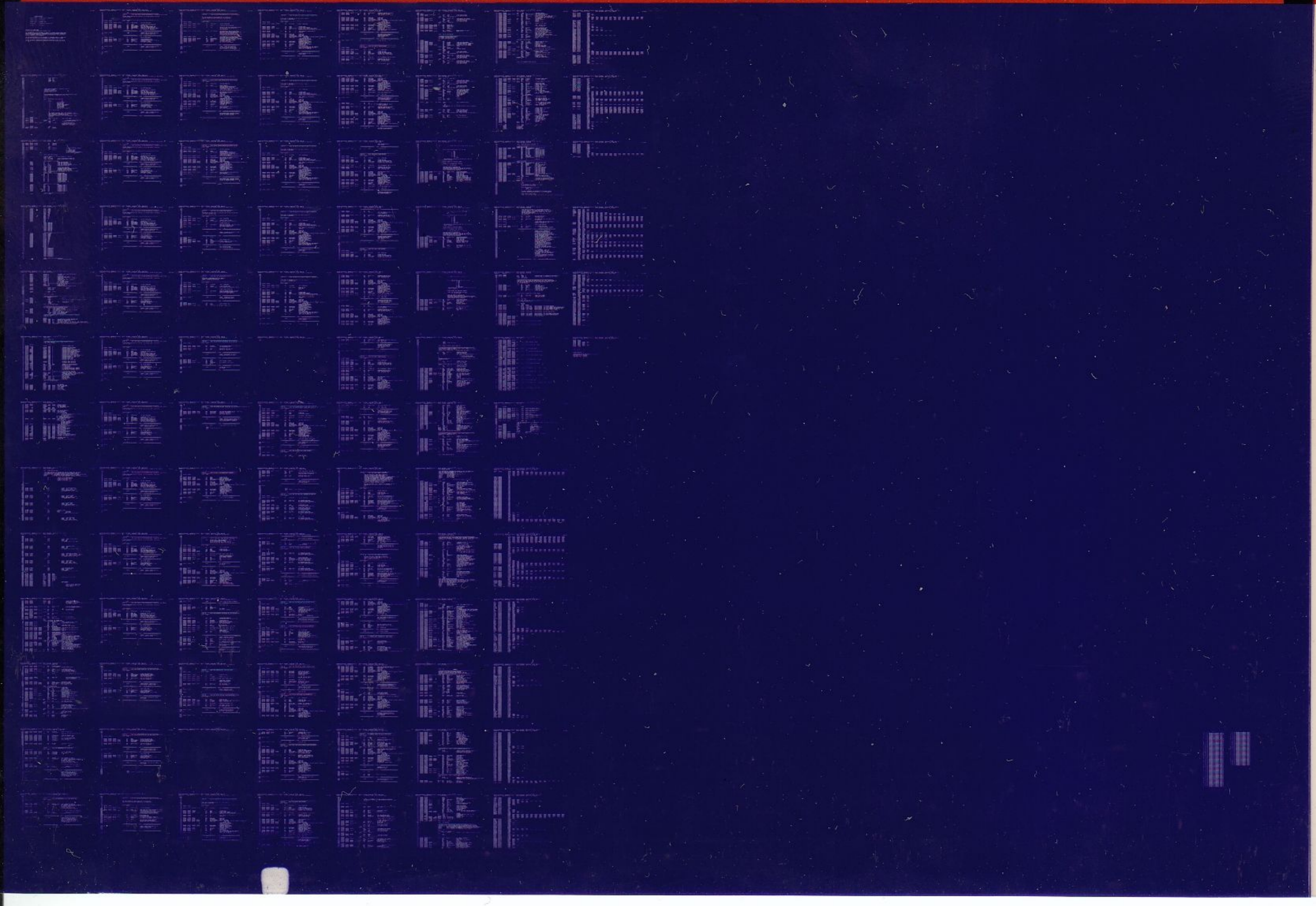
KWV11

KWV11A DIAGNOSTIC
CNKWAAO

AH-T452A-MC
FICHE 1 OF 1

MAY 1983
COPYRIGHT® 82-83
MADE IN USA



IDENTIFICATION

PRODUCT NAME: CNKWAA0 KWV11A DIAGNOSTIC

PRODUCT CODE: AC-T451A-MC

PRODUCT DATE: DECEMBER,1982

MAINTAINER: DIAGNOSTICS SERVICES/ISS

AUTHOR: RAY SHOOP

COPYRIGHT (C) 1982,1983
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

```
                                        .NLIST  MC,MD,CND
                                        .LIST   ME
                                        .ENABL  ABS
                                        .ENABL  AMA
            167400                      $SWR=   167400


                        .TITLE  KWV11A DISAGNOSTIC  MAINDEC-11-CNKWA-A
                        ;*COPYRIGHT (C) 1982
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
            000001      $TN=1

                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        11            INHIBIT ITERATIONS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            LOOP ON TEST IN SWR<7:0>

                                        .SBTTL  TRAP CATCHER

            000000                      .=0
                                        ;*ALL UNUSED LOCATIONS FROM 4-776 CONTAIN A ".+2"
                                        ;*AND "JSR PC,R0" SEQUENCE TO CATCH ILLEGAL INTERRUPTS.
                                        ;*AND INTERRUPTS TO THE WRONG VECTOR.
                                        ;*LOCATION 0 CONTAINS A 0 TO CATCH IMPROPERLY LOADED
                                        ;*VECTORS.
            000004                      .=4
000004  017102  000200                  .WORD   IOTRD,200       ;HANDLE BUSS ERROR.
            000174                      .=174
000174  000000              DISPREG:    .WORD   0                       ;;SOFTWARE DISPLAY REGISTER.
000176  000000              SWREG:      .WORD   0               ;;SOFTWARE SWITCH REGISTER.
            000100                      .=100
000100  000104  000200  000002          .WORD   104,200,2       ;IF 'B EVENT'ON Q-BUS IS
                                                                ;CONNECTED,WE NEED A WAY OF
                                                                ;IGNORING ITS INTERRUPTS.

            000200                      .=200
000200  000137  001506                  JMP     @#START
000204  000137  001444                  JMP     @#QSTART
```

```
 5762  000210  000137  014046                  JMP     @#OITST1
 5763  000214  000137  014136                  JMP     @#OITST2
 5764  000220  000137  014216                  JMP     @#OITST3
 5765
 5766          000230                          .=230
 5767  000230  000137  001472                  JMP     @#WSTART                ;WESTFIELD STARTING ADDRESS
 5768          000240                          .=240
 5769  000240  000137  001456                  JMP     @#TSTSTR                ;ALL TESTER TESTS
 5770                                                                           ;IF STARTED HERE.
 5771                                                                           ;ALLOWS PRODUCTION TO EXERCISE
 5772                                                                           ;UP TO 16 CLOCKS.NORMAL=4.
