  ;;WAIT FOR THE INC
027240  001375                          BNE     1$                      ;;OF  WORD
027242  012677  151672                  MOV     (SP)+,@SWR              ;;POP STACK INTO @SWR
027246  012605                          MOV     (SP)+,R5                ;;POP STACK INTO R5
027250  012604                          MOV     (SP)+,R4                ;;POP STACK INTO R4
027252  012603                          MOV     (SP)+,R3                ;;POP STACK INTO R3
027254  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
027256  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
027260  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
027262  012737  027144  000024          MOV     #$PWRDN,@#PWRVEC        ;;SET UP THE POWER DOWN VECTOR
027270  012737  000340  000026          MOV     #340,@#PWRVEC+2         ;;PRIO:7
027276  104400                          TYPE                            ;;REPORT THE POWER FAILURE
027300  027316                  $PWRMG: .WORD   PWRMSG                  ;;POWER FAIL MESSAGE POINTER
027302  012716                          MOV     (PC)+,(SP)              ;;RESTART AT RSTART
027304  003244                  $PWRAD: .WORD   RSTART                  ;;RESTART ADDRESS
027306  000002                          RTI
027310  000000                  $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
027312  000776                          BR      .-2                     ;; BEFORE THE POWER DOWN WAS COMPLETE
027314  000000                  $SAVR6: 0                               ;;PUT THE SP HERE
027316  005015  042522  052123          PWRMSG: .ASCIZ  <15><12>/RESTARTED AFTER A POWER FAILURE/<15><12><12>
027324  051101  042524  020104
027332  043101  042524  020122
027340  020101  047520  042527
027346  020122  040506  046111
027354  051125  006505  005012
027362     000
```

```
                                            .SBTTL  ASCII MESSAGES

027363     015  051412  040524  MSG1:   .ASCIZ  <15><12>/START OF MD-11-DZVSC-B   VS60 INSTRUCTION TEST PART III/<15><12
027370  052122  047440  020106
027376  042115  030455  026461
027404  055104  051526  026503
027412  020102  020040  051526
027420  030066  044440  051516
027426  051124  041525  044524
027434  047117  052040  051505
027442  020124  040520  052122
027450  044440  044511  005015
027456     000
027457     015  045412  030524  MSG2:   .ASCIZ  <15><12>/KT11 ER TRAP/
027464  020061  051105  052040
027472  040522  000120
027476  005015  042515  047515  MSG3:   .ASCIZ  <15><12>/MEMORY SIZE =/
027504  054522  051440  055111
027512  020105  000075
027516  006513  000012          MSG4:   .ASCIZ  /K/<15><12>
027522  027114  027120  043040  EM1:    .ASCIZ  /L.P. FLAG INTR FAILURE/
027530  040514  020107  047111
027536  051124  043040  044501
027544  052514  042522     000
027551     123  046111  020117  EM2:    .ASCIZ  /SILO FAILURE AT LEVEL INDICATED/
027556  040506  046111  051125
027564  020105  052101  046040
027572  053105  046105  044440
027600  042116  041511  052101
027606  042105     000
027611     123  045524  050040  EM3:    .ASCIZ  /STK PTR DEC ER/
027616  051124  042040  041505
027624  042440  000122
027630  052123  041501  020113  EM4:    .ASCIZ  /STACK OVERFLOW ER/
027636  053117  051105  046106
027644  053517  042440  000122
027652  050104  020103  052123  EM5:    .ASCIZ  /DPC STKING ER/
027660  044513  043516  042440
027666  000122
027670  050104  020125  040516  EM6:    .ASCIZ  /DPU NAME STKING ER/
027676  042515  051440  045524
027704  047111  020107  051105
027712     000
027713     115  042117  020105  EM7:    .ASCIZ  /MODE STKING ER/
027720  052123  044513  043516
027726  042440  000122
027732  042526  052103  051117  EM10:   .ASCIZ  /VECTOR SCALE STKING ER/
027740  051440  040503  042514
027746  051440  045524  047111
027754  020107  051105     000
027761     103  040510  040522  EM11:   .ASCIZ  /CHARACTER SCALE STKING ER/
027766  052103  051105  051440
027774  040503  042514  051440
030002  045524  047111  020107
030010  051105     000
030013     103  040510  040522  EM12:   .ASCIZ  /CHARACTER ROTATE STKING ER/
```

```
030020  052103  051105  051040
030026  052117  052101  020105
030034  052123  044513  043516
030042  042440  000122
030046  047111  042524  051516  EM13:   .ASCIZ  /INTENSITY LEVEL STKING ER/
030054  052111  020131  042514
030062  042526  020114  052123
030070  044513  043516  042440
030076  000122
030100  047503  047514  020122  EM14:   .ASCIZ  /COLOR LEVEL STKING ER/
030106  042514  042526  020114
030114  052123  044513  043516
030122  042440  000122
030126  052111  046101  041511  EM15:   .ASCIZ  /ITALICS STKING ER/
030134  020123  052123  044513
030142  043516  042440  000122
030150  042515  052516  051440  EM16:   .ASCIZ  /MENU STKING ER/
030156  045524  047111  020107
030164  051105     000
030167     102  044514  045516  EM17:   .ASCIZ  /BLINK ENA STKING ER/
030174  042440  040516  051440
030202  045524  047111  020107
030210  051105     000
030213     123  047524  020120  EM20:   .ASCIZ  /STOP INT ENA STKING ER/
030220  047111  020124  047105
030226  020101  052123  044513
030234  043516  042440  000122
030242  050114  044040  052111  EM21:   .ASCIZ  /LP HIT DISABLE STKING ER/
030250  042040  051511  041101
030256  042514  051440  045524
030264  047111  020107  051105
030272     000
030273     114  020104  052123  EM22:   .ASCIZ  /LD STAUTS ENA STKING ER/
030300  052501  051524  042440
030306  040516  051440  045524
030314  047111  020107  051105
030322     000
030323     105  043504  020105  EM23:   .ASCIZ  /EDGE INT ENA STKING ER/
030330  047111  020124  047105
030336  020101  052123  044513
030344  043516  042440  000122
030352  044103  051101  051440  EM24:   .ASCIZ  /CHAR STRING ESCAPE STKING ER/
030360  051124  047111  020107
030366  051505  040503  042520
030374  051440  045524  047111
030402  020107  051105     000
030407     104  050105  044124  EM25:   .ASCIZ  /DEPTH QUE STKING ER/
030414  050440  042525  051440
030422  045524  047111  020107
030430  051105     000
030433     114  047111  020105  EM26:   .ASCIZ  /LINE TYPE STKING ER/
030440  054524  042520  051440
030446  045524  047111  020107
030454  051105     000
030457     111  052116  047105  EM27:   .ASCIZ  /INTENSITY ENA STKING ER/
030464  044523  054524  042440
```

M07


```
030472  040516  051440  045524
030500  047111  020107  051105
030506     000
030507     114  050056  020056  EM30:   .ASCIZ  /L.P. INTR ENA STKING ER/
030514  047111  051124  042440
030522  040516  051440  045524
030530  047111  020107  051105
030536     000
030537     114  050056  020056  EM31:   .ASCIZ  /L.P. SW INTR ENA STKING ER/
030544  053523  044440  052116
030552  020122  047105  020101
030560  052123  044513  043516
030566  042440  000122
030572  044123  043111  020124  EM32:   .ASCIZ  /SHIFT OUT STKING ER/
030600  052517  020124  052123
030606  044513  043516  042440
030614  000122
030616  052123  020113  052120  EM33:   .ASCIZ  /STK PTR INC ER/
030624  020122  047111  020103
030632  051105     000
030635     123  045524  052440  EM34:   .ASCIZ  /STK UNDERFLOW ER/
030642  042116  051105  046106
030650  053517  042440  000122
030656  050104  020103  047520  EM35:   .ASCIZ  /DPC POP ER/
030664  020120  051105     000
030671     116  046501  020105  EM36:   .ASCIZ  /NAME POP ER/
030676  047520  020120  051105
030704     000
030705     126  041505  047524  EM37:   .ASCIZ  /VECTOR SCALE POP ER/
030712  020122  041523  046101
030720  020105  047520  020120
030726  051105     000
030731     103  040510  020122  EM40:   .ASCIZ  /CHAR SCALE POP ER/
030736  041523  046101  020105
030744  047520  020120  051105
030752     000
030753     103  040510  020122  EM41:   .ASCIZ  /CHAR ROTATE POP ER/
030760  047522  040524  042524
030766  050040  050117  042440
030774  000122
030776  047503  047514  020122  EM42:   .ASCIZ  /COLOR LEVEL POP ER/
031004  042514  042526  020114
031012  047520  020120  051105
031020     000
031021     111  040524  044514  EM43:   .ASCIZ  /ITALICS POP ER/
031026  051503  050040  050117
031034  042440  000122
031040  042515  052516  050040  EM44:   .ASCIZ  /MENU POP ER/
031046  050117  042440  000122
031054  047111  042524  051516  EM45:   .ASCIZ  /INTENSITY ENA POP ER/
031062  052111  020131  047105
031070  020101  047520  020120
031076  051105     000
031101     114  050056  020056  EM46:   .ASCIZ  /L.P. INTR ENA POP ER/
031106  047111  051124  042440
031114  040516  050040  050117
```

```
031122  042440  000122
031126  027114  027120  051440  EM47:   .ASCIZ  /L.P. SW INTR ENA POP ER/
031134  020127  047111  051124
031142  042440  040516  050040
031150  050117  042440  000122
031156  044514  042516  052040  EM50:   .ASCIZ  /LINE TYPE POP ER/
031164  050131  020105  047520
031172  020120  051105     000
031177     111  052116  047105  EM51:   .ASCIZ  /INTENSITY LEVEL POP ER/
031204  044523  054524  046040
031212  053105  046105  050040
031220  050117  042440  000122
031226  046102  047111  020113  EM52:   .ASCIZ  /BLINK POP ER/
031234  047520  020120  051105
031242     000
031243     115  042117  020105  EM53:   .ASCIZ  /MODE POP ER/
031250  047520  020120  051105
031256     000
031257     123  047524  020120  EM54:   .ASCIZ  /STOP INTR ENA POP ER/
031264  047111  051124  042440
031272  040516  050040  050117
031300  042440  000122
031304  042504  052120  020110  EM55:   .ASCIZ  /DEPTH QUE POP ER/
031312  052521  020105  047520
031320  020120  051105     000
031325     105  043504  020105  EM56:   .ASCIZ  /EDGE INTR ENA POP ER/
031332  047111  051124  042440
031340  040516  050040  050117
031346  042440  000122
031352  044103  051101  051440  EM57:   .ASCIZ  /CHAR STRING ESC POP ER/
031360  051124  047111  020107
031366  051505  020103  047520
031374  020120  051105     000
031401     114  050056  020056  EM60:   .ASCIZ  /L.P. HIT DISABLE POP ER/
031406  044510  020124  044504
031414  040523  046102  020105
031422  047520  020120  051105
031430     000
031431     123  044510  052106  EM61:   .ASCIZ  /SHIFT OUT POP ER/
031436  047440  052125  050040
031444  050117  042440  000122
031452  042524  046522  047111  EM62:   .ASCIZ  /TERMINATE CHAR POP ER/
031460  052101  020105  044103
031466  051101  050040  050117
031474  042440  000122
031500  052123  020113  053117  EM63:   .ASCIZ  /STK OVFLO INT ER/
031506  046106  020117  047111
031514  020124  051105     000
031521     123  045524  052440  EM64:   .ASCIZ  /STK UNFLO INT ER/
031526  043116  047514  044440
031534  052116  042440  000122
031542  042504  052114  020101  EM65:   .ASCIZ  /DELTA LENGTH ER/
031550  042514  043516  044124
031556  042440  000122
031562  040524  043516  047105  EM66:   .ASCIZ  /TANGENT ER/
031570  020124  051105     000
```

```
031575     104  041520  030440  EM67:   .ASCIZ  /DPC 16-17 ER/
031602  026466  033461  042440
031610  000122
031612  051526  030066  046440  EM70:   .ASCIZ  /VS60 MEM ADRS FAILURE/
031620  046505  040440  051104
031626  020123  040506  046111
031634  051125  000105
031640  052123  050117  043040  EM71:   .ASCIZ  /STOP FLAG FAILED TO SET/
031646  040514  020107  040506
031654  046111  042105  052040
031662  020117  042523  000124
031670  052123  051101  020124  EM72:   .ASCIZ  /START FAILED TO SET 'TOP OF STK'/
031676  040506  046111  042105
031704  052040  020117  042523
031712  020124  052047  050117
031720  047440  020106  052123
031726  023513     000
031731     105  051122  041520  DH1:    .ASCIZ  /ERRPC   TSTNUM  BUSADRS EXPCT   RCVD/
031736  020040  052040  052123
031744  052516  020115  041040
031752  051525  042101  051522
031760  042440  050130  052103
031766  020040  051040  053103
031774  000104
031776  051105  050122  020103  DH2:    .ASCIZ  /ERRPC   TSTNUM  BUSADRS EXPCT   RCVD    STK SEL/
032004  020040  051524  047124
032012  046525  020040  052502
032020  040523  051104  020123
032026  054105  041520  020124
032034  020040  041522  042126
032042  020040  020040  052123
032050  020113  042523  000114
032056  051105  050122  020103  DH3:    .ASCIZ  /ERRPC   TSTNUM  GDPC-HI GDPC    BDPC-HI BDPC/
032064  020040  051524  047124
032072  046525  020040  042107
032100  041520  044055  020111
032106  042107  041520  020040
032114  020040  042102  041520
032122  044055  020111  042102
032130  041520     000
032133     105  051122  041520  DH4:    .ASCIZ  /ERRPC   TSTNUM  VECTOR  EXPCT   RCVD/
032140  020040  052040  052123
032146  052516  020115  053040
032154  041505  047524  020122
032162  042440  050130  052103
032170  020040  051040  053103
032176  000104
032200  051105  050122  020103  DH5:    .ASCIZ  /ERRPC   TSTNUM  BUSADRS EXPCT   RCVD    LEVEL/
032206  020040  051524  047124
032214  046525  020040  052502
032222  040523  051104  020123
032230  054105  041520  020124
032236  020040  041522  042126
032244  020040  020040  042514
032252  042526  000114
                                        .EVEN
```

```
032256  001116  002204  001122  DT1:    .WORD   $ERRPC,TSTNUM,$BDADR,$GDDAT,$BDDAT,0
032264  001124  001126  000000
032272  001116  002204  001122  DT2:    .WORD   $ERRPC,TSTNUM,$BDADR,$GDDAT,$BDDAT,$TMPO,0
032300  001124  001126  001164
032306  000000
032310  001116  002204  002224  DT3:    .WORD   $ERRPC,TSTNUM,G1716,$GDDAT,B1716,$BDDAT,0
032316  001124  002226  001126
032324  000000
032326  001116  002204  001164  DT4:    .WORD   $ERRPC,TSTNUM,$TMPO,$GDDAT,$BDDAT,0
032334  001124  001126  000000

                                ;;*************************************************************
                                        ;THIS IS THE WORKING AREA FOR ALL VS60 NPR'S (INSTR & DATA)
                                        ;FROM HERE TO THE END OF MEMORY
                                ;;*************************************************************
032342  000000                  BUFFER: 0                       ;LOCATIONS START HERE FOR TESTING
        000001                          .END                    ;END OF PROGRAM
```

D08


```
ABASE = 172000      433#    535     576
ACDW1 = 000000      535     578
ACDW2 = 000000      535
ACPUOP= 000000      535     550
ADDW0 = 000000      535
ADDW1 = 000000      535
ADDW10= 000000      535
ADDW11= 000000      535
ADDW12= 000000      535
ADDW13= 000000      535
ADDW14= 000000      535
ADDW15= 000000      535
ADDW2 = 000000      535
ADDW3 = 000000      535
ADDW4 = 000000      535
ADDW5 = 000000      535
ADDW6 = 000000      535
ADDW7 = 000000      535
ADDW8 = 000000      535
ADDW9 = 000000      535
ADEVCT= 000000      535     541
ADEVM = 000000      535     577
AENV  = 000000      535     546
AENVM = 000000      535     547
AFATAL= 000000      535     538
AMADR1= 000000      535     563
AMADR2= 000000      535     567
AMADR3= 000000      535     570
AMADR4= 000000      535     573
AMAMS1= 000000      535     557
AMAMS2= 000000      535     565
AMAMS3= 000000      535     568
AMAMS4= 000000      535     571
AMSGAD= 000000      535     543
AMSGLG= 000000      535     544
AMSGTY= 000000      535     537
AMTYP1= 000000      535     558
AMTYP2= 000000      535     566
AMTYP3= 000000      535     569
AMTYP4= 000000      535     572
ANAME   002264     1036#
APASS = 000000      535     540
APRIOR= 000004      435#    535
APTCSU= 000040     3867#   3998
APTENV= 000001     3823    3865#   3899    3991
APTSIZ= 000200     1086    3864#
APTSPO= 000100     3825    3866#   3993
ASWREG= 000000      535     548
ATESTN= 000000      535     539
AUNIT = 000000      535     542
AUSWR = 000000      535     549
AVECT1= 000320      434#    535     574
AVECT2= 000000      535     575
BADR    023402     3361    3395    3437#
BIT0  = 000001      418#   1195    1218    1260    1263    1286    1289    1292    1319    1354    1388    1421    1453
                   1485    1517    1548    1579    1610    1641    1672    1699    1729    1760    1791    1822    1852
```

		1882	1913	1945	1976	2007	2038	2069	2104	2175	2178	2606	2633	2660
		2687	2714	2741	2808	2836	2864	2903	3165	3213	3216	3284	3287	3527
BIT00 =	000001	408*	418											
BIT01 =	000002	407*	417											
BIT02 =	000004	406*	416											
BIT03 =	000010	405*	415											
BIT04 =	000020	404*	414											
BIT05 =	000040	403*	413											
BIT06 =	000100	402*	412											
BIT07 =	000200	401*	411											
BIT08 =	000400	400*	410	3777										
BIT09 =	001000	399*	409	3785	3909									
BIT1 =	000002	417*												
BIT10 =	002000	398*	3887											
BIT11 =	004000	397*	3792											
BIT12 =	010000	396*	3583											
BIT13 =	020000	395*	3894											
BIT14 =	040000	394*	3763											
BIT15 =	100000	393*												
BIT2 =	000004	416*	2776	3507	3649	3654								
BIT3 =	000010	415*	2101											
BIT4 =	000020	414*	2813	2841										
BIT5 =	000040	413*												
BIT6 =	000100	412*												
BIT7 =	000200	411*												
BIT8 =	000400	410*												
BIT9 =	001000	409*												
BPTVEC=	000014	425*												
BSTART	023416	3358	3444*											
BSTRT	024236	3497	3565*											
BUFFER	032342	1191*	1193	1215*	1216	1252*	1255	1258	1278*	1279	1284	1308*	1309*	1315
		1332*	1343*	1348*	1349*	1350	1378*	1379*	1384	1401*	1411*	1412*	1417	1434*
		1443*	1444*	1449	1464*	1475*	1476*	1481	1483	1496*	1507*	1508*	1513	1528*
		1538*	1539*	1544	1559*	1569*	1570*	1575	1590*	1600*	1601*	1606	1608	1621*
		1631*	1632*	1637	1652*	1662*	1663*	1668	1683*	1692*	1693*	1695	1707	1709*
		1719*	1720*	1725	1740*	1750*	1751*	1756	1771*	1781*	1782*	1787	1802*	1812*
		1813*	1818	1833*	1844*	1845*	1847	1860	1862*	1872*	1873*	1878	1880	1893*
		1903*	1904*	1909	1924*	1935*	1936*	1941	1956*	1966*	1967*	1972	1987*	1997*
		1998*	2003	2018*	2028*	2029*	2034	2049*	2059*	2060*	2065	2080*	2090*	2091*
		2092*	2097	2115*	2126*	2127	2147*	2148	2169	2171*	2172*	2173	2176	2195*
		2196*	2206*	2220*	2221*	2231*	2245*	2246*	2257*	2270*	2279*	2280*	2294*	2295*
		2306*	2319*	2320*	2329*	2342*	2351*	2352*	2365*	2374*	2375*	2388*	2397*	2398*
		2409*	2420*	2421*	2432*	2443*	2444*	2455*	2466*	2467*	2478*	2489*	2490*	2503*
		2504*	2514*	2528*	2529*	2540*	2551*	2560*	2563*	2577*	2578*	2579*	2581*	2600*
		2604	2615*	2616*	2627*	2631	2642*	2643*	2654*	2658	2669*	2670*	2681*	2685
		2696*	2697*	2708*	2712	2723*	2724*	2735*	2739	2750*	2751*	2763	2772	2785*
		2786*	2796	2797*	2798*	2799*	2803*	2804	2826	2827*	2828*	2829*	2832	2833*
		2860*	2862	2899*	2901	2934	2945*	2967	2978*	3000	3012*	3013*	3043	3053*
		3083	3093*	3199*	3207	3312	3415	3420	3425	3430	3444	3446	3447	3448
		3464	3565	3567	3568	3569	3636*	3637*	3638*	3645	3650	3738	4567*	
B1716	002226	1015*	3295*	3298	4557									
CALTAN	024706	3054	3094	3710*										
CENAB	002216	1011*	1100*	1119*	3147	3191								
CNTR	025070	3736*	3742*	3746*										
CONS	002266	1037*	2289	2297	2361	2367	2384	2390	2407	2413	2430	2436	2453	2459
		2476	2482	3373*	3379	3381	3440	3511*	3517	3519				

```
CR     = 000015      333#   4037   4047
CRLF   = 000200      334#   4008   4047
CSIZR    002616     1100#
DDISP  = 177570      340#    512   1074
DELAY    024426     1194    1217   1259   1285   1288   1318   1353   1387   1420   1452   1484   1516   1547
                    1578    1609   1640   1671   1698   1728   1759   1790   1821   1651   1881   1912   1944
                    1975    2006   2037   2068   2100   2103   2174   2177   2605   2632   2659   2686   2713
                    2740    2775   2807   2812   2835   2840   2863   2902   3164   3212   3215   3283   3286
                    3506    3526   3627#  3648   3653
DH1      031731      599     613    620    779    786    793    800    807    814    821    827    834    841
                     848     855    862    869    875    882    889    896    903    910    917    923    930
                     937     944    951    971    985    992   4513#
DH2      031776      627     633    640    647    654    661    668    675    682    689    696    703    710
                     717     724    730    737    744    751    758    765    773   4520#
DH3      032056      978    4528#
DH4      032133      958     965   4536#
DH5      032200      606    4543#
DISPLA   001142      512#   1074*  1082*  3807*  3886*
DISPRE   000174      443#   1082
DLYCNT   002206     1007#   3627*  3628*
DNAME    002270     1038#   2191   2196   3441
DO       024472     2197    2222   2247   2271   2296   2321   2343   2366   2389   2412   2435   2458   2481
                    2505    2530   2554   2580   2603   2630   2657   2684   2711   2738   3644#
DPC      002244     1028#   1094   1098   1193*  1216*  1239*  1258*  1284*  1287*  1315*  1317*  1350*  1352*
                    1384*   1386*  1417*  1419*  1449*  1451*  1481*  1483*  1513*  1515*  1544*  1546*  1575*
                    1577*   1606*  1608*  1637*  1639*  1668*  1670*  1695*  1697*  1725*  1727*  1756*  1758*
                    1787*   1789*  1818*  1820*  1847*  1850*  1878*  1880*  1909*  1911*  1941*  1943*  1972*
                    1974*   2003*  2005*  2034*  2036*  2065*  2067*  2097*  2099*  2102*  2127*  2148*  2168
                    2173*   2176*  2179   2582*  2604*  2631*  2658*  2685*  2712*  2739*  2772*  2774*  2804*
                    2806*   2809*  2810   2811*  2814   2832*  2834*  2837*  2838   2839*  2842   2862*  2901*
                    3121*   3134*  3163*  3209*  3210   3211*  3214*  3217   3282*  3285*  3288*  3366*  3368*
                    3437    3503*  3505*  3523   3525*  3645*  3647*  3650*  3652*  3738*  3741*
DPCONV   024556     3272    3660#
DPCON1   024570     3293    3665#
DPCSAV   024226     3523*   3524*  3525   3528*  3557#
DSWR   = 177570      339#    511   1073
DT1      032256      600     614    621    780    787    794    801    808    815    822    828    835    842
                     849     856    863    870    876    883    890    897    904    911    918    924    931
                     938     945    952    972    986    993   4552#
DT2      032272      607     628    634    641    648    655    662    669    676    683    690    697    704
                     711     718    725    731    738    745    752    759    766    774   4554#
DT3      032310      979    4557#
DT4      032326      959     966   4560#
EMTVEC=  000030      428#   1057*  1058*
EM1      027522      598    4292#
EM10     027732      646    4318#
EM11     027761      653    4322#
EM12     030013      660    4327#
EM13     030046      667    4332#
EM14     030100      674    4337#
EM15     030126      681    4341#
EM16     030150      688    4344#
EM17     030167      695    4347#
EM2      027551      605    4296#
EM20     030213      702    4351#
EM21     030242      709    4355#
```