 5773
 5774                                  .SBTTL  BASIC DEFINITIONS
  (1)
  (1)                                  ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
  (1)           001100                 STACK=  1100
  (1)                                  .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
  (1)                                  .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
  (1)
  (1)                                  ;*MISCELLANEOUS DEFINITIONS
  (1)           000011                 HT=     11              ;;CODE FOR HORIZONTAL TAB
  (1)           000012                 LF=     12              ;;CODE FOR LINE FEED
  (1)           000015                 CR=     15              ;;CODE FOR CARRIAGE RETURN
  (1)           000200                 CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
  (1)           177776                 PS=     177776          ;;PROCESSOR STATUS WORD
  (1)                                  .EQUIV  PS,PSW
  (1)           177774                 STKLMT= 177774          ;;STACK LIMIT REGISTER
  (1)           177772                 PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
  (1)           177570                 DSWR=   177570          ;;HARDWARE SWITCH REGISTER
  (1)           177570                 DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
  (1)                                  ;*****  THE FOLLOWING ODT START ADDRESS FOR SBC 11/21 IS ADDED
  (1)           170000                 ODTST=  170000
  (1)                                  ;*GENERAL PURPOSE REGISTER DEFINITIONS
  (1)           000000                 R0=     %0              ;;GENERAL REGISTER
  (1)           000001                 R1=     %1              ;;GENERAL REGISTER
  (1)           000002                 R2=     %2              ;;GENERAL REGISTER
  (1)           000003                 R3=     %3              ;;GENERAL REGISTER
  (1)           000004                 R4=     %4              ;;GENERAL REGISTER
  (1)           000005                 R5=     %5              ;;GENERAL REGISTER
  (1)           000006                 R6=     %6              ;;GENERAL REGISTER
  (1)           000007                 R7=     %7              ;;GENERAL REGISTER
  (1)           000006                 SP=     %6              ;;STACK POINTER
  (1)           000007                 PC=     %7              ;;PROGRAM COUNTER
  (1)
  (1)                                  ;*PRIORITY LEVEL DEFINITIONS
  (1)           000000                 PR0=    0               ;;PRIORITY LEVEL 0
  (1)           000040                 PR1=    40              ;;PRIORITY LEVEL 1
  (1)           000100                 PR2=    100             ;;PRIORITY LEVEL 2
  (1)           000140                 PR3=    140             ;;PRIORITY LEVEL 3
  (1)           000200                 PR4=    200             ;;PRIORITY LEVEL 4
  (1)           000240                 PR5=    240             ;;PRIORITY LEVEL 5
  (1)           000300                 PR6=    300             ;;PRIORITY LEVEL 6
  (1)           000340                 PR7=    340             ;;PRIORITY LEVEL 7
  (1)
  (1)                                  ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
  (1)           100000                 SW15=   100000
```

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

```
   (1)          000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
   (1)          000014                  TBITVEC=14              ;;"T" BIT
   (1)          000014                  TRTVEC= 14              ;;TRACE TRAP
   (1)          000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
   (1)          000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
   (1)          000024                  PWRVEC= 24              ;;POWER FAIL
   (1)          000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
   (1)          000034                  TRAPVEC=34              ;;"TRAP" TRAP
   (1)          000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
   (1)          000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
   (1)                                  ;*****  THE FOLLOWING BREAK VECTOR AND LINE CLOCK VECTOR ARE INCLUDED
   (1)          000100                  LKVEC=  100             ;;LINE CLOCK VECTOR
   (1)          000140                  BRKVEC= 140             ;;BREAK VECTOR
   (1)          000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
 5775
 5776          174420                          ABASE=  174420                  ;VER:0
 5777          000240                          AVECT1= 240                     ;VER:0
 5778          000200                          APRIOR= 200
 5779
 5780          167400                          $SWR=   167400
 5781          000001                          $TN=    1
 5782
 5783                                  .SBTTL  ACT11 HOOKS
   (1)
   (2)                                  ;;**************************************************************
   (1)                                  ;HOOKS REQUIRED BY ACT11
   (1)          000244                          $SVPC=.                 ;SAVE PC
   (1)          000046                          .=46
   (1)  000046  013720                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
   (1)          000052                          .=52
   (1)  000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
   (1)          000244                          .=$SVPC                 ;; RESTORE PC
 5784          001000                          .=1000
 5785                                  .SBTTL  APT PARAMETER BLOCK
   (1)
   (2)                                  ;;**************************************************************
   (1)                                  ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
   (2)                                  ;;**************************************************************
   (1)          001000                          .$X=.   ;;SAVE CURRENT LOCATION
   (1)          000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
   (1)  000024  000200                          200     ;;FOR APT START UP
   (1)          000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
   (1)  000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
   (1)          001000                          .=.$X   ;;RESET LOCATION COUNTER
   (2)                                  ;;**************************************************************
   (1)                                  ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
   (1)                                  ;INTERFACE SPEC.
   (1)
   (1)  001000                          $APTHD:
   (1)  001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
   (1)  001002  001174                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
   (1)  001004  000002                  $TSTM:  .WORD   2       ;;RUN TIM OF LONGEST TEST
   (1)  001006  000170                  $PASTM: .WORD   120.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
   (1)  001010  000170                  $UNITM: .WORD   120.    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
   (1)  001012  000031                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
5786                                    .SBTTL  COMMON TAGS
 (1)
 (2)                                    ;;**********************************************************************
 (1)                                    ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
 (1)                                    ;*USED IN THE PROGRAM.
 (1)
 (1)                 001100                     .=1100
 (1)  001100                            $CMTAG:                         ;;START OF COMMON TAGS
 (1)  001100  000000                            .WORD   0
 (1)  001102     000                    $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
 (1)  001103     000                    $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
 (1)  001104  000000                    $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
 (1)  001106  000000                    $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
 (1)  001110  000000                    $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
 (1)  001112  000000                    $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
 (1)  001114     000                    $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
 (1)  001115     001                    $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
 (1)  001116  000000                    $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
 (1)  001120  000000                    $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
 (1)  001122  000000                    $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
 (1)  001124  000000                    $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
 (1)  001126  000000                    $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
 (1)  001130  000000                            .WORD   0               ;;RESERVED--NOT TO BE USED
 (1)  001132  000000                            .WORD   0
 (1)  001134     000                    $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
 (1)  001135     000                    $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
 (1)  001136  000000                            .WORD   0
 (1)  001140  177570                    $SWR:   .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
 (1)  001142  177570                    DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
 (1)  001144  177560                    $TKS:   177560                  ;;TTY KBD STATUS
 (1)  001146  177562                    $TKB:   177562                  ;;TTY KBD BUFFER
 (1)  001150  177564                    $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
 (1)  001152  177566                    $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
 (1)  001154     000                    $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
 (1)  001155     002                    $FILLS: .BYTE   2
 (1)                                            ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
 (1)  001156     012                    $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
 (1)  001157     000                    $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
 (1)  001160  000000                    $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
 (1)  001162  000000                    $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
 (1)  001164  177607  000377            $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
 (1)  001170     077                    $QUES:  .ASCII  /?/             ;;QUESTION MARK
 (1)  001171     015                    $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
 (1)  001172  000012                    $LF:    .ASCIZ  <12>            ;;LINE FEED
 (2)                                    ;;**********************************************************************
 (2)                                    .SBTTL  APT MAILBOX-ETABLE
 (2)
 (3)                                    ;;**********************************************************************
 (2)                                    .EVEN
 (2)  001174                            $MAIL:                          ;;APT MAILBOX
 (2)  001174  000000                    $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
 (2)  001176  000000                    $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
 (2)  001200  000000                    $TESTN: .WORD   ATESTN          ;;TEST NUMBER
 (2)  001202  000000                    $PASS:  .WORD   APASS           ;;PASS COUNT
 (2)  001204  000000                    $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
 (2)  001206  000000                    $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
```

```
(2)  001210  000000            $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
(2)  001212  000000            $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
(2)  001214                    $ETABLE:                ;;APT ENVIRONMENT TABLE
(2)  001214      000           $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
(2)  001215      000           $ENVM:  .BYTE   AENVM
(2)                            ;;ENVIRONMENT MODE BITS
(2)  001216  000000            $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
(2)  001220  000000            $USWR:  .WORD   AUSWR   ;;USER SWITCHES
(2)  001222  000000            $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
(2)                            ;*                      BITS 15-11=CPU TYPE
(2)                            ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(2)                            ;*                              11/70=06,PDQ=07,Q=10
(2)                            ;*                      BIT 10=REAL TIME CLOCK
(2)                            ;*                      BIT  9=FLOATING POINT PROCESSOR
(2)                            ;*                      BIT  8=MEMORY MANAGEMENT
(2)  001224      000           $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
(2)  001225      000           $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
(2)                            ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
(2)                            ;*                              900 NSEC CORE=001
(2)                            ;*                              300 NSEC BIPOLAR=002
(2)                            ;*                              500 NSEC MOS=003
(2)  001226  000000            $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
(2)                            ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
(2)  001230      000           $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
(2)  001231      000           $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
(2)  001232  000000            $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
(2)  001234      000           $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
(2)  001235      000           $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
(2)  001236  000000            $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
(2)  001240      000           $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
(2)  001241      000           $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
(2)  001242  000000            $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
(2)  001244  000240            $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)  001246  000000            $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(2)  001250  174420            $BASE:  .WORD   ABASE
(2)                                    ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)  001252  000000            $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
(2)  001254  000000            $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
(2)  001256                    $ETEND:
(2)                            .MEXIT
```

```
   (1)                          .SBTTL  ERROR POINTER TABLE
   (1)
   (1)                          ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
   (1)                          ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
   (1)                          ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
   (1)                          ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
   (1)                          ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
   (1)
   (1)                          ;*      EM              ;;POINTS TO THE ERROR MESSAGE
   (1)                          ;*      DH              ;;POINTS TO THE DATA HEADER
   (1)                          ;*      DT              ;;POINTS TO THE DATA
   (1)                          ;*      DF              ;;POINTS TO THE DATA FORMAT
   (1)
   (1)
   (1)  001256                  $ERRTB:
 5787
 5792
 5793                           ;ITEM   1
 5794
 5795  001256  017232                  EM1                     ;CLOCK   SR FUNCTION ERROR
 5796  001260  017555                  DH1                     ;ERRPC   ASR     WAS     S/B
 5797  001262  017764                  DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
 5798  001264  020062                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
   (1)
 5799
 5800                           ;ITEM   2
 5801
 5802  001266  017264                  EM2                     ;CLOCK   SR DATA ERROR
 5803  001270  017555                  DH1                     ;ERRPC   ASR     WAS     S/B
 5804  001272  017764                  DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
 5805  001274  020062                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
   (1)
 5806
 5807                           ;ITEM   3
 5808
 5809  001276  017312                  EM3                     ;CLOCK  BR DATA ERROR
 5810  001300  017601                  DH3                     ;ERRPC   ABR     WAS
 5811  001302  017776                  DT3                     ;$ERRPC,ABR,$BDDAT,$GDDAT
 5812  001304  020062                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
   (1)
 5813
 5814                           ;ITEM   4
 5815
 5816  001306  017340                  EM4             ;INTERRUPT ERROR.