```
EM22     030273       716    4360#
EM23     030323       723    4365#
EM24     030352       729    4369#
EM25     030407       736    4374#
EM26     030433       743    4378#
EM27     030457       750    4382#
EM3      027611       612    4302#
EM30     030507       757    4387#
EM31     030537       764    4392#
EM32     030572       772    4397#
EM33     030616       778    4401#
EM34     030635       785    4404#
EM35     030656       792    4407#
EM36     030671       799    4409#
EM37     030705       806    4412#
EM4      027630       619    4305#
EM40     030731       813    4416#
EM41     030753       820    4420#
EM42     030776       826    4424#
EM43     031021       833    4428#
EM44     031040       840    4431#
EM45     031054       847    4433#
EM46     031101       854    4437#
EM47     031126       861    4441#
EM5      027652       626    4308#
EM50     031156       868    4445#
EM51     031177       874    4448#
EM52     031226       881    4452#
EM53     031243       888    4455#
EM54     031257       895    4458#
EM55     031304       902    4462#
EM56     031325       909    4465#
EM57     031352       916    4469#
EM6      027670       632    4311#
EM60     031401       922    4473#
EM61     031431       929    4478#
EM62     031452       936    4481#
EM63     031500       943    4485#
EM64     031521       950    4488#
EM65     031542       957    4491#
EM66     031562       964    4494#
EM67     031575       970    4496#
EM7      027713       639    4315#
EM70     031612       977    4499#
EM71     031640       984    4503#
EM72     031670       991    4507#
ERRVEC=  000004       421#   1071     1072*    1083*    1102*    1113*    1124*    1135*    1137*    1149*    3768     3769*    3771*
                     3774*
EXECUT   025020      2946    2979     3014     3055     3095     3735#
GNS    = ****** U     442    4215     4216     4217     4218     4219
G1716    002224      1014#   3274*    3298     4557
HT     = 000011       331#   4006     4047
IOTVEC=  000020       426#   1055*    1056*
KIPAR0   002230      1019#   1126*
KIPAR2   002234      1021#   1128*    1143*    1144     1150*    1151     3256*    3264     3266*    3269*    3271     3301*    3302
KIPAR7   002240      1023#   1130*    1144
```

```
KIPDR0   002232        1020#   1127*
KIPDR2   002236        1022#   1129*
KIPDR7   002242        1024#   1131*
KTENAB   002220        1012#   1101*   1110*   1120    1134*   1154*   3252    3255*
KTERRV=  000250        1001#   1123*
KTMAX    002222        1013#   1151*   1152    1156    3264    3302
KTSER    024610        1123    3676#
KTSIZR   002736        1121    1123#
KTSR0  = 177572        1002#   1125    1139*   1142*   1147*   3259*   3263*   3676*
LDRSV1   002210        1008#   3160*   3174    3204*   3224
LDRSV2   002212        1009#   3205*   3225
LF     = 000012         332#   4041    4047
MAINT1=  010000         998#   3119
MAINT2=  020000         999#
MEMMAX   002214        1010#   1115*   1162    3152    3195
MSG1     027363        1092    4274#
MSG2     027457        3677    4285#
MSG3     027476        1170    4288#
MSG4     027516        1173    4291#
OPEN   = 000000         997#   1006    1007    1008    1009    1010    1011    1012    1013    1014    1015    1028    1029
                       1030    1031    1032    1033    1034    1035    1036    1037    1038    1039    1040    1041    1042
                       1043
PC     =%000007         352#   2197*   2222*   2247*   2271*   2296*   2321*   2343*   2366*   2389*   2412*   2435*   2458*
                       2481*   2505*   2530*   2554*   2580*   2603*   2630*   2657*   2684*   2711*   2738*   2884*   2923*
                       3054*   3061*   3094*   3101*   3272*   3293*   3411*   3554*   3600*   3603*   3613*   3618    3655*
                       3671*   3692*   3704*   3726*   3842*   3859*   3896*   3902*   3958*   3996*   4015*   4022*   4029*
                       4043*   4045*   4258
PIRQ   = 177772         338#
PIRQVE=  000240         432#
PCPSET   024442        2193    2218    2243    2268    2292    2317    2340    2363    2386    2410    2433    2456    2479
                       2501    2526    2552    2575    2601    2628    2655    2682    2709    2736    3635#
PR0    = 000000         355#
PR1    = 000040         356#
PR2    = 000100         357#
PR3    = 000140         358#
PR4    = 000200         359#
PR5    = 000240         360#
PR6    = 000300         361#
PR7    = 000340         362#
PS     = 177776         335#    336
PSW    = 177776         336#   1176*   1210*   2759*   2861*   2865*   2900*   2904*   3365*   3374*   3502*   3512*   3684*
PWRMSG   027316        4257    4264#
PWRVEC=  000024         427#   1061*   1062*   4226*   4227*   4236*   4242*   4254*   4255*
RCNT     023430        3360    3404    3451#
RCNT1    024250        3499    3545    3572#
RESVEC=  000010         422#
RLO      002254        1032#   3120*   3133*   3140*   3270*   3277*   3280*   3304*
RSTART   003244         483    1169    1174#   3619    4259
RSTVEC   024626        2884    2923    3411    3554    3684#
R0     =%000000         343#   1093*   1095    1096*   1103*   1104    1107*   1108    1114*   1115    1116    1138*   1140
                       1141    1156*   1157*   1158*   1159*   1160*   1162*   1163*   1164*   1165*   1166*   1167*   1171
                       1189*   1192    1201*   1214*   1219*   1253*   1261    1266    1269*   1281*   1290    1295    1298*
                       1313*   1320    1325    1328*   1346*   1355    1360    1363*   1382*   1389    1394    1397*   1415*
                       1422    1427    1430*   1447*   1454    1459    1462*   1479*   1486    1491    1494*   1511*   1518
                       1523    1526*   1542*   1549    1554    1557*   1573*   1580    1585    1588*   1604*   1611    1616
                       1619*   1635*   1642    1647    1650*   1666*   1673    1678    1681*   1723*   1730    1735    1738*
```

```
                          1754*  1761   1766   1769*  1785*  1792   1797   1800*  1816*  1823   1828   1831*  1876*
                          1883   1888   1891*  1907*  1914   1919   1922*  1939*  1946   1951   1954*  1970*  1977
                          1982   1985*  2001*  2008   2013   2016*  2032*  2039   2044   2047*  2063*  2070   2075
                          2078*  2095*  2105   2110   2113*  2146*  2150*  2242*  2246   2254*  2291*  2295   2303*
                          2525*  2529   2536*  2800*  2802   2803   2818*  2858*  2866*  2896*  2905*  2934*  2935*
                          2936*  2937*  2938*  2939*  2940*  2941*  2942*  2943*  2945   2952   2955*  2956   2967*
                          2968*  2969*  2970*  2971*  2972*  2973*  2974*  2975*  2976*  2978   2985   2988*  2989
                          3000*  3001*  3002*  3003*  3004*  3005*  3006*  3007*  3008*  3009*  3011   3012   3023*
                          3026   3031   3043*  3044*  3045*  3046*  3047*  3048*  3049*  3050*  3051*  3052*  3053
                          3064   3066*  3067   3083*  3084*  3085*  3086*  3087*  3088*  3089*  3090*  3091*  3092*
                          3093   3104   3106*  3107   3117*  3120   3129*  3152*  3153   3154   3180   3195*  3196
                          3197   3231   3258*  3260*  3261*  3262*  3271*  3273   3290*  3291*  3294   3312*  3313*
                          3314*  3315*  3316*  3317*  3318*  3319*  3320*  3321*  3322*  3323*  3324*  3325*  3326*
                          3327*  3328*  3329*  3330*  3331*  3333*  3334*  3335*  3336*  3337*  3338*  3339*  3340*
                          3341*  3342*  3343*  3344*  3345*  3346*  3347*  3348*  3349*  3350*  3351*  3352*  3353*
                          3355*  3356*  3357*  3367*  3370*  3464*  3465*  3466*  3467*  3468*  3469*  3470*  3471*
                          3472*  3473*  3474*  3475*  3476*  3477*  3478*  3479*  3480*  3481*  3482*  3483*  3484*
                          3485*  3486*  3487*  3488*  3489*  3490*  3491*  3492*  3493*  3494*  3495*  3504*  3508*
                          3610*  3613   3661*  3662*  3663*  3664*  3667*  3670*  3686*  3687*  3688*  3689*  3819
                          3827*  3831   3832   3834*  3835*  3836   3858*  3930   3931*  3932*  3939*  3940*  3941*
                          3942*  3943*  3944   3949   3954*  3956*  3960   3962   3989   3990*  3995   4000   4003*
                          4137   4147*  4151   4167   4168   4181*  4199   4200*  4201   4202*  4203*  4204*  4205*
                          4228   4253*
R1      =%000001           344#  1094*  1095*  1097*  1098   1104*  1105   1106   1254*  1257   1268*  1280*  1283
                          1297*  1312*  1314   1327*  1345*  1347   1362*  1381*  1383   1396*  1414*  1416   1429*
                          1446*  1448   1461*  1478*  1480   1493*  1510*  1512   1525*  1541*  1543   1556*  1572*
                          1574   1587*  1603*  1605   1618*  1634*  1636   1649*  1665*  1667   1680*  1722*  1724
                          1737*  1753*  1755   1768*  1784*  1786   1799*  1815*  1817   1830*  1875*  1877   1890*
                          1906*  1908   1921*  1938*  1940   1953*  1969*  1971   1984*  2000*  2002   2015*  2031*
                          2033   2046*  2062*  2064   2077*  2034*  2096   2112*  2853*  2854*  2855*  2892*  2893*
                          2894*  3010*  3013   3025*  3026   3029   3042*  3060   3071*  3082*  3100   3153*  3155*
                          3183   3196*  3198*  3234   3257*  3261   3267*  3274   3275*  3278*  3292*  3295   3358*
                          3366   3407*  3409   3497*  3503   3550*  3552   3660*  3666*  3669*  3685*  3690*  3820
                          3857*  4138   4151*  4152   4156   4180*  4229   4252*
R2      =%000002           345#  1342*  1349   1365   1367*  2999*  3017   3018   3028*  3041*  3058   3059   3070*
                          3081*  3098   3099   3151*  3175   3177*  3179*  3181   3194*  3226   3228*  3230*  3232
                          3359*  3387   3391*  3396   3398*  3406*  3498*  3532   3536*  3537   3539*  3546*  3547
                          3549*  4139   4150*  4154*  4157   4164*  4165*  4166   4171*  4179*  4230   4251*
R3      =%000003           346#  3154*  3158   3159*  3160   3161*  3163   3174*  3180*  3181*  3183   3197*  3202
                          3203*  3204   3205   3206*  3207*  3209   3224*  3225*  3231*  3232*  3234   3360*  3367
                          3399*  3404*  3499*  3504   3540*  3545*  3711*  3718*  3722*  4080   4089*  4095*  4096*
                          4099*  4104*  4105*  4106   4115*  4140   4148*  4149*  4163*  4166*  4175*  4176*  4178*
                          4231   4250*
R4      =%000004           347#  2193*  2218*  2243*  2268*  2292*  2317*  2340*  2363*  2386*  2410*  2433*  2456*
                          2479*  2501*  2526*  2552*  2575*  2601*  2628*  2655*  2682*  2709*  2736*  2763*  2764*
                          2765*  2766*  2767*  2768*  2769*  2770*  2773*  2777*  3156*  3162   3171*  3200*  3208
                          3221*  3361*  3385   3392*  3393   3395*  3638   3639*  4081   4083*  4084*  4085*  4086
                          4087*  4101   4103*  4111*  4114*  4232   4249*
R5      =%000005           348#  1194*  1217*  1259*  1285*  1288*  1318*  1353*  1387*  1420*  1452*  1484*  1516*
                          1547*  1578*  1609*  1640*  1671*  1698*  1728*  1759*  1790*  1821*  1851*  1881*  1912*
                          1944*  1975*  2006*  2037*  2068*  2100*  2103*  2174*  2177*  2605*  2632*  2659*  2686*
                          2713*  2740*  2775*  2807*  2812*  2835*  2840*  2863*  2902*  2946*  2979*  3014*  3055*
                          3060*  3095*  3100*  3164*  3212*  3215*  3283*  3286*  3362*  3405*  3406   3506*  3526*
                          3627   3630*  3648*  3653*  3700*  3701*  3702*  3703   3736   3744*  4082   4088*  4090*
                          4092*  4093*  4094*  4095   4113*  4141   4143*  4145*  4152*  4156*  4171   4177*  4233
                          4248*
```

```
R6      =%000006     349#   351    1049*  1050*  1051   1112   1133   1148   2872   2911
R7      =%000007     350#   352
SETPTR   002576     1095#  1099
SILOR    023330     3359   3362   3398   3415#
SILOR1   024230     3498   3539   3549   3559#
SF      =%000006     351#  1053*  1071*  1079*  1083   1171*  1175*  3375   3376   3377   3378   3384   3513
                    3514   3515   3516   3522   3607*  3768*  3771   3773   3774   3803   3804   3808*  3819*
                    3820*  3827   3828*  3839   3840*  3841*  3851   3852*  3857   3858   3891   3911*  3914*
                    3930*  3935*  3956   3960*  3989*  3990   4000*  4002   4003   4004*  4006   4008   4010
                    4016   4018*  4020*  4028*  4032   4036   4037   4041   4072*  4073   4074   4075*  4080*
                    4081*  4082*  4088   4113   4114   4115   4116*  4117*  4137*  4138*  4139*  4140*  4141*
                    4142*  4143   4146*  4159   4161*  4163   4173   4175   4177   4178   4179   4180   4181
                    4183*  4184*  4199*  4200   4228*  4229*  4230*  4231*  4232*  4233*  4234*  4235   4243*
                    4247   4248   4249   4250   4251   4252   4253   4258*
SREG0    002246     1029#  2315   2322   2499   2506   2523   2531   2549   2555   2573   2584   2880   2919
                    3438
SREG1    002256     1033#  1211   1221   1225   2149   2152   2156   2216   2223   2240   2248   2266   2272
                    2338   2344   2856   2868   2873   2895   2907   2912   3122   3123   3136   3439
STACK  = 001100      326#  1053   1175
START    002304      446   1046#
START1   002566     1091   1093#
STKLMT= 177774       337#
STKPT    002276     1041#  1188   1192*  1196   1212*  1236   1238*  1240   1257*  1261*  1283*  1290*  1314*
                    1320*  1347*  1355*  1383*  1389*  1416*  1422*  1448*  1454*  1480*  1486*  1512*  1518*
                    1543*  1549*  1574*  1580*  1605*  1611*  1636*  1642*  1667*  1673*  1694*  1700*  1724*
                    1730*  1755*  1761*  1786*  1792*  1817*  1823*  1846*  1853*  1877*  1883*  1908*  1914*
                    1940*  1946*  1971*  1977*  2002*  2008*  2033*  2039*  2064*  2070*  2096*  2105*  2124*
                    2128   2129   2144*  2170*  2607*  2634*  2661*  2688*  2715*  2742*  2771*  2779*  2801*
                    2831*  2857*  2897*  3119*  3162*  3166*  3208*  3281*  3364*  3372*  3501*  3510*  3530*
                    3635*  3644*  3646*  3651*  3737*  3739*  3740*
STKVAL   002272     1039#  1251   1262   1277   1291   1316   1321   1351   1356   1385   1390   1418   1423
                    1450   1455   1482   1487   1514   1519   1545   1550   1576   1581   1607   1612   1638
                    1643   1669   1674   1696   1701   1726   1731   1757   1762   1788   1793   1819   1824
                    1848   1854   1879   1884   1910   1915   1933   1942   1947   1973   1978   2004   2009
                    2035   2040   2066   2071   2098   2106   2598   2608   2625   2635   2652   2662   2679
                    2689   2706   2716   2733   2743   2761   2780   3150   3167   3529   3531
SWR      001140      511#  1051   1073*  1075   1081*  1088*  3583   3763   3777   3779   3785   3792   3887
                    3894   3906   3909   4234   4247*
SWREG    000176      444#  1081
SW0    = 000001      390#
SW00   = 000001      380#   390
SW01   = 000002      379#   389
SW02   = 000004      378#   388
SW03   = 000010      377#   387
SW04   = 000020      376#   386
SW05   = 000040      375#   385
SW06   = 000100      374#   384
SW07   = 000200      373#   383
SW08   = 000400      372#   382
SW09   = 001000      371#   381
SW1    = 000002      389#
SW10   = 002000      370#
SW11   = 004000      369#
SW12   = 010000      368#
SW13   = 020000      367#
SW14   = 040000      366#
```

```
SW15   = 100000            365#
SW2    = 000004            388#
SW3    = 000010            387#
SW4    = 000020            386#
SW5    = 000040            385#
SW6    = 000100            384#
SW7    = 000200            383#
SW8    = 000400            382#
SW9    = 001000            381#
TANCON   024662           3061     3101     3699#
TANGNT   025016           3713*    3715     3717*    3721*    3728#
TBITVE=  000014            423#
TERMCH   002274           1040#    2802*    2830*
TKVEC  = 000060            430#
TPVEC  = 000064            431#
TRAPVE=  000034            429#    1059*    1060*
TRTVEC=  000014            424#
TSTNUM   002204           1006#    3883*    4552     4554     4557     4560
TST1     003264           1185#
TST10    004546           1366     1374#
TST100   020370           2962#
TST101   020544           2995#
TST102   020742           3024     3037#
TST103   021134           3069     3077#
TST104   021312           3113#
TST105   021506           3145#
TST106   021722           3148     3182     3189#
TST107   022174           3192     3233     3240#
TST11    004744           1400     1407#
TST110   022536           3253     3309#
TST111   023442           3413     3461#
TST112   024262           3555     3581#
TST12    005144           1433     1440#
TST13    005334           1471#
TST14    005534           1503#
TST15    005732           1535#
TST16    006122           1566#
TST17    006310           1597#
TST2     003412           1208#
TST20    006502           1628#
TST21    006670           1659#
TST22    007060           1689#
TST23    007230           1708     1716#
TST24    007420           1747#
TST25    007606           1778#
TST26    007776           1809#
TST27    010166           1840#
TST3     003546           1226     1234#
TST30    010346           1869#
TST31    010534           1900#
TST32    010724           1931#
TST33    011122           1963#
TST34    011312           1994#
TST35    011502           2025#
TST36    011672           2056#
TST37    012062           2087#
```

```
TST4      003632       1243    1249#
TST40     012272       2122#
TST41     012404       2141#
TST42     012534       2166#
TST43     012652       2181    2187#
TST44     013006       2204    2212#
TST45     013142       2229    2237#
TST46     013300       2255    2263#
TST47     013426       2278    2286#
TST5      003772       1275#
TST50     013564       2304    2312#
TST51     013712       2328    2335#
TST52     014040       2350    2358#
TST53     014166       2373    2381#
TST54     014314       2396    2404#
TST55     014442       2419    2427#
TST56     014570       2442    2450#
TST57     014716       2465    2473#
TST6      004144       1305#
TST60     015044       2488    2496#
TST61     015172       2512    2520#
TST62     015330       2538    2546#
TST63     015456       2562    2569#
TST64     015624       2595#
TST65     015772       2614    2622#
TST66     016142       2641    2649#
TST67     016312       2668    2676#
TST7      004342       1331    1338#
TST70     016462       2695    2703#
TST71     016632       2722    2730#
TST72     017002       2749    2757#
TST73     017220       2793#
TST74     017406       2824#
TST75     017554       2844    2850#
TST76     017774       2889#
TST77     020214       2928#
TYPCOR    003202       1118    1122    1136    1155    1162#
TYPDS   = 104404       1172    3608    4219#
TYPE    = 104400       1092    1170    1173    3606    3609    3677    3889    3897    3929    3946    3948    3951    3953
                       3957    3964    4011    4107    4182    4215#   4256
TYPMEM    003216       1161    1167#
TYPOC   = 104401       3937    3961    4216#
TYPON   = 104403       4218#
TYPOS   = 104402       4217#
XDOFF     002260       1034#   3071    3082
XPOS      002250       1030#   2933    2949    3028    3070    3081    3290
YDOFF     002262       1035#   3042
YPOS      002252       1031#   2966    2982    2999    3041
ZDOFF     002302       1043#
ZPOS      002300       1042#
$APTHD    001000        468     474#
$ASTAT= ****** U       3845    3860
$ATYC     025374       3816    3818#
$ATY1     025350       3814#
$ATY3     025356       3815#   3996
$ATY4     025366       3817#   3902
```

```
$AUTOB   001134         508#
$BASE    001256         576#    1093
$BDADR   001122         503#    1188*   1211*   1236*   1251*   1277*   1316*   1351*   1385*   1418*   1450*   1482*   1514*
                       1545*    1576*   1607*   1638*   1669*   1696*   1726*   1757*   1788*   1819*   1848*   1879*   1910*
                       1933*    1942*   1973*   2004*   2035*   2066*   2098*   2128*   2149*   2168*   2191*   2216*   2240*
                       2266*    2289*   2315*   2338*   2361*   2384*   2407*   2430*   2453*   2476*   2499*   2523*   2549*
                       2573*    2598*   2625*   2652*   2679*   2706*   2733*   2761*   2810*   2838*   2856*   2895*   2933*
                       2966*    3017*   3058*   3098*   3122*   3150*   3210*   3379*   3385*   3386    3517*   3529*   4552
                       4554
$BDDAT   001126         505#    1196*   1197*   1198    1223*   1228*   1240*   1241*   1242    1262*   1263*   1264    1291*
                       1292*    1293    1321*   1322*   1323    1356*   1357*   1358    1390*   1391*   1392    1423*   1424*
                       1425     1455*   1456*   1457    1487*   1488*   1489    1519*   1520*   1521    1550*   1551*   1552
                       1581*    1582*   1583    1612*   1613*   1614    1643*   1644*   1645    1674*   1675*   1676    1701*
                       1702*    1703    1731*   1732*   1733    1762*   1763*   1764    1793*   1794*   1795    1824*   1825*
                       1826     1854*   1855*   1856    1884*   1885*   1886    1915*   1916*   1917    1947*   1948*   1949
                       1978*    1979*   1980    2009*   2010*   2011    2040*   2041*   2042    2071*   2072*   2073    2106*
                       2107*    2108    2129*   2130*   2131    2154*   2159*   2179*   2180    2198*   2199*   2200    2223*
                       2224*    2225    2248*   2249*   2250    2272*   2273*   2274    2297*   2298*   2299    2322*   2323*
                       2324     2344*   2345*   2346    2367*   2368*   2369    2390*   2391*   2392    2413*   2414*   2415
                       2436*    2437*   2438    2459*   2460*   2461    2482*   2483*   2484    2506*   2507*   2508    2531*
                       2532*    2533    2555*   2556*   2557    2584*   2585*   2586    2608*   2609*   2610    2635*   2636*
                       2637     2662*   2663*   2664    2689*   2690*   2691    2716*   2717*   2718    2743*   2744*   2745
                       2780*    2781*   2782    2814*   2815    2842*   2843    2868*   2869*   2875*   2882*   2907*   2908*
                       2914*    2921*   2949*   2950    2982*   2983    3018*   3019    3059*   3062    3099*   3102    3123*
                       3124*    3125    3136*   3137*   3167*   3168    3217*   3218    3294*   3296    3381*   3386*   3388
                       3519*    3531*   3533    3699*   3703*   4552    4554    4557    4560
$BELL    001172         527#    3889    3921
$CDWI    001262         578#
$CHARC   026432        4013*    4023*   4030    4039*   4044#
$CKSWR= ****** U       4222
$CMTAG   001100         491#    1048    1049    1057    1063    1064    1065
$CM1   = 000001         523#     524#
$CM2   = 000002         523#     524#
$CM3   = 000001         521#     523
$CM4   = 000001         524#     525#
$CPUOP   001230         550#
$CRLF    001177         529#    3897    3921    3929    3948    3953    3957    4012    4047
$DBLK    027100        4148     4182    4190#
$DEVCT   001212         541#
$DEVM    001260         577#
$DOAGN   024402        3602     3611    3617#
$DTBL    027070        4151     4186#
$ENDAD   024372         453     3613#   3916
$ENDCT   024340        1063     3604#
$ENDMG   024411        3606     3621#
$ENULL   024406        3609     3620#
$ENV     001222         546#    3823    3847    3899    3991
$ENVM    001223         547#    1086    3825    3993    3998
$EOP     024304        3584     3594#
$EOPCT   024332        1063*    3601#   3605
$ERFLG   001103         494#    3753    3781    3783    3789*   3811    3884*   3921
$ERMAX   001115         500#    1066*   3783    3806*   3811
$ERROR   025616        1057     3882#
$ERRPC   001116         501#    3891*   3892*   3893    3921    3935    4552    4554    4557    4560
$ERRTB   001264         595#    3943
$ERRTY   026020        3896     3928#
```