 5817  001310  017625                  DH4A            ;ERRPC   TO      ROM ADDR.
 5818  001312  020010                  DT4             ;$ERRPC,        TRTO,TRFRO
 5819  001314  020062                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
   (1)
 5820
 5821                           ;ITEM   5
 5822
 5823  001316  017361                  EM5                     ;CLOCK   COUNT REG ERROR
 5824  001320  017555                  DH1                     ;ERRPC   ASR   WAS   S/B
 5825  001322  017764                  DT1                     ;$ERRPC, ACR, $BDDAT, $GDDAT
 5826  001324  020062                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
   (1)
```

```
5827
5828                                    ;ITEM   6
5829
5830  001326  017423                            EM12                    ;CLOCK   COUNT FUNCTION ERROR
5831  001330  017661                            DH12                    ;ERRPC   ASR
5832  001332  020020                            DT12                    ;ERRPC, ASR
5833  001334  020062                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
5834
5835                                    ;ITEM   7
5836
5837  001336  017452                            EM16                    ;CLOCK   INTERRUPT ERROR
5838  001340  017661                            DH12                    ;ERRPC   ASR
5839  001342  020020                            DT12                    ;$ERRPC, ASR
5840  001344  020062                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
5841
5842                                    ;ITEM   10
5843
5844  001346  017503                            EM20                    ;CLOCK   REPEATABILITY ERROR
5845  001350  017676                            DH20                    ;ERROR   ASR   2ND CNT 1ST CNT  3RD CNT
5846  001352  020026                            DT20                    ;$ERRPC, ASR, $BDDAT, $GDDAT, $TMP0
5847  001354  020062                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
5848
5849                                    ;ITEM   11
5850
5851  001356  017404                            EM11                    ;CLOCK   COUNT ERROR
5852  001360  017555                            DH1                     ;ERRPC   ASR   WAS   S/B
5853  001362  020042                            DT22                    ;$ERRPC, ASR, $BDDAT, $TMP0
5854  001364  020062                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
5855
5856                                    ;ITEM   12
5857
5858  001366  017532                            EM26                    ;CLOCK ADDRESSING ERROR
5859  001370  017737                            DH26                    ;ERRPC   CLOCK ADDR.
5860  001372  020054                            DT26                    ;$ERRPC,$TMP0
5861  001374  020062                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
5862
5863  001376  174420                    ASR:    .WORD   ABASE
5864  001400  174422                    ABR:    .WORD   ABASE+2
5865  001402  000240                    VECT1:  .WORD   AVECT1
5866  001404  000242                    VECTP:  .WORD   AVECT1+2
5867  001406  000244                    VECT2:  .WORD   AVECT1+4        ;VECTOR ADDR. OF ST2 INTRS.
5868  001410  000246                    VECT2P: .WORD   AVECT1+6
5869  001412  000200                    PRIOR:  .WORD   APRIOR
5870  001414  167774                    DR:     .WORD   167774
5871  001416  167772                    DR2:    .WORD   167772
5872  001420  000000                    $TMP0:  .WORD   0               ;TEMP STORAGE.
5873  001422  000000                    $TMP1:  .WORD   0               ;TMP STORAGE.
5874  001424  000000                    $TMP3:  .WORD   0
5875  001426  000000                    ROTATE: .WORD   0                       ;POINT TO DEVICE UNDER TEST.
5876  001430  000000                    UTEST:  .WORD   0                       ;KEEPS TRACK OF GOOD UNITS.
5877  001432  000000                    ERCNT:  .WORD   0                       ;COUNTS ERRORS.
```

```
 5878  001434  000000                   MDEVCT: .WORD   0                       ;COUNTS DEVICES TESTED.
 5879  001436  000C00                   TSTCNT: .WORD   0                       ;MAX DEVICES TO BE TESTED.
 5880  001440  000000                   EXS:    .WORD   0                       ;=0, NORMAL: =1 SPECIAL TESTOR START, BY L+S @ 2
 5881  001442  000000                   LCNT:   .WORD   0                       ;TOTAL UNITS TESTED.
 5882
 5883
 5885  001444  012737  002144  001420   QSTART: MOV     #RSTART,$TMP0           ;LOAD SETUP RETURN ADDRESS
 5886  001452  000137  001526           JMP     INIT                            ;INIT THE PROGRAM VECTOR SPACE
 5887
 5888  001456  005237  001440           TSTSTR: INC     EXS                     ;SET FOR TESTOR.
 5889  001462  012737  000020  001436           MOV     #16.,TSTCNT             ;ALLOW 16 UNITS
 5890  001470  000413                   BR      1$
 5891          001472                   WSTART=.
 5892  001472  012737  000020  001436           MOV     #16.,TSTCNT     ;TEST UP TO 16 UNITS.
 5893  001500  005037  001440           CLR     EXS
 5894  001504  000405                   BR      1$
 5895          001506                   START=.
 5896  001506  012737  000004  001436           MOV     #4,TSTCNT       ;TEST UP TO FOUR UNITS.
 5897  001514  005037  001440           CLR     EXS
 5898  001520  012737  001774  001420   1$:     MOV     #ZSTART,$TMP0           ;LOAD SETUP RETURN
 5899  001526                           INIT:
  (1)                                   .SBTTL  INITIALIZE THE COMMON TAGS
  (1)                                   ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
  (1)   001526  012706  001100                   MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
  (1)   001532  005026                   CLR     (R6)+           ;;CLEAR MEMORY LOCATION
  (1)   001534  022706  001140                   CMP     #SWR,R6 ;;DONE?
  (1)   001540  001374                   BNE     .-6             ;;LOOP BACK IF NO
  (1)   001542  012706  001100                   MOV     #STACK,SP       ;;SETUP THE STACK POINTER
  (1)                                   ;;INITIALIZE A FEW VECTORS
  (1)   001546  012737  015136  000020           MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
  (1)   001554  012737  000300  000022           MOV     #PR6,@#IOTVEC+2 ;;LEVEL 6
  (1)   001562  012737  014574  000030           MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
  (1)   001570  012737  000300  000032           MOV     #PR6,@#EMTVEC+2 ;;LEVEL 6
  (1)                                   ;;BIT02
  (1)   001576  012737  017152  000034           MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)   001604  012737  000300  000036           MOV     #PR6,@#TRAPVEC+2;LEVEL 6
  (1)   001612  012737  016724  000024           MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
  (1)   001620  012737  000300  000026           MOV     #PR6,@#PWRVEC+2 ;;LEVEL 6
  (1)   001626  005037  001160           CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
  (1)   001632  005037  001162           CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
  (1)   001636  112737  000001  001115   MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
  (1)   001644  012737  001644  001106           MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
  (1)   001652  012737  001652  001110           MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
  (2)                                   ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
  (2)                                   ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
  (2)   001660  013746  000004           MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
  (2)   001664  012737  001720  000004           MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
  (2)   001672  012737  177570  001140           MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
  (2)   001700  012737  177570  001142           MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
  (2)   001706  022777  177777  177224           CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
  (2)   001714  001012                   BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  (2)                                                           ;;AND  THE HARDWARE SWR IS NOT = -1
  (2)   001716  000403                   BR      65$             ;;BRANCH IF NO TIMEOUT
  (2)   001720  012716  001726   64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
  (2)   001724  000002                   RTI
  (2)   001726  012737  000176  001140   65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
```

```
   (2)  001734  012737  000174  001142          MOV     #DISPREG,DISPLAY
   (2)  001742  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
   (1)
   (2)  001746  005037  001202                  CLR     $PASS           ;;CLEAR PASS COUNT
   (2)  001752  132737  000200  001215          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
   (2)  001760  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
   (2)  001762  012737  001216  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
   (2)  001770                          67$:
  5900  001770  000177  177424                  JMP     @$TMP0                  ;EXIT PROGRAM VECTOR SETUP SPACE
  5901
  5902  001774                          ZSTART:
   (1)
   (1)  001774  012746  000300                  MOV     #300,-(SP)              ;SET CPU PRIORITY ON RETERN.