```
$ERTTL  001112          498#   3890*  3921
$ESCAP  001170          526#   1065*  3805*  3912   3914   3921
$ETABL  001222          545#
$ETEND  001264          480    579#
$FATAL  001204          538#   3851*
$FFLG   025614         3814*   3817*  3845   3854*  3862#
$FILLC  001156          519#   4016   4047
$FILLS  001155          518#   4047
$GDADR  001120          502#
$GDDAT  001124          504#   1190*  1198   1202*  1213*  1227*  1237*  1242   1255*  1264   1279*  1293   1310*
                       1323    1330*  1341*  1358   1368*  1377*  1392   1399*  1410*  1425   1432*  1442*  1457
                       1465*   1474*  1489   1497*  1506*  1521   1529*  1537*  1552   1560*  1568*  1583   1591*
                       1599*   1614   1622*  1630*  1645   1653*  1661*  1676   1684*  1691*  1703   1718*  1733
                       1741*   1749*  1764   1772*  1780*  1795   1803*  1811*  1826   1834*  1843*  1856   1871*
                       1886    1894*  1902*  1917   1925*  1934*  1949   1957*  1965*  1980   1988*  1996*  2011
                       2019*   2027*  2042   2050*  2058*  2073   2081*  2089*  2108   2116*  2125*  2131   2134*
                       2135    2145*  2158*  2169*  2180   2192*  2196   2200   2203*  2217*  2221   2225   2228*
                       2241*   2250   2253*  2267*  2274   2277*  2290*  2299   2302*  2316*  2320   2324   2327*
                       2339*   2346   2349*  2362*  2369   2372*  2385*  2392   2395*  2408*  2415   2418*  2431*
                       2438    2441*  2454*  2461   2464*  2477*  2484   2487*  2500*  2504   2508   2511*  2524*
                       2533    2537*  2550*  2557   2561*  2574*  2579   2586   2589*  2599*  2610   2613*  2626*
                       2637    2640*  2653*  2664   2667*  2680*  2691   2694*  2707*  2718   2721*  2734*  2745
                       2748*   2762*  2782   2787*  2796*  2815   2826*  2843   2859*  2878*  2898*  2917*  2944*
                       2950    2954*  2977*  2983   2987*  3011*  3019   3021   3029*  3031*  3062   3102   3116*
                       3125    3128*  3132*  3158*  3168   3202*  3218   3273*  3282   3296   3380*  3387*  3388
                       3518*   3532*  3533   3712*  3715*  3716*  3720*  3723   3725*  4552   4554   4557   4560
$GET42  024362         3610#
$GTSWR= ****** U       4221
$HD   = 000000          311
$HIBTS  001000          475#
$ICNT   001104          495#   3796*  3797   3799*  3810
$ILLUP  027310         4226    4242   4261#
$INTAG  001135          509#
$ITEMB  001114          499#   3893*  3901   3921   3932
$LF     001200          530#   3921   4047
$LFLG   025613         3855*   3861#
$LPADR  001106          496#   1067*  3787*  3803*  3808   3810
$LPERR  001110          497#   1068*  1187*  1256*  1282*  1311*  1344*  1380*  1413*  1445*  1477*  1509*  1540*
                       1571*   1602*  1633*  1664*  1721*  1752*  1783*  1814*  1874*  1905*  1937*  1968*  1999*
                       2030*   2061*  2093*  2190*  2215*  2239*  2265*  2288*  2314*  2337*  2360*  2383*  2406*
                       2429*   2452*  2475*  2498*  2522*  2548*  2572*  2597*  2624*  2651*  2678*  2705*  2732*
                       2760*   2805*  2932*  2965*  2998*  3040*  3080*  3118*  3131*  3149*  3193*  3289*  3369*
                       3496*   3787   3804*  3810   3911
$MADR1  001234          563#
$MADR2  001240          567#
$MADR3  001244          570#
$MADR4  001250          573#
$MAIL   001202          476     480    536#  1085   1186   1209   1235   1250   1276   1307   1340   1376   1409
                       1441    1473   1505   1536   1567   1598   1629   1660   1690   1717   1748   1779   1810
                       1842    1870   1901   1932   1964   1995   2026   2057   2088   2123   2143   2167   2189
                       2214    2238   2264   2287   2313   2336   2359   2382   2405   2428   2451   2474   2497
                       2521    2547   2571   2596   2623   2650   2677   2704   2731   2758   2795   2825   2852
                       2891    2930   2964   2997   3039   3079   3115   3146   3190   3242   3311   3463   3582
                       3802    3899   3991
$MAMS1  001232          557#
$MAMS2  001236          565#
```

```
$MAMS3   001242           568#
$MAMS4   001246           571#
$MBADR   001002           476#
$MFLG    025612          3815*   3821    3856*   3860#
$MSGAD   001216           543#   3831*   3834
$MSGLG   001220           544#   3836*
$MSGTY   001202           537#   3829    3837*   3849    3853*
$MTYP1   001233           558#
$MTYP2   001237           566#
$MTYP3   001243           569#
$MTYP4   001247           572#
$MXCNT   025346          3800    3810#
$NULL    001154           517#   4019    4047
$NWTST=  000001          1182#   1205#   1231#   1246#   1272#   1302#   1335#   1371#   1404#   1437#   1468#   1500#   1532#
                         1563#   1594#   1625#   1656#   1686#   1713#   1744#   1775#   1806#   1837#   1866#   1897#   1928#
                         1960#   1991#   2022#   2053#   2084#   2119#   2138#   2163#   2184#   2209#   2234#   2260#   2283#
                         2309#   2332#   2355#   2378#   2401#   2424#   2447#   2470#   2493#   2517#   2543#   2566#   2592#
                         2619#   2646#   2673#   2700#   2727#   2754#   2790#   2821#   2847#   2886#   2925#   2959#   2992#
                         3034#   3074#   3110#   3142#   3186#   3237#   3306#   3458#   3578#
$OCNT    026660          4079*   4108*   4121#
$OMODE   026662          4074*   4078*   4083    4086*   4097*   4123#
$OVER    025332          3764    3780    3788    3796    3807#
$PASS    001210           540#   1085*   3598*   3599*   3607    3620    3794    3811
$PASTM   001006           478#
$PWRAD   027304          4259#
$PWRDN   027144          1061    4226#   4254
$PWRMG   027300          4257#
$PWRUP   027216          4236    4242#
$QUES    001176           528#   3921    4047
$RDCHR=  ****** U        4222
$RDDEC=  ****** U        4222
$RDLIN=  ****** U        4222
$RDOCT=  ****** U        4222
$REGAD   001160           521#
$REGO    001162           523#
$RTNAD   024404          3619#
$R2A   = ****** U        4222
$SAVRE=  ****** U        4222
$SAVR6   027314          4235*   4243    4244*   4245*   4263#
$SCOPE   025072          1055    3762#
$SETUP=  000037          1046#   1054    1055    1057    1059    1061    1063    1064    1065    1067    3596    3763    3883
                         3909    3916
$STUP  = 177777          1046#
$SVLAD   025276          3772    3801#
$SVPC  = 000204           451#    456
$SWR   = 167400           301#    311     315     316     317     318     319     320     321     322     525     526     527
                         1064    1065    1067    1068    1186    1209    1235    1250    1276    1306    1339    1375    1408
                         1441    1472    1504    1536    1567    1598    1629    1660    1690    1717    1748    1779    1810
                         1841    1870    1901    1932    1964    1995    2026    2057    2088    2123    2142    2167    2188
                         2213    2238    2264    2287    2313    2336    2359    2382    2405    2428    2451    2474    2497
                         2521    2547    2570    2596    2623    2650    2677    2704    2731    2758    2794    2825    2851
                         2890    2929    2963    2996    3038    3078    3114    3146    3190    3241    3310    3462    3582
                         3591    3597    3612    3618    3620    3754    3755    3756    3757    3758    3763    3775    3777
                         3778    3781    3782    3783    3790    3791    3792    3804    3807    3810    3874    3875    3876
                         3877    3878    3887    3894    3906    3909    3921    4260
$SWREG   001224           548#   1088
```

```
$SWRMK= 000000       322     323     3758    3759    3779
$TESTN  001206       539#    1186*   1209*   1235*   1250*   1276*   1307*   1340*   1376*   1409*   1441*   1473*   1505*
                     1536*   1567*   1598*   1629*   1660*   1690*   1717*   1748*   1779*   1810*   1842*   1870*   1901*
                     1932*   1964*   1995*   2026*   2057*   2088*   2123*   2143*   2167*   2189*   2214*   2238*   2264*
                     2287*   2313*   2336*   2359*   2382*   2405*   2428*   2451*   2474*   2497*   2521*   2547*   2571*
                     2596*   2623*   2650*   2677*   2704*   2731*   2758*   2795*   2825*   2852*   2891*   2930*   2964*
                     2997*   3039*   3079*   3115*   3146*   3190*   3242*   3311*   3463*   3582*   3802*
$TIMES  001166       525#    1064*   1306*   1339*   1375*   1408*   1472*   1504*   1841*   2142*   2188*   2213*   2570*
                     2794*   2851*   2890*   2929*   2963*   2996*   3038*   3078*   3114*   3241*   3310*   3462*   3597*
                     3790*   3797    3800*   3810
$TKB    001146       514#
$TKS    001144       513#
$TMPO   001164       524#    1266*   1295*   1325*   1360*   1394*   1427*   1459*   1491*   1523*   1554*   1585*   1616*
                     1647*   1678*   1705*   1735*   1766*   1797*   1828*   1858*   1888*   1919*   1951*   1982*   2013*
                     2044*   2075*   2110*   2952*   2985*   3021*   3064*   3104*   3157*   3166    3172*   3201*   3222*
                     3363*   3400*   3401    3403*   3500*   3541*   3542    3544*   4554    4560
$TN   = 000113       301#    311     1182    1186#   1205    1209#   1226    1231    1235#   1243    1246    1250#   1272
                     1276#   1302    1306#   1307    1331    1335    1339#   1340    1366    1371    1375#   1376    1400
                     1404    1408#   1409    1433    1437    1441#   1468    1472#   1473    1500    1504#   1505    1532
                     1536#   1563    1567#   1594    1598#   1625    1629#   1656    1660#   1686    1690#   1708    1713
                     1717#   1744    1748#   1775    1779#   1806    1810#   1837    1841#   1842    1866    1870#   1897
                     1901#   1928    1932#   1960    1964#   1991    1995#   2022    2026#   2053    2057#   2084    2088#
                     2119    2123#   2138    2142#   2143    2163    2167#   2181    2184    2188#   2189    2204    2209
                     2213#   2214    2229    2234    2238#   2255    2260    2264#   2278    2283    2287#   2304    2309
                     2313#   2328    2332    2336#   2350    2355    2359#   2373    2378    2382#   2396    2401    2405#
                     2419    2424    2428#   2442    2447    2451#   2465    2470    2474#   2488    2493    2497#   2512
                     2517    2521#   2538    2543    2547#   2562    2566    2570#   2571    2592    2596#   2614    2619
                     2623#   2641    2646    2650#   2668    2673    2677#   2695    2700    2704#   2722    2727    2731#
                     2749    2754    2758#   2790    2794#   2795    2821    2825#   2844    2847    2851#   2852    2886
                     2890#   2891    2925    2929#   2930    2959    2963#   2964    2992    2996#   2997    3024    3034
                     3038#   3039    3069    3074    3078#   3079    3110    3114#   3115    3142    3146#   3149    3182
                     3186    3190#   3192    3233    3237    3241#   3242    3253    3306    3310#   3311    3413    3458
                     3462#   3463    3555    3578    3582#
$TPB    001152       516#    4036*   4047
$TPFLG  001157       520#    3985    4047
$TPS    001150       515#    4034    4047
$TRAP   027110       1059    4199#
$TRP  = 000005       4207#   4216#   4217#   4218#   4219#   4220#
$TRPAD  027132       4204    4214#
$TSTM   001004       477#
$TSTNM  001102       493#    1177*   3596*   3753    3779    3801*   3802    3807    3811    3883    3886    3921
$TYPBN= ****** U     4220
$TYPDS  026664       4136#   4219
$TYPE   026154       3842    3985#   4207    4215
$TYPEC  026366       4015    4022    4029    4034#   4035
$TYPEX  026434       4040    4042    4045#
$TYPOC  026462       4077#   4216
$TYPON  026476       4076    4079#   4218
$TYPOS  026436       4072#   4217
$UNIT   001214       542#
$UNITM  001010       479#
$USWR   001226       549#
$VECT1  001252       574#    2853    2892    3355    3493    3686
$VECT2  001254       575#
$XTSTR  025102       3766#
$$GET4= 000000       3612#
```

```
$DFILL  026661          4073*   4077*   4087    4122*
$4OCAT= ****** U        3763    3896
.     = 032344           438*    442*    451     452*    454*    456*    457*    464     465*    467*    469*    482*    490*
                         531    1052    1067    1068    3620    3624    3810    3811    3863*   3921    3967*   4047    4190*
                        4238    4262
.$ASTA= ****** U        3815    3818
.$X   = 001000           464*    469
```

```
COMMEN      1#    433#
ENDCOM      1#    433#
ERROR     327#   1200   1224   1229   1244   1267   1296   1326   1361   1395   1428   1460   1492   1524   1555
         1586    1617   1648   1679   1706   1736   1767   1798   1829   1859   1889   1920   1952   1983   2014
         2045    2076   2111   2133   2155   2160   2182   2202   2227   2252   2276   2301   2326   2348   2371
         2394    2417   2440   2463   2486   2510   2535   2559   2588   2612   2639   2656   2693   2720   2747
         2784    2817   2845   2870   2876   2883   2909   2915   2922   2953   2986   3022   3065   3105   3127
         3139    3170   3220   3300   3382   3390   3520   3535
ESCAPE      1#    433#
GETPRI      1#    433#
GETSWR      1#    433#
MULT        1#    433#
NEWTST      1#    433#   1182   1205   1231   1246   1272   1302   1335   1371   1404   1437   1468   1500   1532
         1563    1594   1625   1656   1686   1713   1744   1775   1806   1837   1866   1897   1928   1960   1991
         2022    2053   2084   2119   2138   2163   2184   2209   2234   2260   2283   2309   2332   2355   2378
         2401    2424   2447   2470   2493   2517   2543   2566   2592   2619   2646   2673   2700   2727   2754
         2790    2821   2847   2886   2925   2959   2992   3034   3074   3110   3142   3186   3237   3306   3458
         3578
POP         1#    433#   3857   3858   4177   4247   4248
PUSH        1#    433#   3818   3820   3841   4136   4228   4234
REPORT      1#    433#
SCOPE     328#   1185   1208   1234   1249   1275   1305   1338   1374   1407   1440   1471   1503   1535   1566
         1597    1628   1659   1689   1716   1747   1778   1809   1840   1869   1900   1931   1963   1994   2025
         2056    2087   2122   2141   2166   2187   2212   2237   2263   2286   2312   2335   2358   2381   2404
         2427    2450   2473   2496   2520   2546   2569   2595   2622   2649   2676   2703   2730   2757   2793
         2824    2850   2889   2928   2962   2995   3037   3077   3113   3145   3189   3240   3309   3461   3581
         3595
SETPRI      1#    433#
SETTRA   4207#   4216   4217   4218   4219
SETUP       1#    433#   1046
SKIP        1#    433#   1226   1243   1331   1366   1400   1433   1708   2181   2204   2229   2255   2278   2304
         2328    2350   2373   2396   2419   2442   2465   2488   2512   2538   2562   2614   2641   2668   2695
         2722    2749   2844   3024   3069   3148   3182   3192   3233   3253   3413   3555
SLASH       1#    433#
SPACE     433#
STARS       1#    433#    449    461    463    470    486    531    534   1182   1184   1205   1207   1231   1233
         1246    1248   1272   1274   1302   1304   1335   1337   1371   1373   1404   1406   1437   1439   1468
         1470    1500   1502   1532   1534   1563   1565   1594   1596   1625   1627   1656   1658   1686   1688
         1713    1715   1744   1746   1775   1777   1806   1808   1837   1839   1866   1868   1897   1899   1928
         1930    1960   1962   1991   1993   2022   2024   2053   2055   2084   2086   2119   2121   2138   2140
         2163    2165   2184   2186   2209   2211   2234   2236   2260   2262   2283   2285   2309   2311   2332
         2334    2355   2357   2378   2380   2401   2403   2424   2426   2447   2449   2470   2472   2493   2495
         2517    2519   2543   2545   2566   2568   2592   2594   2619   2621   2646   2648   2673   2675   2700
         2702    2727   2729   2754   2756   2790   2792   2821   2823   2847   2849   2886   2888   2925   2927
         2959    2961   2992   2994   3034   3036   3074   3076   3110   3112   3142   3144   3186   3188   3237
         3239    3306   3308   3458   3460   3578   3580   3588   3624   3626   3632   3634   3641   3643   3657
         3659    3673   3675   3681   3683   3694   3698   3706   3709   3730   3734   3750   3813   3870   3923
         3970    4049   4126   4193   4224   4240   4563   4566
SWRSU       1#    433#   1069#
TRMTRP   4207#
TYPBIN      1#    433#
TYPDEC      1#    433#   3607
TYPNAM      1#    433#
TYPNUM      1#    433#
TYPOCS      1#    433#
TYPOCT      1#    433#   3935   3959
```

```
TYPTXT      1#     433#
$$CMRE    484#     523
$$CMTM    484#     524
$$ESCA      1#     433#
$$NEWT      1#     433#    1182    1205    1231    1246    1272    1302    1335    1371    1404    1437    1468    1500    1532
         1563     1594     1625    1656    1686    1713    1744    1775    1806    1837    1866    1697    1928    1960    1991
         2022     2053     2084    2119    2138    2163    2184    2209    2234    2260    2283    2309    2332    2355    2378
         2401     2424     2447    2470    2493    2517    2543    2566    2592    2619    2646    2673    2700    2727    2754
         2790     2821     2847    2886    2925    2959    2992    3034    3074    3110    3142    3186    3237    3306    3458
         3578
$$SET    4207#    4216     4217    4218    4219
$$SETM   1085#
$$SKIP      1#     433#    1226    1243    1331    1366    1400    1433    1708    2181    2204    2229    2255    2278    2304
         2328     2350     2373    2396    2419    2442    2465    2488    2512    2538    2562    2614    2641    2668    2695
         2722     2749     2844    3024    3069    3148    3182    3192    3233    3253    3413    3555
.EQUAT      1#     301#     323
.HEADE      1#     301#
.KT11       1#
.SETUP      1#     301#    1046
.SWRHI      1#     301#     311
.SWRLO    323#
.$ACT1      1#     301#     447
.$APTB      1#     301#     532#
.$APTH      1#     301#     459
.$APTY      1#     301#    3811
.$ASTA      1#
.$CATC      1#     301#     436
.$CMTA      1#     301#     484
.$DB2D      1#
.$DB20      1#
.$DIV       1#
.$EOP       1#     301#    3586
.$ERRO      1#     301#    3868
.$ERRT      1#     301#    3921
.$MULT      1#
.$POWE      1#     301#    4222
.$RAND      1#
.$RDDE      1#
.$RDOC      1#
.$READ      1#
.$R2AZ      1#
.$SAVE      1#
.$SB2D      1#
.$SB20      1#
.$SCOP      1#     301#    3748
.$SIZE      1#
.$SUPR      1#
.$TRAP      1#     301#    4191
.$TYPB      1#
.$TYPD      1#     301#    4124
.$TYPE      1#     301#    3968
.$TYPO      1#     301#    4047
.$40CA      1#
.1170       1#
```

ADD	1096	1097	1107	1143	1167	1367	2134	3155	3181	3198	3232	3266	3267	3301	3391
	3392	3399	3405	3407	3536	3540	3546	3550	3688	3703	3716	3717	3720	3721	3828
	3840	3852	3943	4004	4075	4085	4156								
ASL	1157	1159	1160	3179	3230	3661	3662	3663	3664	3667	3670	3940	3941	3942	4203
ASLB	4161														
ASR	1164	1165	1166	3275	3278	3701	3702	3835							
BCC	3184	3235	3276	3279	4162										
BCS	3069	3182	3233												
BEQ	1087	1199	1220	1222	1243	1265	1294	1324	1359	1366	1393	1426	1458	1490	1522
	1553	1584	1615	1646	1677	1704	1734	1765	1796	1827	1857	1887	1918	1950	1981
	2012	2043	2074	2109	2132	2151	2153	2181	2201	2226	2251	2275	2300	2325	2347
	2370	2393	2416	2439	2462	2485	2509	2534	2558	2587	2611	2638	2665	2692	2719
	2746	2783	2816	2844	2881	2920	2951	2957	2984	2990	3020	3063	3103	3126	3138
	3148	3169	3192	3219	3253	3265	3299	3389	3534	3584	3611	3724	3778	3780	3782
	3786	3795	3822	3826	3846	3848	3885	3888	3910	3913	3945	3950	3963	3994	4007
	4042	4102													
BGE	3796														
BGT	3602	4109	4170												
BHI	1117	3027	3068	3784											
BHIS	1153														
BIC	1142	1197	1241	1263	1292	1322	1357	1391	1424	1456	1488	1520	1551	1582	1613
	1644	1675	1702	1709	1732	1763	1794	1825	1855	1862	1885	1916	1948	1979	2010
	2041	2072	2107	2130	2199	2224	2249	2273	2279	2298	2323	2345	2351	2368	2374
	2391	2397	2414	2420	2437	2443	2460	2466	2483	2489	2507	2532	2556	2581	2585
	2609	2615	2636	2642	2663	2669	2690	2696	2717	2723	2744	2750	2781	2869	2908
	3124	3137	3263	3291	3599	3676	3699	3700	4099						
BIS	1139	1349	2196	2206	2221	2231	2246	2257	2280	2295	2306	2320	2329	2352	2375
	2398	2421	2444	2467	2490	2504	2514	2529	2540	2563	2577	2579	2616	2643	2670
	2697	2724	2751	3255	3259	3277	3280	3372	3373	3510	3511	3530	3646	3651	3739
	3740	4104	4105	4164	4165										
BISB	3932														
BIT	1221	1225	2152	2156	2873	2880	2912	2919	2956	2989	3583	3723	3763	3777	3785
	3792	3887	3894	3909											
BITB	1086	3825	3993	3998	4030										
BLOS	3303														
BLT	4021	4110	4153	4169											
BMI	1331	1400	1433	2204	2229	2255	2278	2304	2328	2350	2373	2396	2419	2442	2465
	2488	2512	2538	2562	2614	2641	2668	2695	2722	2749	2819	3024	3254	4160	
BNE	1052	1076	1091	1099	1109	1121	1145	1169	1203	1226	1708	2136	2157	2205	2230
	2256	2305	2513	2539	2778	2867	2874	2906	2913	3108	3176	3227	3297	3371	3394
	3397	3402	3410	3509	3538	3543	3548	3553	3629	3691	3710	3719	3743	3764	3793
	3824	3830	3833	3850	3895	3900	3917	3933	3955	3992	3999	4001	4009	4017	4031
	4038	4100	4158	4246											
BPL	1270	1299	1329	1364	1398	1431	1463	1466	1495	1498	1527	1530	1558	1561	1589
	1592	1620	1623	1651	1654	1682	1685	1739	1742	1770	1773	1801	1804	1832	1835
	1861	1892	1895	1923	1926	1955	1958	1986	1989	2017	2020	2048	2051	2079	2082
	2114	2117	2590	2788	3130	3173	3223	3665	3668	3907	3986	4035	4098	4144	4174
BR	1078	1111	1132	1146	1161	1333	1369	1402	1435	1710	1863	2207	2232	2258	2281
	2307	2330	2353	2376	2399	2422	2445	2468	2491	2515	2541	2564	2617	2644	2671
	2698	2725	2752	2871	2877	2910	2916	3030	3032	3072	3178	3229	3268	3383	3413
	3521	3555	3679	3714	3766	3772	3775	3788	3791	3816	3838	3905	3938	3965	3988
	4014	4024	4033	4040	4076	4091	4112	4155	4172	4238	4262				
CLR	1050	1064	1065	1085	1100	1101	1126	1134	1147	1154	1213	1228	1239	1341	1342
	2145	2159	2865	2875	2878	2904	2914	2917	2937	2938	2940	2941	2970	2973	2974
	3003	3004	3006	3007	3010	3046	3047	3050	3052	3086	3087	3089	3092	3121	3131
	3132	3134	3140	3151	3194	3270	3292	3304	3320	3321	3330	3331	3341	3342	3351

	3352	3363	3374	3403	3500	3512	3544	3596	3597	3660	3689	3713	3790	3805	3931
	4089	4147	4150	4244											
CLRB	1177	3789	3854	3855	3856	4013	4039	4176							
CMP	1051	1075	1098	1108	1112	1116	1133	1144	1148	1152	1198	1242	1264	1293	1323
	1358	1365	1392	1425	1457	1489	1521	1552	1583	1614	1645	1676	1703	1707	1733
	1764	1795	1826	1856	1886	1917	1949	1980	2011	2042	2073	2108	2131	2135	2180
	2200	2225	2250	2274	2299	2324	2346	2369	2392	2415	2438	2461	2484	2508	2533
	2557	2586	2610	2637	2664	2691	2718	2745	2782	2815	2843	2872	2911	2950	2983
	3019	3026	3062	3067	3102	3107	3125	3168	3183	3218	3234	3264	3296	3298	3302
	3375	3376	3377	3378	3384	3388	3401	3513	3514	3515	3516	3522	3533	3542	3773
	3797	3916	4168												
CMPB	3779	3783	3823	3899	3991	4006	4008	4016	4037	4041					
DEC	1219	1332	1401	1590	1653	1771	1893	1894	2150	2203	2228	2511	2613	2777	2866
	2905	3023	3370	3508	3600	3628	3690	3718	3742	3939					
DECB	2818	4020	4023	4097	4109										
EMT	327														
HALT	442	3585	3678	3908	3918	3987	4237	4261							
INC	1110	1119	1287	1317	1352	1368	1386	1419	1451	1515	1546	1577	1639	1670	1697
	1727	1758	1789	1820	1850	1911	1943	1974	2005	2036	2067	2099	2102	2582	2774
	2806	2809	2811	2834	2837	2839	2954	2955	2987	2988	3025	3066	3106	3211	3214
	3285	3288	3368	3400	3505	3541	3598	3647	3652	3669	3741	3796	3853	3890	4103
	4111	4154	4245												
INCB	3801	3884	4043												
IOT	328														
JMP	446	483	1118	1122	1136	1155	3618								
JSR	1194	1217	1259	1285	1288	1318	1353	1387	1420	1452	1484	1516	1547	1578	1609
	1640	1671	1698	1728	1759	1790	1821	1851	1881	1912	1944	1975	2006	2037	2068
	2100	2103	2174	2177	2193	2197	2218	2222	2243	2247	2268	2271	2292	2296	2317
	2321	2340	2343	2363	2366	2386	2389	2410	2412	2433	2435	2456	2458	2479	2481
	2501	2505	2526	2530	2552	2554	2575	2580	2601	2603	2605	2628	2630	2632	2655
	2657	2659	2682	2684	2686	2709	2711	2713	2736	2738	2740	2775	2807	2812	2835
	2840	2863	2884	2902	2923	2946	2979	3014	3054	3055	3061	3094	3095	3101	3164
	3212	3215	3272	3283	3286	3293	3411	3506	3526	3554	3613	3648	3653	3842	3896
	3902	3996	4015	4022	4029										
MOV	1049	1053	1055	1056	1057	1058	1059	1060	1061	1062	1063	1067	1068	1071	1072
	1073	1074	1079	1081	1082	1083	1088	1093	1094	1095	1102	1103	1104	1113	1115
	1123	1124	1127	1128	1129	1130	1131	1135	1137	1138	1149	1151	1156	1162	1171
	1175	1176	1186	1187	1188	1189	1190	1191	1192	1193	1196	1209	1210	1211	1212
	1214	1215	1216	1223	1227	1235	1236	1237	1238	1240	1250	1251	1252	1253	1254
	1255	1256	1257	1258	1261	1262	1266	1276	1277	1278	1279	1280	1281	1282	1283
	1284	1290	1291	1295	1306	1307	1308	1309	1310	1311	1312	1313	1314	1315	1316
	1320	1321	1325	1339	1340	1343	1344	1345	1346	1347	1348	1350	1351	1355	1356
	1360	1375	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1389	1390	1394
	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1422	1423	1427	1441
	1442	1443	1444	1445	1446	1447	1448	1449	1450	1454	1455	1459	1472	1473	1474
	1475	1476	1477	1478	1479	1480	1481	1482	1483	1486	1487	1491	1504	1505	1506
	1507	1508	1509	1510	1511	1512	1513	1514	1518	1519	1523	1536	1537	1538	1539
	1540	1541	1542	1543	1544	1545	1549	1550	1554	1567	1568	1569	1570	1571	1572
	1573	1574	1575	1576	1580	1581	1585	1598	1599	1600	1601	1602	1603	1604	1605
	1606	1607	1608	1611	1612	1616	1629	1630	1631	1632	1633	1634	1635	1636	1637
	1638	1642	1643	1647	1660	1661	1662	1663	1664	1665	1666	1667	1668	1669	1673
	1674	1678	1690	1691	1692	1693	1694	1695	1696	1700	1701	1705	1717	1718	1719
	1720	1721	1722	1723	1724	1725	1726	1730	1731	1735	1748	1749	1750	1751	1752
	1753	1754	1755	1756	1757	1761	1762	1766	1779	1780	1781	1782	1783	1784	1785
	1786	1787	1788	1792	1793	1797	1810	1811	1812	1813	1814	1815	1816	1817	1818
	1819	1823	1824	1828	1841	1842	1843	1844	1845	1846	1847	1848	1853	1854	1858

	1870	1871	1872	1873	1874	1875	1876	1877	1878	1879	1880	1883	1884	1888	1901
	1902	1903	1904	1905	1906	1907	1908	1909	1910	1914	1915	1919	1932	1933	1934
	1935	1936	1937	1938	1939	1940	1941	1942	1946	1947	1951	1964	1965	1966	1967
	1968	1969	1970	1971	1972	1973	1977	1978	1982	1995	1996	1997	1998	1999	2000
	2001	2002	2003	2004	2008	2009	2013	2026	2027	2028	2029	2030	2031	2032	2033
	2034	2035	2039	2040	2044	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066
	2070	2071	2075	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2105
	2106	2110	2115	2123	2124	2125	2126	2127	2128	2129	2142	2143	2144	2146	2147
	2148	2149	2154	2158	2167	2168	2169	2170	2171	2172	2173	2176	2179	2188	2189
	2190	2191	2192	2195	2198	2213	2214	2215	2216	2217	2220	2223	2238	2239	2240
	2241	2242	2245	2248	2264	2265	2266	2267	2270	2272	2287	2288	2289	2290	2291
	2294	2297	2313	2314	2315	2316	2319	2322	2336	2337	2338	2339	2342	2344	2359
	2360	2361	2362	2365	2367	2382	2383	2384	2385	2388	2390	2405	2406	2407	2408
	2409	2413	2428	2429	2430	2431	2432	2436	2451	2452	2453	2454	2455	2459	2474
	2475	2476	2477	2478	2482	2497	2498	2499	2500	2503	2506	2521	2522	2523	2524
	2525	2528	2531	2547	2548	2549	2550	2551	2555	2570	2571	2572	2573	2574	2578
	2584	2596	2597	2598	2599	2600	2604	2607	2608	2623	2624	2625	2626	2627	2631
	2634	2635	2650	2651	2652	2653	2654	2658	2661	2662	2677	2678	2679	2680	2681
	2685	2688	2689	2704	2705	2706	2707	2708	2712	2715	2716	2731	2732	2733	2734
	2735	2739	2742	2743	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767	2768
	2769	2770	2771	2772	2773	2779	2780	2785	2786	2794	2795	2796	2797	2798	2799
	2800	2801	2802	2803	2804	2805	2810	2814	2825	2826	2827	2828	2829	2830	2831
	2832	2833	2838	2842	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861
	2862	2868	2882	2890	2891	2892	2893	2894	2895	2896	2897	2898	2899	2900	2901
	2907	2921	2929	2930	2932	2933	2934	2935	2936	2939	2942	2943	2944	2945	2949
	2952	2963	2964	2965	2966	2967	2968	2969	2971	2972	2975	2976	2977	2978	2982
	2985	2996	2997	2998	2999	3000	3001	3002	3005	3008	3009	3011	3012	3013	3017
	3018	3021	3028	3029	3031	3038	3039	3040	3041	3042	3043	3044	3045	3048	3049
	3051	3053	3058	3059	3060	3064	3070	3071	3078	3079	3080	3081	3082	3083	3084
	3085	3088	3090	3091	3093	3098	3099	3100	3104	3114	3115	3116	3117	3118	3119
	3120	3122	3123	3133	3136	3146	3149	3150	3152	3153	3154	3156	3157	3158	3160
	3161	3162	3163	3166	3167	3174	3177	3180	3190	3193	3195	3196	3197	3199	3200
	3201	3202	3204	3205	3206	3207	3208	3209	3210	3217	3224	3225	3228	3231	3241
	3242	3256	3257	3258	3260	3261	3262	3269	3271	3273	3274	3281	3282	3289	3290
	3294	3295	3310	3311	3312	3313	3314	3315	3316	3317	3318	3319	3322	3323	3324
	3325	3326	3327	3328	3329	3333	3334	3335	3336	3337	3338	3339	3340	3343	3344
	3345	3346	3347	3348	3349	3350	3353	3355	3356	3357	3358	3359	3360	3361	3362
	3364	3365	3366	3367	3369	3379	3380	3381	3385	3386	3387	3395	3398	3404	3406
	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471	3472	3473	3474	3475	3476
	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487	3488	3489	3490	3491
	3492	3493	3494	3495	3496	3497	3498	3499	3501	3502	3503	3504	3517	3518	3519
	3523	3524	3525	3528	3529	3531	3532	3539	3545	3549	3582	3603	3607	3610	3627
	3635	3636	3637	3638	3644	3645	3650	3666	3684	3685	3686	3687	3711	3712	3715
	3722	3725	3736	3737	3738	3768	3769	3771	3774	3787	3799	3800	3803	3804	3807
	3808	3819	3820	3827	3831	3836	3837	3839	3841	3851	3857	3858	3886	3891	3911
	3914	3930	3935	3944	3949	3954	3956	3960	3989	3990	3995	4003	4018	4072	4080
	4081	4082	4088	4095	4113	4114	4115	4116	4117	4137	4138	4139	4140	4141	4142
	4143	4148	4151	4171	4177	4178	4179	4180	4181	4183	4184	4199	4200	4204	4226
	4227	4228	4229	4230	4231	4232	4233	4234	4235	4236	4242	4243	4247	4248	4249
	4250	4251	4252	4253	4254	4255	4258								
MOVB	1066	3802	3806	3814	3815	3817	3883	3893	3901	4000	4028	4036	4073	4074	4077
	4078	4079	4083	4086	4087	4106	4146	4149	4163	4166	4175	4202			
NEG	4084	4145													
NOP	1849	2583	3614	3615	3616										
RESET	1174	1864	2161	2879	2918	2931	3135	3412	3612						
ROL	4090	4092	4093	4094	4096										