   (1)  002000  012746  002006                  MOV     #64$,-(SP)              ;SHOW RETURN ADDRESS.
   (1)  002004  000002                          RTI                             ;CAUSE A RETURN(PUTS STATUS IN STATUS REG.).
   (1)  002006                          64$:
   (1)
  5903  002006  005037  001204                  CLR     $DEVCT          ;ZERO DEVICE COUNT.
  5904  002012  012737  017102  000004          MOV     #IOTRD,@#ERRVEC ;FIX TRAP CATCHER.
  5905  002020  013737  001244  001402          MOV     $VECT1,VECT1    ;NOW FIX VECTOR ADDR.
  5906  002026  013737  001250  001376          MOV     $BASE,ASR       ;FIX ADDRESS OF CSR.
  5907
  5908                                  .SBTTL  TYPE PROGRAM NAME
   (1)                                  ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
   (1)  002034  005227  177777                  INC     #-1             ;;FIRST TIME?
   (1)  002040  001041                          BNE     65$             ;;BRANCH IF NO
   (1)  002042  104401  002110                  TYPE    ,66$            ;;TYPE ASCIZ STRING
   (2)                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
   (2)  002046  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
   (2)  002052  001012                          BNE     67$             ;;BRANCH IF YES
   (2)  002054  123727  001214  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
   (2)  002062  001406                          BEQ     67$             ;;BRANCH IF YES
   (2)  002064  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
   (2)  002072  001005                          BNE     68$             ;;BRANCH IF NO
   (2)  002074  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
   (2)  002076  000403                          BR      68$
   (2)  002100  112737  000001  001134  67$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
   (2)  002106                          68$:
   (1)  002106  000416                          BR      65$             ;;GET OVER THE ASCIZ
   (1)                                  ;;66$:  .ASCIZ  <CRLF>#CNKWAA   KWV11 DIAGNOSTIC#<CRLF>
   (1)  002144                          65$:
  5909  002144                          RSTART:
  5910  002144  005737  001440                  TST     EXS             ;TESTOR MODE ENABLED??
  5911  002150  001441                          BEQ     1$              ;NO DON'T TYPE NEXT MESSAGE.
  5912  002152  104401  002160                  TYPE    ,65$            ;;TYPE ASCIZ STRING
   (1)  002156  000436                          BR      64$             ;;GET OVER THE ASCIZ
   (1)                                  ;;65$:  .ASCIZ  <15><12>#TESTOR MODE ENABLED--SEE DOCUMENTATION FOR INSTRUCTIONS.#
   (1)  002254                          64$:
  5913  002254                          1$:
   (1)  002254  104401  002262                  TYPE    ,67$            ;;TYPE ASCIZ STRING
   (1)  002260  000411                          BR      66$             ;;GET OVER THE ASCIZ
   (1)                                  ;;67$:  .ASCIZ  <15><12>#TEST RUNNING...#
   (1)  002304                          66$:
  5914  002304  005037  001434                  CLR     MDEVCT          ;TESTING FIRST UNIT.
  5915  002310  005037  001432                  CLR     ERCNT           ;NO ERRORS.
  5916  002314  005037  001202                  CLR     $PASS           ;NO PASSES.
```

```
 5917  002320  012737  000001  001426          MOV     #1,ROTATE       ;POINT TO FIRST UNIT.
 5918  002326  013737  001426  001430          MOV     ROTATE,UTEST
 5919  002334                                  LCOP:                   ;COME HERE FOR NEXT UNIT,OR END PASS.
 5920
 5921  002334  042737  170000  001402          BIC     #170000,VECT1   ;CLEAR OUT PRIORITY BITS.
 5922  002342  013737  001402  001404          MOV     VECT1,VECTP     ;NOW FIX VECTOR +2 ADDR.
 5923  002350  062737  000002  001404          ADD     #2,VECTP
 5924  002356  013737  001402  001406          MOV     VECT1,VECT2     ;LETS FIX ST2 VECTOR ADDR.
 5925  002364  062737  000004  001406          ADD     #4,VECT2        ;ITS 4 GREATER THEN THE 1ST.
 5926  002372  013737  001406  001410          MOV     VECT2,VECT2P    ;VECTOR +2 ADDR.
 5927  002400  062737  000002  001410          ADD     #2,VECT2P
 5928
 5929
 5930  002406  013737  001376  001400          MOV     ASR,ABR         ;FIX ADDR OF PRESET REG=
 5931  002414  062737  000002  001400          ADD     #2,ABR          ;CSR + 2
 5968
  (1)
  (5)                                    ;;*****************************************************************
  (4)                                    ;*TEST 1        *TEST THE ADDRESSABILITY OF CLOCK CSR
  (4)                                    ;;*****************************************************************
  (3)  002422  000240                    TST1:   NOP
  (3)
  (2)  002424  012737  000050  001160          MOV     #50,$TIMES      ;;DO 50 ITERATIONS
  (2)  002432  012737  002454  001106          MOV     #1$,$LPADR      ;;SET SCOPE LOOP ADDRESS
  (1)  002440  112737  000001  001102          MOVB    #1,$TSTNM       ;SET TEST #1.
  (1)  002446  012737  002454  001110          MOV     #1$,$LPERR
  (1)
  (1)
  (1)
  (1)  002454  013746  000004                  1$:     MOV     @#ERRVEC,-(SP)  ;SAVE CONTENTS OF ADDRS 6.
  (1)  002460  012737  002474  000004          MOV     #2$,@#ERRVEC    ;SET TIME-OUT TRAP VECTOR TO HANDLER IN CASE.
  (1)                                                                  ;WE TIME-OUT WHEN ADDRESSING THE KW11.
  (1)  002466  005777  176704                  TST     @ASR            ;ADDRESS THE CLOCK!
  (1)                                                                  ;IF CLOCK DOES NOT RETURN
  (1)                                                                  ;"BUS SSYN" THEN WE'LL TIME-OUT
  (1)
  (1)  002472  000406                          BR      3$              ;THE CLOCK WAS THERE! EXIT SUB-TEST.
  (1)  002474                                  2$:
  (2)  002474  062706  000004                  ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
  (1)  002500  013737  001376  001420          MOV     ASR,$TMPO       ;FOR ERROR TYPEOUT.
  (1)
  (2)
                                         ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  002506  104012                          ERROR   12              ;REPORT ERROR=CLOCK CSR FAILED TO RETURN
  (1)                                                                  ;"BUS SSYN" WHEN ADDRESSED.
  (1)                                                                  ;NOTE: IF PROGRAM HAS INCORRECT
  (1)                                                                  ;ADDRESS THEN WE MIG NOT BE
  (1)                                                                  ;TALKING TO THE CLOCK. MAKE SURE
  (1)                                                                  ;OF CLOCK ADDRESS.
  (1)
  (2)
                                         ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  002510  012637  000004                  3$:     MOV     (SP)+,@#ERRVEC
  (1)
```

```
 5969
  (5)                                   ;;*****************************************************************
  (4)                                   ;*TEST 2        *TEST THE ADDRESSABILITY OF CLOCK BUFFER REG.
  (4)                                   ;;*****************************************************************
  (3)  002514  000004                   TST2:   SCOPE
  (3)
  (1)
  (1)
  (1)  002516  013746  000004           1$:     MOV     @#ERRVEC,-(SP)  ;SAVE CONTENTS OF ADDRS 6.
  (1)  002522  012737  002536  000004           MOV     #2$,@#ERRVEC    ;SET TIME-OUT TRAP VECTOR TO HANDLER IN CASE.
  (1)                                                                   ;WE TIME-OUT WHEN ADDRESSING THE KW11.
  (1)  002530  005777  176644                   TST     @%BR            ;ADDRESS THE CLOCK!
  (1)                                                                   ;IF CLOCK DOES NOT RETURN
  (1)                                                                   ;'BUS SSYN' THEN WE'LL TIME-OUT
  (1)
  (1)  002534  000406                           BR      3$              ;THE CLOCK WAS THERE! EXIT SUB-TEST.
  (1)  002536                           2$:
  (2)  002536  062706  000004                   ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
  (1)  002542  013737  001400  001420           MOV     ABR,$TMPO       ;FOR ERROR TYPEOUT.