```
RTI       1080   3809   3920   4005   4118   4185   4260
RTS       3630   3639   3655   3671   3692   3704   3726   3744   3859   3958   4045   4205
SUB       1114   1150   1201   1202   1268   1269   1297   1298   1327   1328   1330   1362   1363   1396   1397
          1399   1429   1430   1432   1434   1461   1462   1464   1465   1493   1494   1496   1497   1525   1526
          1528   1529   1556   1557   1559   1560   1587   1588   1591   1618   1619   1621   1622   1649   1650
          1652   1680   1681   1683   1684   1737   1738   1740   1741   1768   1769   1772   1799   1800   1802
          1803   1830   1831   1833   1834   1890   1891   1921   1922   1924   1925   1953   1954   1956   1957
          1984   1985   1987   1988   2015   2016   2018   2019   2046   2047   2049   2050   2077   2078   2080
          2081   2112   2113   2116   2253   2254   2277   2302   2303   2327   2349   2372   2395   2418   2441
          2464   2487   2536   2537   2560   2561   2589   2640   2667   2694   2721   2748   2787   3128   3129
          3159   3171   3172   3203   3221   3222   3834   3892   4152
SWAB      1158   1163
TRAP      4207   4216   4217   4218   4219
TST       1090   1105   1106   1120   1125   1140   1141   1168   3147   3175   3191   3226   3252   3393   3396
          3409   3537   3547   3552   3770   3794   3829   3847   3849   3906   3912   3962   4002   4010   4032
          4101   4157   4167   4201
TSTB      1860   3781   3821   3832   3845   3985   4034   4159   4173
.ASCII     528    529
.ASCIZ     527    530   3621   3966   4264   4274   4285   4288   4291   4292   4296   4302   4305   4308   4311
          4315   4318   4322   4327   4332   4337   4341   4344   4347   4351   4355   4360   4365   4369   4374
          4378   4382   4387   4392   4397   4401   4404   4407   4409   4412   4416   4420   4424   4428   4431
          4433   4437   4441   4445   4448   4452   4455   4458   4462   4465   4469   4473   4478   4481   4485
          4488   4491   4494   4496   4499   4503   4507   4513   4520   4528   4536   4543
.BLKW     4190
.BYTE      493    494    499    500    508    509    517    518    519    520    546    547    557    558    565
           566    568    569    571    572   3620   3860   3861   3862   3903   3904   4119   4120   4121   4122
.ENABL       1    301
.END      4568
.ENDC      306    319    321    322    323    327    419    433    447    450    454    456    462    464    471
           487    491    493    521    524    525    526    527    528    532    535    557    565    568    571
           574    575    576    577    578    581   1046   1053   1054   1057   1059   1061   1063   1064   1065
          1067   1069   1090   1183   1184   1185   1186   1187   1206   1207   1208   1209   1210   1227   1232
          1233   1234   1235   1236   1244   1247   1248   1249   1250   1251   1273   1274   1275   1276   1277
          1303   1304   1305   1306   1307   1308   1332   1336   1337   1338   1339   1340   1341   1367   1372
          1373   1374   1375   1376   1377   1401   1405   1406   1407   1408   1409   1410   1434   1438   1439
          1440   1441   1442   1469   1470   1471   1472   1473   1474   1501   1502   1503   1504   1505   1506
          1533   1534   1535   1536   1537   1564   1565   1566   1567   1568   1595   1596   1597   1598   1599
          1626   1627   1628   1629   1630   1657   1658   1659   1660   1661   1687   1688   1689   1690   1691
          1709   1714   1715   1716   1717   1718   1745   1746   1747   1748   1749   1776   1777   1778   1779
          1780   1807   1808   1809   1810   1811   1838   1839   1840   1841   1842   1843   1867   1868   1869
          1870   1871   1898   1899   1900   1901   1902   1929   1930   1931   1932   1933   1961   1962   1963
          1964   1965   1992   1993   1994   1995   1996   2023   2024   2025   2026   2027   2054   2055   2056
          2057   2058   2085   2086   2087   2088   2089   2120   2121   2122   2123   2124   2139   2140   2141
          2142   2143   2144   2164   2165   2166   2167   2168   2182   2185   2186   2187   2188   2189   2190
          2205   2210   2211   2212   2213   2214   2215   2230   2235   2236   2237   2238   2239   2256   2261
          2262   2263   2264   2265   2279   2284   2285   2286   2287   2288   2305   2310   2311   2312   2313
          2314   2329   2333   2334   2335   2336   2337   2351   2356   2357   2358   2359   2360   2374   2379
          2380   2381   2382   2383   2397   2402   2403   2404   2405   2406   2420   2425   2426   2427   2428
          2429   2443   2448   2449   2450   2451   2452   2466   2471   2472   2473   2474   2475   2489   2494
          2495   2496   2497   2498   2513   2518   2519   2520   2521   2522   2539   2544   2545   2546   2547
          2548   2563   2567   2568   2569   2570   2571   2572   2593   2594   2595   2596   2597   2615   2620
          2621   2622   2623   2624   2642   2647   2648   2649   2650   2651   2669   2674   2675   2676   2677
          2678   2696   2701   2702   2703   2704   2705   2723   2728   2729   2730   2731   2732   2750   2755
          2756   2757   2758   2759   2791   2792   2793   2794   2795   2796   2822   2823   2824   2825   2826
          2845   2848   2849   2850   2851   2852   2853   2887   2888   2889   2890   2891   2892   2926   2927
          2928   2929   2930   2931   2960   2961   2962   2963   2964   2965   2993   2994   2995   2996   2997
```

	2998	3025	3035	3036	3037	3038	3039	3040	3070	3075	3076	3077	3078	3079	3080
	3111	3112	3113	3114	3115	3116	3143	3144	3145	3146	3147	3149	3183	3187	3188
	3189	3190	3191	3193	3234	3238	3239	3240	3241	3242	3243	3254	3307	3308	3309
	3310	3311	3312	3414	3459	3460	3461	3462	3463	3464	3556	3579	3580	3581	3582
	3583	3589	3590	3591	3593	3596	3602	3605	3606	3610	3612	3618	3620	3621	3624
	3625	3627	3633	3635	3642	3644	3658	3660	3674	3676	3682	3694	3695	3699	3707
	3710	3731	3735	3751	3754	3759	3763	3765	3776	3779	3780	3781	3783	3785	3792
	3796	3801	3803	3807	3810	3811	3814	3815	3818	3845	3860	3871	3874	3884	3891
	3896	3897	3898	3906	3916	3920	3921	3924	3939	3968	3971	4000	4050	4127	4194
	4200	4203	4215	4216	4217	4218	4219	4220	4221	4222	4225	4234	4235	4241	4247
	4248	4258	4260	4264	4564	4567									
.EQUIV	327	328	336	351	352	381	382	383	384	385	386	387	388	389	390
	409	410	411	412	413	414	415	416	417	418					
.EVEN	535	3863	3967	4551											
.IF	302	318	320	321	322	323	325	391	419	445	449	452	454	461	463
	470	486	490	492	521	524	525	526	527	531	532	534	557	565	568
	571	574	575	576	577	578	579	581	1046	1048	1053	1055	1057	1059	1061
	1063	1064	1065	1067	1085	1182	1184	1186	1205	1207	1209	1226	1231	1233	1235
	1243	1246	1248	1250	1272	1274	1276	1302	1304	1306	1307	1331	1335	1337	1339
	1340	1366	1371	1373	1375	1376	1400	1404	1406	1408	1409	1433	1437	1439	1441
	1468	1470	1472	1473	1500	1502	1504	1505	1532	1534	1536	1563	1565	1567	1594
	1596	1598	1625	1627	1629	1656	1658	1660	1686	1688	1690	1708	1713	1715	1717
	1744	1746	1748	1775	1777	1779	1806	1808	1810	1837	1839	1841	1842	1866	1868
	1870	1897	1899	1901	1928	1930	1932	1960	1962	1964	1991	1993	1995	2022	2024
	2026	2053	2055	2057	2084	2086	2088	2119	2121	2123	2138	2140	2142	2143	2163
	2165	2167	2181	2184	2186	2188	2189	2204	2209	2211	2213	2214	2229	2234	2236
	2238	2255	2260	2262	2264	2278	2283	2285	2287	2304	2309	2311	2313	2328	2332
	2334	2336	2350	2355	2357	2359	2373	2378	2380	2382	2396	2401	2403	2405	2419
	2424	2426	2428	2442	2447	2449	2451	2465	2470	2472	2474	2488	2493	2495	2497
	2512	2517	2519	2521	2538	2543	2545	2547	2562	2566	2568	2570	2571	2592	2594
	2596	2614	2619	2621	2623	2641	2646	2648	2650	2668	2673	2675	2677	2695	2700
	2702	2704	2722	2727	2729	2731	2749	2754	2756	2758	2790	2792	2794	2795	2821
	2823	2825	2844	2847	2849	2851	2852	2886	2888	2890	2891	2925	2927	2929	2930
	2959	2961	2963	2964	2992	2994	2996	2997	3024	3034	3036	3038	3039	3069	3074
	3076	3078	3079	3110	3112	3114	3115	3142	3144	3146	3148	3182	3186	3188	3190
	3192	3233	3237	3239	3241	3242	3253	3306	3308	3310	3311	3413	3458	3460	3462
	3463	3555	3578	3580	3582	3588	3589	3590	3591	3592	3593	3595	3601	3604	3606
	3610	3612	3618	3620	3621	3624	3626	3632	3634	3641	3643	3657	3659	3673	3675
	3681	3683	3694	3698	3706	3709	3730	3734	3750	3753	3758	3763	3775	3777	3778
	3779	3781	3782	3783	3792	3794	3802	3804	3809	3810	3811	3813	3815	3818	3845
	3860	3870	3873	3883	3887	3894	3896	3897	3899	3906	3909	3916	3920	3921	3923
	3938	3954	3970	3991	4049	4126	4193	4199	4203	4207	4216	4217	4218	4219	4220
	4221	4222	4224	4234	4235	4240	4247	4248	4256	4258	4260	4264	4563	4566	
.IFF	318	321	322	323	325	450	454	456	462	464	471	487	490	493	521
	532	535	1053	1183	1184	1185	1186	1206	1207	1208	1209	1227	1232	1233	1234
	1235	1244	1247	1248	1249	1250	1273	1274	1275	1276	1303	1304	1305	1306	1332
	1336	1337	1338	1339	1367	1372	1373	1374	1375	1401	1405	1406	1407	1408	1434
	1438	1439	1440	1441	1469	1470	1471	1472	1501	1502	1503	1504	1533	1534	1535
	1536	1564	1565	1566	1567	1595	1596	1597	1598	1626	1627	1628	1629	1657	1658
	1659	1660	1687	1688	1689	1690	1709	1714	1715	1716	1717	1745	1746	1747	1748
	1776	1777	1778	1779	1807	1808	1809	1810	1838	1839	1840	1841	1867	1868	1869
	1870	1898	1899	1900	1901	1929	1930	1931	1932	1961	1962	1963	1964	1992	1993
	1994	1995	2023	2024	2025	2026	2054	2055	2056	2057	2085	2086	2087	2088	2120
	2121	2122	2123	2139	2140	2141	2142	2164	2165	2166	2167	2182	2185	2186	2187
	2188	2205	2210	2211	2212	2213	2230	2235	2236	2237	2238	2256	2261	2262	2263
	2264	2279	2284	2285	2286	2287	2305	2310	2311	2312	2313	2329	2333	2334	2335