  (1)
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  002550  104012                           ERROR   12              ;REPORT ERROR=CLOCK BUFFER REG. FAILED TO RETURN
  (1)                                                                   ;'BUS SSYN' WHEN ADDRESSED.
  (1)                                                                   ;NOTE: IF PROGRAM HAS INCORRECT
  (1)                                                                   ;ADDRESS THEN WE MIG NOT BE
  (1)                                                                   ;TALKING TO THE CLOCK. MAKE SURE
  (1)                                                                   ;OF CLOCK ADDRESS.
  (1)
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  002552  012637  000004           3$:     MOV     (SP)+,@#ERRVEC
  (1)
 5977
 6015
 6016
```

```
T2      *TEST THE ADDRESSABILITY OF CLOCK BUFFER REG.                                   SEQ 0014

                        ;/*
                        ;;*****************************************************************
                        ;*TEST 3         *TEST THAT CLOCK A STATUS REGISTER BIT 14 CAN BE SET AND CLEARED
                        ;*
                        ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                        ;*F/FS OR GATES
                        ;*
                        ;;*****************************************************************
002556  000004         TST3:   SCOPE
002560  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
002566  005077  176604                  CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
002572  052777  040000  176576          BIS     #BIT14,@ASR     ;/SET BIT 14.
002600  012737  040000  001124          MOV     #BIT14,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
002606  017737  176564  001126          MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
002614  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 14 AND ONLY BIT 14 SET?
002622  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

002624  104002                          ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                ;/BIT 14 FAILED TO BIT SET.

                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

002626  000412                          BR      2$              ;/BR TO END SUBTEST.
002630  042777  040000  176540  1$:     BIC     #BIT14,@ASR     ;/TRY CLEARING BIT 14.
002636  005037  001124                  CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
002642  017737  176530  001126          MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
002650  001401                          BEQ     2$              ;/IF ZERO - NO ERROR!

                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

002652  104002                          ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                ;/BIT 14 FAILED TO CLEAR.

                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

002654                  2$:
6017
```

```
                                        T3      *TEST THAT CLOCK A STATUS REGISTER BIT 14 CAN BE SET AND CLEARED          SEQ 0015

    (1)                                         ;/#
    (5)                                 ;;**********************************************************************
    (4)                                 ;*TEST 4         *TEST THAT CLOCK A STATUS REGISTER BIT 13 CAN BE SET AND CLEARED
    (5)                                 ;*
    (5)                                 ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
    (5)                                 ;*F/FS OR GATES
    (5)                                 ;*
    (5)
    (4)                                 ;;**********************************************************************
    (3)  002654  000004                 TST4:   SCOPE
    (3)
    (2)  002656  012737  000100  001160         MOV     #100,$TIMES     ;;DO 100 ITERATIONS
    (1)
    (1)  002664  005077  176506                 CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
    (1)  002670  052777  020000  176500         BIS     #BIT13,@ASR     ;/SET BIT 13.
    (1)  002676  012737  020000  001124         MOV     #BIT13,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
    (1)  002704  017737  176466  001126         MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
    (1)  002712  023737  001124  001126         CMP     $GDDAT,$BDDAT   ;/DID BIT 13 AND ONLY BIT 13 SET?
    (1)  002720  001402                         BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
    (2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  002722  104002                         ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
    (1)                                                                 ;/BIT 13 FAILED TO BIT SET.
    (2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  002724  000412                         BR      2$              ;/BR TO END SUBTEST.
    (1)
    (1)  002726  042777  020000  176442  1$:    BIC     #BIT13,@ASR     ;/TRY CLEARING BIT 13.
    (1)  002734  005037  001124                 CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
    (1)  002740  017737  176432  001126         MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
    (1)  002746  001401                         BEQ     2$              ;/IF ZERO - NO ERROR!
    (1)
    (2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  002750  104002                         ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
    (1)                                                                 ;/BIT 13 FAILED TO CLEAR.
    (1)
    (2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  002752                         2$:
    (1)
   6018
```

```
                                        T4      *TEST THAT CLOCK A STATUS REGISTER BIT 13 CAN BE SET AND CLEARED          SEQ 0016

    (1)                                         ;/#
    (5)                                 ;;**********************************************************************
    (4)                                 ;*TEST 5        *TEST THAT CLOCK A STATUS REGISTER BIT 11 CAN BE SET AND CLEARED
    (5)                                 ;*
    (5)                                 ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
    (5)                                 ;*F/FS OR GATES
    (5)                                 ;*
    (5)
    (4)                                 ;;**********************************************************************
    (3)  002752  000004                 TST5:   SCOPE
    (3)
    (2)  002754  012737  000100  001160         MOV     #100,$TIMES     ;;DO 100 ITERATIONS
    (1)
    (1)  002762  005077  176410                 CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
    (1)  002766  052777  004000  176402         BIS     #BIT11,@ASR     ;/SET BIT 11.
    (1)  002774  012737  004000  001124         MOV     #BIT11,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
    (1)  003002  017737  176370  001126         MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
    (1)  003010  023737  001124  001126         CMP     $GDDAT,$BDDAT   ;/DID BIT 11 AND ONLY BIT 11 SET?
    (1)  003016  001402                         BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
    (2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003020  104002                         ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
    (1)                                                                 ;/BIT 11 FAILED TO BIT SET.
    (2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003022  000412                         BR      2$              ;/BR TO END SUBTEST.
    (1)
    (1)  003024  042777  004000  176344  1$:    BIC     #BIT11,@ASR     ;/TRY CLEARING BIT 11.
    (1)  003032  005037  001124                 CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
    (1)  003036  017737  176334  001126         MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
    (1)  003044  001401                         BEQ     2$              ;/IF ZERO - NO ERROR!
    (1)
    (2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003046  104002                         ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
    (1)                                                                 ;/BIT 11 FAILED TO CLEAR.
    (1)
    (2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003050                         2$:
    (1)
   6019
```

```
                                        T5      *TEST THAT CLOCK A STATUS REGISTER BIT 11 CAN BE SET AND CLEARED          SEQ 0017

    (1)                                         :/#
    (5)                                 ;;**********************************************************************
    (4)                                 ;*TEST 6        *TEST THAT CLOCK A STATUS REGISTER BIT 6 CAN BE SET AND CLEARED
    (5)                                 ;*
    (5)                                 ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
    (5)                                 ;*F/FS OR GATES
    (5)                                 ;*
    (5)
    (4)                                 ;;*********************************************   *****************
    (3)  003050  000004                 TST6:   SCOPE
    (3)
    (2)  003052  012737  000100  001160         MOV     #100,$TIMES     ;;DO 100 ITERATIONS
    (1)
    (1)  003060  005077  176312                 CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
    (1)  003064  052777  000100  176304         BIS     #BIT6,@ASR      ;/SET BIT 6.
    (1)  003072  012737  000100  001124         MOV     #BIT6,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
    (1)  003100  017737  176272  001126         MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
    (1)  003106  023737  001124  001126         CMP     $GDDAT,$BDDAT   ;/DI[ BIT 6 AND ONLY BIT 6 SET?
    (1)  003114  001402                         BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003116  104002                         ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
    (1)                                                                 ;/BIT 6 FAILED TO BIT SET.
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003120  000412                         BR      2$              ;/BR TO END SUBTEST.
    (1)
    (1)  003122  042777  000100  176246  1$:    BIC     #BIT6,@ASR      ;/TRY CLEARING BIT 6.
    (1)  003130  005037  001124                 CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
    (1)  003134  017737  176236  001126         MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
    (1)  003142  001401                         BEQ     2$              ;/IF ZERO - NO ERROR!
    (1)
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003144  104002                         ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
    (1)                                                                 ;/BIT 6 FAILED TO CLEAR.
    (1)
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003146                         2$:
    (1)
   6020
```

```
                                        T6      *TEST THAT CLOCK A STATUS REGISTER BIT 6 CAN BE SET AND CLEARED          SEQ 0018

(1)                                             :/#
(5)                                     ;;**************************************************************
(4)                                     ;*TEST 7        *TEST THAT CLOCK A STATUS REGISTER BIT 5 CAN BE SET AND CLEARED
(5)                                     ;*
(5)                                     ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(5)                                     ;*F/FS OR GATES
(5)                                     ;*
(5)
(4)                                     ;;**************************************************************
(3)  003146  000004                     TST7:   SCOPE
(3)
(2)  003150  012737  000100  001160             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(1)
(1)  003156  005077  176214              CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
(1)  003162  052777  000040  176206      BIS     #BIT5,@ASR      ;/SET BIT 5.
(1)  003170  012737  000040  001124      MOV     #BIT5,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
(1)  003176  017737  176174  001126      MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
(1)  003204  023737  001124  001126      CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 SET?
(1)  003212  001402                      BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003214  104002                     ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
(1)                                             ;/BIT 5 FAILED TO BIT SET.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003216  000412                      BR      2$              ;/BR TO END SUBTEST.
(1)
(1)  003220  042777  000040  176150  1$:     BIC     #BIT5,@ASR      ;/TRY CLEARING BIT 5.
(1)  003226  005037  001124              CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(1)  003232  017737  176140  001126      MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
(1)  003240  001401                      BEQ     2$              ;/IF ZERO - NO ERROR!
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003242  104002                     ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
(1)                                             ;/BIT 5 FAILED TO CLEAR.
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003244                             2$:
(1)
6021
```

```
    (1)                                         ;/#
    (5)                                 ;;**********************************************************************
    (4)                                 ;*TEST 10        *TEST THAT CLOCK A STATUS REGISTER BIT 4 CAN BE SET AND CLEARED
    (5)                                 ;*
    (5)                                 ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
    (5)                                 ;*F/FS OR GATES
    (5)                                 ;*
    (5)
    (4)                                 ;;**********************************************************************
    (3)  003244  000004                 TST10:  SCOPE
    (3)
    (2)  003246  012737  000100  001160         MOV     #100,$TIMES     ;;DO 100 ITERATIONS
    (1)
    (1)  003254  005077  176116                 CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
    (1)  003260  052777  000020  176110         BIS     #BIT4,@ASR      ;/SET BIT 4.
    (1)  003266  012737  000020  001124         MOV     #BIT4,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
    (1)  003274  017737  176076  001126         MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
    (1)  003302  023737  001124  001126         CMP     $GDDAT,$BDDAT   ;/DID BIT 4 AND ONLY BIT 4 SET?
    (1)  003310  001402                         BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003312  104002                         ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
    (1)                                                                 ;/BIT 4 FAILED TO BIT SET.
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003314  000412                         BR      2$              ;/BR TO END SUBTEST.
    (1)
    (1)  003316  042777  000020  176052  1$:    BIC     #BIT4,@ASR      ;/TRY CLEARING BIT 4.
    (1)  003324  005037  001124                 CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
    (1)  003330  017737  176042  001126         MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
    (1)  003336  001401                         BEQ     2$              ;/IF ZERO - NO ERROR!
    (1)
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003340  104002                         ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
    (1)                                                                 ;/BIT 4 FAILED TO CLEAR.
    (1)
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003342                         2$:
    (1)
  6022
```

```
                                        T10     *TEST THAT CLOCK A STATUS REGISTER BIT 4 CAN BE SET AND CLEARED          SEQ 0020

    (1)                                         ;/#
    (5)                                 ;;**********************************************************************
    (4)                                 ;*TEST 11       *TEST THAT CLOCK A STATUS REGISTER BIT 3 CAN BE SET AND CLEARED
    (5)                                 ;*
    (5)                                 ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
    (5)                                 ;*F/FS OR GATES
    (5)                                 ;*
    (5)
    (4)                                 ;;**********************************************************************
    (3)  003342  000004                 TST11:  SCOPE
    (3)
    (2)  003344  012737  000100  001160         MOV     #100,$TIMES     ;;DO 100 ITERATIONS
    (1)
    (1)  003352  005077  176020                 CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
    (1)  003356  052777  000010  176012         BIS     #BIT3,@ASR      ;/SET BIT 3.
    (1)  003364  012737  000010  001124         MOV     #BIT3,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
    (1)  003372  017737  176000  001126         MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
    (1)  003400  023737  001124  001126         CMP     $GDDAT,$BDDAT   ;/DID BIT 3 AND ONLY BIT 3 SET?
    (1)  003406  001402                         BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003410  104002                         ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
    (1)                                                                 ;/BIT 3 FAILED TO BIT SET.
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003412  000412                         BR      2$              ;/BR TO END SUBTEST.
    (1)
    (1)  003414  042777  000010  175754  1$:    BIC     #BIT3,@ASR      ;/TRY CLEARING BIT 3.
    (1)  003422  005037  001124                 CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
    (1)  003426  017737  175744  001126         MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
    (1)  003434  001401                         BEQ     2$              ;/IF ZERO - NO ERROR!
    (1)
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003436  104002                         ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
    (1)                                                                 ;/BIT 3 FAILED TO CLEAR.
    (1)
    (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  003440                         2$:
    (1)
   6023
```

```
 (1)                                           ;/#
 (5)                                    ;;*******************************************************************
 (4)                                    ;*TEST 12       *TEST THAT CLOCK A STATUS REGISTER BIT 2 CAN BE SET AND CLEARED
 (5)                                    ;*
 (5)                                    ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
 (5)                                    ;*F/FS OR GATES
 (5)                                    ;*
 (5)
 (4)                                    ;;*******************************************************************
 (3)  003440  000004                    TST12:  SCOPE
 (3)
 (2)  003442  012737  000100  001160            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
 (1)
 (1)  003450  005077  175722                    CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
 (1)  003454  052777  000004  175714            BIS     #BIT2,@ASR      ;/SET BIT 2.
 (1)  003462  012737  000004  001124            MOV     #BIT2,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
 (1)  003470  017737  175702  001126            MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
 (1)  003476  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
 (1)  003504  001402                            BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
 (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  003506  104002                            ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
 (1)                                                                    ;/BIT 2 FAILED TO BIT SET.
 (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  003510  000412                            BR      2$              ;/BR TO END SUBTEST.
 (1)
 (1)  003512  042777  000004  175656    1$:     BIC     #BIT2,@ASR      ;/TRY CLEARING BIT 2.
 (1)  003520  005037  001124                    CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
 (1)  003524  017737  175646  001126            MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
 (1)  003532  001401                            BEQ     2$              ;/IF ZERO - NO ERROR!
 (1)
 (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  003534  104002                            ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
 (1)                                                                    ;/BIT 2 FAILED TO CLEAR.
 (1)
 (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  003536                            2$:
 (1)
6024
```

```
                                        ;/*
                                ;;**************************************************************
                                ;*TEST 13        *TEST THAT CLOCK A STATUS REGISTER BIT 1 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*

                                ;;**************************************************************
003536  000004                  TST13:  SCOPE

003540  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003546  005077  175624                  CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003552  052777  000002  175616          BIS     #BIT1,@ASR      ;/SET BIT 1.
003560  012737  000002  001124          MOV     #BIT1,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003566  017737  175604  001126          MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003574  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
003602  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

003604  104002                          ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                ;/BIT 1 FAILED TO BIT SET.

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

003606  000412                          BR      2$              ;/BR TO END SUBTEST.

003610  042777  000002  175560  1$:     BIC     #BIT1,@ASR      ;/TRY CLEARING BIT 1.
003616  005037  001124                  CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003622  017737  175550  001126          MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003630  001401                          BEQ     2$              ;/IF ZERO - NO ERROR!

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

003632  104002                          ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                ;/BIT 1 FAILED TO CLEAR.

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

003634                          2$:
6025
```

```
T13     *TEST THAT CLOCK A STATUS REGISTER BIT 1 CAN BE SET AND CLEARED                SEQ 0023

                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 14       *TEST THAT CLOCK A STATUS REGISTER BIT 0 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;;**********************************************************************
(3)  003634  000004             TST14:  SCOPE
(3)
(2)  003636  012737  000100  001160        MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(1)
(1)  003644  005077  175526                CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
(1)  003650  052777  000001  175520        BIS     #BIT0,@ASR      ;/SET BIT 0.
(1)  003656  012737  000001  001124        MOV     #BIT0,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
(1)  003664  017737  175506  001126        MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
(1)  003672  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
(1)  003700  001402                        BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(2)

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003702  104002                        ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
(1)                                                             ;/BIT 0 FAILED TO BIT SET.
(2)

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003704  000412                        BR      2$              ;/BR TO END SUBTEST.
(1)
(1)  003706  042777  000001  175462  1$:   BIC     #BIT0,@ASR      ;/TRY CLEARING BIT 0.
(1)  003714  005037  001124                CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(1)  003720  017737  175452  001126        MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
(1)  003726  001401                        BEQ     2$              ;/IF ZERO - NO ERROR!
(1)
(2)

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003730  104002                        ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
(1)                                                             ;/BIT 0 FAILED TO CLEAR.