```
          2336  2351  2356  2357  2358  2359  2374  2379  2380  2381  2382  2397  2402  2403  2404
          2405  2420  2425  2426  2427  2428  2443  2448  2449  2450  2451  2466  2471  2472  2473
          2474  2489  2494  2495  2496  2497  2513  2518  2519  2520  2521  2539  2544  2545  2546
          2547  2563  2567  2568  2569  2570  2593  2594  2595  2596  2615  2620  2621  2622  2623
          2642  2647  2648  2649  2650  2669  2674  2675  2676  2677  2696  2701  2702  2703  2704
          2723  2728  2729  2730  2731  2750  2755  2756  2757  2758  2791  2792  2793  2794  2822
          2823  2824  2825  2845  2848  2849  2850  2851  2887  2888  2889  2890  2926  2927  2928
          2929  2960  2961  2962  2963  2993  2994  2995  2996  3025  3035  3036  3037  3038  3070
          3075  3076  3077  3078  3111  3112  3113  3114  3143  3144  3145  3146  3149  3183  3187
          3188  3189  3190  3193  3234  3238  3239  3240  3241  3254  3307  3308  3309  3310  3414
          3459  3460  3461  3462  3556  3579  3580  3581  3582  3589  3592  3596  3602  3605  3620
          3625  3627  3633  3635  3642  3644  3658  3660  3674  3676  3682  3684  3695  3699  3707
          3710  3731  3735  3751  3776  3779  3780  3783  3810  3811  3814  3871  3873  3887  3916
          3921  3924  3939  3968  3971  4050  4127  4194  4200  4225  4241  4256  4564  4567
.IFT      3791  3897
.IFTF     3789  3896
.IIF       301   306   311   315   316   317   319   322   323   442   531   535  1054  1057  1063
          1064  1065  1067  1068  3590  3596  3597  3608  3620  3624  3754  3755  3756  3757  3758
          3759  3763  3790  3791  3807  3810  3811  3874  3875  3876  3877  3878  3883  3909  3916
          3921  3936  3961  4047  4215  4216  4217  4218  4219
.IRP      1046  1182  1186  1205  1209  1231  1235  1246  1250  1272  1276  1302  1307  1335  1340
          1371  1376  1404  1409  1437  1441  1468  1473  1500  1505  1532  1536  1563  1567  1594
          1598  1625  1629  1656  1660  1686  1690  1713  1717  1744  1748  1775  1779  1806  1810
          1837  1842  1866  1870  1897  1901  1928  1932  1960  1964  1991  1995  2022  2026  2053
          2057  2084  2088  2119  2123  2138  2143  2163  2167  2184  2189  2209  2214  2234  2238
          2260  2264  2283  2287  2309  2313  2332  2336  2355  2359  2378  2382  2401  2405  2424
          2428  2447  2451  2470  2474  2493  2497  2517  2521  2543  2547  2566  2571  2592  2596
          2619  2623  2646  2650  2673  2677  2700  2704  2727  2731  2754  2758  2790  2795  2821
          2825  2847  2852  2886  2891  2925  2930  2959  2964  2992  2997  3034  3039  3074  3079
          3110  3115  3142  3146  3186  3190  3237  3242  3306  3311  3458  3463  3578  3582  3819
          3820  3841  3857  3858  3883  4137  4177  4228  4234  4247  4248
.LIST        1   301   322   433   442   521   523   524   525   532   535  1046  1069  1182  1186
          1205  1209  1231  1235  1246  1250  1272  1276  1302  1306  1335  1339  1371  1375  1404
          1408  1437  1441  1468  1472  1500  1504  1532  1536  1563  1567  1594  1598  1625  1629
          1656  1660  1686  1690  1713  1717  1744  1748  1775  1779  1806  1810  1837  1841  1866
          1870  1897  1901  1928  1932  1960  1964  1991  1995  2022  2026  2053  2057  2084  2088
          2119  2123  2138  2142  2163  2167  2184  2188  2209  2213  2234  2238  2260  2264  2283
          2287  2309  2313  2332  2336  2355  2359  2378  2382  2401  2405  2424  2428  2447  2451
          2470  2474  2493  2497  2517  2521  2543  2547  2566  2570  2592  2596  2619  2623  2646
          2650  2673  2677  2700  2704  2727  2731  2754  2758  2790  2794  2821  2825  2847  2851
          2886  2890  2925  2929  2959  2963  2992  2996  3034  3038  3074  3078  3110  3114  3142
          3146  3186  3190  3237  3241  3306  3310  3458  3462  3578  3582  3596  3612  3758  3916
          4207  4215  4216  4217  4218  4219  4220
.MACRO       1   323   484  1085  4207
.MCALL     301   433   532  1069
.MEXIT     580
.NLIST       1   301   322   433   442   521   523   524   525   532   535  1046  1069  1182  1186
          1205  1209  1231  1235  1246  1250  1272  1276  1302  1306  1335  1339  1371  1375  1404
          1408  1437  1441  1468  1472  1500  1504  1532  1536  1563  1567  1594  1598  1625  1629
          1656  1660  1686  1690  1713  1717  1744  1748  1775  1779  1806  1810  1837  1841  1866
          1870  1897  1901  1928  1932  1960  1964  1991  1995  2022  2026  2053  2057  2084  2088
          2119  2123  2138  2142  2163  2167  2184  2188  2209  2213  2234  2238  2260  2264  2283
          2287  2309  2313  2332  2336  2355  2359  2378  2382  2401  2405  2424  2428  2447  2451
          2470  2474  2493  2497  2517  2521  2543  2547  2566  2570  2592  2596  2619  2623  2646
          2650  2673  2677  2700  2704  2727  2731  2754  2758  2790  2794  2821  2825  2847  2851
          2886  2890  2925  2929  2959  2963  2992  2996  3034  3038  3074  3078  3110  3114  3142
```

```
          3146   3186   3190   3237   3241   3306   3310   3458   3462   3578   3582   3596   3612   3758   3916
          4207   4215   4216   4217   4218   4219   4220
.PAGE      484    581    596   3586   3624   3748
.REM         1
.REPT      442    523    524
.SBTTL     311    323    436    445    447    459    484    532    581   1047   1178   1179   1180   1182   1205
          1231   1246   1272   1302   1335   1371   1404   1437   1468   1500   1532   1563   1594   1625   1656
          1686   1713   1744   1775   1806   1837   1866   1897   1928   1960   1991   2022   2053   2084   2119
          2138   2163   2184   2209   2234   2260   2283   2309   2332   2355   2378   2401   2424   2447   2470
          2493   2517   2543   2566   2592   2619   2646   2673   2700   2727   2754   2790   2821   2847   2886
          2925   2959   2992   3034   3074   3110   3142   3186   3237   3306   3458   3578   3586   3748   3811
          3868   3921   3968   4047   4124   4191   4207   4222   4272
.TITLE     301
.WORD      442    443    444    455    475    476    477    478    479    480    492    495    496    497    498
           501    502    503    504    505    506    507    510    511    512    521    523    524    537    538
           539    540    541    542    543    544    548    549    550    563    567    570    573    574    575
           576    577    578   3601   3604   3619   3843   3947   3952   3997   4044   4123   4257   4259   4552
          4554   4557   4560


 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 *,DZVSCB.SEQ/SOL/CRF/PAGNUM/NL:TOC=SYSMAC.CD,DZVSCB.P11
 RUN-TIME: 55 76 10 SECONDS
 RUN-TIME RATIO: 464/143=3.2
 CORE USED:  34K  (67 PAGES)
```