(1)
(2)

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003732                     2$:
(1)
6026            000010                  .RADIX  8
6057
```

```
 6059
 6060
  (5)                                   ;;*****************************************************************
  (4)                                   ;*TEST 15       *TEST THAT PATERN 125252 WILL SET AND CLEAR IN BUFFER REG.
  (4)                                   ;;*****************************************************************
  (3)  003732  000004                   TST15:  SCOPE
  (3)
  (1)
  (1)  003734  005077  175440                   CLR     @ABR            ;/CLEAR THE BUFFER REG.
  (1)  003740  012737  125252  001124           MOV     #125252,$GDDAT  ;/RECORD PATTERN: 125252 .
  (1)  003746  013777  001124  175424           MOV     $GDDAT,@ABR     ;/SET PATTERN IN BUFFER REG.
  (1)  003754  017737  175420  001126           MOV     @ABR,$BDDAT     ;/READ THE BUFFER REG.
  (1)
  (1)  003762  023737  001124  001126           CMP     $GDDAT,$BDDAT   ;/DID THE PATTERN SET OK?
  (1)  003770  001402                           BEQ     1$              ;/YES-TRY CLEARING IT.
  (1)
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  003772  104003                           ERROR   3               ;/ERROR PATTERN 125252 FAILED TO
  (1)                                                                   ;/SET PROPERLY IN BUFFER REG.
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  003774  000412                           BR      2$              ;/GOTO SCOPE LOOP.
  (1)
  (1)  003776  042777  125252  175374  1$:     BIC     #125252,@ABR    ;/TRY CLEARING PATTERN.
  (1)  004004  005037  001124                   CLR     $GDDAT          ;/EXPECT ZERO BACK.
  (1)  004010  017737  175364  001126           MOV     @ABR,$BDDAT     ;/READ BUFFER REG.,WAS IT ZERO?
  (1)  004016  001401                           BEQ     2$              ;/YES-NEXT TEST.
  (1)
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  004020  104003                           ERROR   3               ;/BUFFER REG. COULD NOT BE LOADED
  (1)                                                                   ;/TO A ZERO.
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  004022                           2$:
  (1)
```

```
 6062
 6063
  (5)                                ;;*****************************************************************
  (4)                                ;*TEST 16       *TEST THAT PATERN 052525 WILL SET AND CLEAR IN BUFFER REG.
  (4)                                ;;*****************************************************************
  (3)  004022  000004                TST16:  SCOPE
  (3)
  (1)
  (1)  004024  005077  175350                CLR     @ABR            ;/CLEAR THE BUFFER REG.
  (1)  004030  012737  052525  001124        MOV     #052525,$GDDAT  ;/RECORD PATTERN: 052525 .
  (1)  004036  013777  001124  175334        MOV     $GDDAT,@ABR     ;/SET PATTERN IN BUFFER REG.
  (1)  004044  017737  175330  001126        MOV     @ABR,$BDDAT     ;/READ THE BUFFER REG.
  (1)
  (1)  004052  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;/DID THE PATTERN SET OK?
  (1)  004060  001402                        BEQ     1$              ;/YES-TRY CLEARING IT.
  (1)
  (2)
                                     ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  004062  104003                        ERROR   3               ;/ERROR PATTERN 052525 FAILED TO
  (1)                                                                ;/SET PROPERLY IN BUFFER REG.
  (2)
                                     ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  004064  000412                        BR      2$              ;/GOTO SCOPE LOOP.
  (1)
  (1)  004066  042777  052525  175304  1$:   BIC     #052525,@ABR    ;/TRY CLEARING PATTERN.
  (1)  004074  005037  001124                CLR     $GDDAT          ;/EXPECT ZERO BACK.
  (1)  004100  017737  175274  001126        MOV     @ABR,$BDDAT     ;/READ BUFFER REG.,WAS IT ZERO?
  (1)  004106  001401                        BEQ     2$              ;/YES-NEXT TEST.
  (1)
  (2)
                                     ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  004110  104003                        ERROR   3               ;/BUFFER REG. COULD NOT BE LOADED
  (1)                                                                ;/TO A ZERO.
  (2)
                                     ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  004112                        2$:
  (1)
 6064
 6065
 6066                                        .SBTTL  *
 6067                                        .SBTTL  * PHASE 2 ADVANCED BASIC LOGIC TESTS
 6068                                        .SBTTL  *
 6069
 6078
```

```
 6080
 6081                                   ;;*****************************************************************
  (3)                                   ;*TEST 17       *TEST THE LOW BYTE OPERATION OF CLOCK'S STATUS REGISTER
  (4)                                   ;
  (4)                                   ;*
  (4)                                   ;*WE CAN SUCCESSFULLY WRITE EVERY BIT IN STATUS REG A
  (4)                                   ;*NOW LETS CHECK THE BYTE OPERATION OF THIS REGISTER.
  (4)                                   ;*
  (4)                                   ;
  (3)                                   ;*****************************************************************
  (2)  004112  000004                   TST17:  SCOPE
  (2)
  (1)  004114  012737  000050  001160           MOV     #50,$TIMES      ;;DO 50 ITERATIONS
 6082
 6083  004122  005077  175250                   CLR     @ASR            ;MAKE SURE THE STATUS REGISTER IS CLEAR.
 6084  004126  112777  127677  175242           MOVB    #127677,@ASR    ;TRY WRITING ALL BITS IN THE
 6085                                                                    ;STATUS REGISTER. LOGIC SHOULD PREVENT IT
 6086                                                                    ;FROM BEING WRITTEN INTO BECAUSE
 6087                                                                    ;WE ARE USING A DATOB INSTRUCTION.
 6088
 6089  004134  017777  175236  174764           MOV     @ASR,@$BDDAT    ;NOW EXAMINE THE
 6090                                                                    ;STATUS REGISTER.
 6091  004142  013737  001126  001124           MOV     $BDDAT,$GDDAT   ;FIX $GDDAT FOR ERROR TYPEOUT IF
 6092  004150  105037  001125                   CLRB    $GDDAT+1        ;ANY RROR HAS OCCURRED, UPPER BYTE CLEARED.
 6093
 6094  004154  105737  001127                   TSTB    $BDDAT+1        ;ARE ANY BITS IN THE UPPER BYTE
 6095                                                                    ;OF THE STATUS REGISTER SET?
 6096  004160  001401                           BEQ     1$              ;BRANCH NEXT TEST IF UPPER BYTE=0.
 6097
 6098                                   ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6099  004162  104001                           ERROR   1               ;ERROR - WROTE INTO UPPER BYTE OF
 6100                                                                    ;CLOCK'S STATUS WHEN
 6101                                                                    ;DOING A DATOB TO THE LOW BYTE.
 6102
 6103
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6104  004164                           1$:
 6105
```

```
 6107
 6108                                   ;;*****************************************************************
  (3)                                   ;*TEST 20       *TEST THE HIGH BYTE OPERATION OF A'S STATUS REGISTER
  (4)                                   ;*
  (4)                                   ;*WE CAN SUCCESSFULLY WRITE EVERY BIT IN STATUS REG A
  (4)                                   ;*NOW LETS CHECK THE BYTE OPERATION OF THIS REGISTER.
  (4)                                   ;*
  (4)
  (3)                                   ;;*****************************************************************
  (2)  004164  000004                   TST20:  SCOPE
  (2)
  (1)  004166  012737  000050  001160           MOV     #50,$TIMES      ;;DO 50 ITERATIONS
 6109
 6110  004174  005077  175176                 CLR     @ASR            ;CLEAR THE STATUS REGISTER.
 6111
 6112  004200  005237  001376                 INC     ASR             ;ADD #1 TO THE STATUS REGISTER'S ADDRESS
 6113                                                                   ;SO THAT WE WILL BE WRITING INTO
 6114                                                                   ;THE HIGH BYTE.
 6115
 6116  004204  112777  177313  175164         MOVB    #177313,@ASR    ;TRY WRITING ALL BITS IN THE STATUS
 6117                                                                   ;REGISTER. LOGIC SHOULD PREVENT THE LOW
 6118                                                                   ;BYTE OF THE STATUS REGISTER FROM
 6119                                                                   ;BEING WRITTEN INTO BECAUSE WE ARE USING
 6120                                                                   ;A DATOB INSTRUCTION WITH A00 SET.
 6121
 6122  004212  005337  001376                 DEC     ASR             ;FIX ADDRESS OF THE STATUS REGISTER ADDR.
 6123                                                                   ;SO WE CAN LOOK AT THE WHOLE WORD.
 6124  004216  017737  175154  001126         MOV     @ASR,$BDDAT     ;READ BACK WHAT THE STATUS REG. CONTAINS
 6125  004224  013737  001126  001124         MOV     $BDDAT,$GDDAT   ;FIX $GDDAT FOR ERROR TYPEOUT IF AN ERROR
 6126  004232  105037  001124                 CLRB    $GDDAT          ;OCCURRED, LOWER BYTE CLEARED.
 6127  004236  105737  001126                 TSTB    $BDDAT          ;IS LOWER BYTE CLEAR?
 6128  004242  001401                         BEQ     1$              ;BR IF YES TO NEXT SUBTEST.
 6129
 6130                                   ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6131  004244  104001                         ERROR   1               ;ERROR - WROTE INTO LOWER BYTE
 6132                                                                   ;OF CLOCKS STATUS REGISTER WHEN
 6133                                                                   ;DOING A DATOB TO THE HIGH BYTE.
 6134
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6135  004246                           1$:
 6136
```

```
 6138
 6139                                   ;;*****************************************************************
  (3)                                   ;*TEST 21       *TEST CLOCK'S COUNT REGISTER WITH 125252 PATTERN
  (3)                                   ;;*****************************************************************
  (2)  004246  000004                   TST21:  SCOPE
  (2)
  (1)  004250  012737  000100  001160           MOV     #100,$TIMES     ;;DO 100 ITERATIONS
 6140
 6141  004256  005077  175114                   CLR     @ASR            ;SELECT MODE 0.
 6142  004262  012777  125252  175110           MOV     #125252,@ABR    ;LOAD THE BUFFER REGISTER WITH
 6143                                                                    ;PATTERN 125252. IT WILL BE
 6144                                                                    ;TRANSFERRED TO THE COUNT REGISTER
 6145                                                                    ;SINCE THIS IS MODE 0.
 6146  004270  052777  000001  175100           BIS     #BIT0,@ASR      ;SET GO BIT(ALLOWS BUFFER-COUNT REG XFER).
 6147
 6148  004276  012737  125252  001124           MOV     #125252,$GDDAT  ;SET EXPECTED TO PATTERN IN CASE OF
 6149                                                                    ;NEED OF ERROR TYPEOUT.
 6150  004304  017746  175066                   MOV     @ASR,-(6)       ;/SAVE CSR
  (1)  004310  011637  001424                   MOV     (6),$TMP3       ;/GET CSR.
  (1)  004314  042737  177707  001424           BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (1)  004322  052737  004005  001424           BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1)  004330  013777  001424  175040           MOV     $TMP3,@ASR              ;/LOAD CSR.
  (1)                                                                   ;/THIS MUST BE DONE IN
  (1)                                                                   ;/ORDER TO XFERR COUNTER
  (1)                                                                   ;/TO BUFFER ON ST2.
  (1)  004336  052777  001000  175032           BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (1)  004344  017737  175030  001126           MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (1)                                                                   ;/PREVIOUS COUNTER
  (1)  004352  012677  175020                   MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (1)  004356  005737  001126                   TST     $BDDAT          ;/RESTORE CSR
 6151
 6152  004362  023737  001124  001126           CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
 6153                                                                    ;COME THROUGH?
 6154  004370  001401                           BEQ     1$              ;BR IF YES TO NEXT TEST.
 6155
 6156
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6157  004372  104005                           ERROR   5               ;DATA ERROR CLOCK - PATTERN "125252"
 6158                                                                   ;FAILED TO TRANSFER PROPERLY BETWEEN
 6159                                                                   ;BUFFER AND COUNT REGISTERS.
 6160
 6161
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6162  004374                           1$:
 6163
```

```
                                ;;*****************************************************************
                                ;*TEST 22       *TEST CLOCKS COUNTER REGISTER WITH 052525 PATTERN
                                ;;*****************************************************************
004374  000004                  TST22:  SCOPE
004376  012737  000050  001160          MOV     #50,$TIMES      ;;DO 50 ITERATIONS

004404  005077  174766          CLR     @ASR            ;SELECT MODE 0.
004410  012777  052525  174762  MOV     #052525,@ABR    ;LOAD THE BUFFER REGISTER WITH
                                                        ;PATTERN 052525. IT WILL BE
                                                        ;TRANSFERRED TO THE COUNT REGISTER
                                                        ;SINCE THIS IS MODE 0.
004416  052777  000001  174752  BIS     #BIT0,@ASR      ;SET BO BIT(ALLOWS BUFFER-COUNT REG XFER).

004424  012737  052525  001124  MOV     #052525,$GDDAT  ;SET EXPECTED TO PATTERN IN CASE OF
                                                        ;NEED OF ERROR TYPEOUT.
004432  017746  174740          MOV     @ASR,-(6)       ;/SAVE CSR
004436  011637  001424          MOV     (6),$TMP3       ;/GET CSR.
004442  042737  177707  001424  BIC     #177707,$TMP3   ;/SAVE RATE BITS.
004450  052737  004005  001424  BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
004456  013777  001424  174712  MOV     $TMP3,@ASR              ;/LOAD CSR.
                                                        ;/THIS MUST BE DONE IN
                                                        ;/ORDER TO XFERR COUNTER
                                                        ;/TO BUFFER ON ST2.
004464  052777  001000  174704  BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
004472  017737  174702  001126  MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                        ;/PREVIOUS COUNTER
004500  012677  174672          MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
004504  005737  001126          TST     $BDDAT          ;/RESTORE CSR

004510  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
                                                        ;COME THROUGH?
004516  001401                  BEQ     1$              ;BR IF YES TO NEXT TEST.


                                ;;$$$$$$$*$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$\$$$$$$$$$

004520  104005                  ERROR   5               ;DATA ERROR CLOCK - PATTERN '052525''
                                                        ;FAILED TO TRANSFER PROPERLY BETWEEN
                                                        ;BUFFER AND COUNT REGISTERS.


                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

004522                          1$:

```

```
 6199
 6200                                   ;;*****************************************************************
  (3)                                   ;*TEST 23       *TEST THAT INIT CLEARS STATUS REGISTER
  (4)                                   ;*
  (4)                                   ;*TESTING OF THE INIT LOGIC AS RECEIVED FROM THE QBUS AND BUFFERED
  (4)                                   ;*TO STATUS REGISTER F/FS.
  (4)                                   ;*
  (3)                                   ;*****************************************************************
  (2)  004522  000004                   TST23:  SCOPE
  (2)
  (1)  004524  012737  000005  001160           MOV     #5,$TIMES       ;;DO 5 ITERATIONS
 6201
 6202  004532  005037  001124                   CLR     $GDDAT          ;EXPECTED DATA IS ZERO.
 6203  004536  012777  177777  174632           MOV     #177777,@ASR    ;SET ALL BITS IN THE STATUS REG.
 6204
 6205  004544  000005                           RESET                   ;SYSTEM INITIALIZE.
 6206
 6207  004546  017737  174624  001126           MOV     @ASR,$BDDAT     ;READ THE STATUS REG., ALL BITS SHOULD
 6208                                                                   ;HAVE BEEN CLEARED BY INIT.
 6209  004554  001402                           BEQ     1$              ;BR IF YES TO NEXT TEST.
 6210
 6211
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6212
 6213  004556  104002                           ERROR   2               ;ERROR - SYSTEM INIT FAILED TO CLEAR
 6214                                                                   ;STATUS REGISTER CLOCK A.
 6215
 6216
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6217  004560  000413                           BR      TST24           ;;
 6218  004562                         1$:
 6219  004562  005737  001440                   TST     EXS             ;TEST EXTERNAL SIGS?
 6220  004566  001410                           BEQ     TST24           ;;
 6221  004570  052777  016000  174620           BIS     #BIT11!BIT12!BIT10,@DR2 ;ENABLE ST1,ST2 TO LATCH.
 6222  004576  032777  000006  174610           BIT     #6,@DR          ;ST1,ST2, OVERFLOW SET?
 6223  004604  001401                           BEQ     TST24           ;;
 6224
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6225  004606  104006                           ERROR   6               ;INIT FAILED TO CLEAR
 6226                                                                   ;ST1,ST2, AND/OR OVERFLOW
 6227
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6235
```

```
 6237
 6238                                   ;;*****************************************************************
  (3)                                   ;*TEST 24       *TEST THAT INIT CLEARS BUFFER REGISTER
  (4)                                   ;*
  (4)                                   ;*THIS TEST IS DESIGNED TO SEE IF "INIT H"
  (4)                                   ;*CLEARS THE BUFFER REGISTER. WE ALREADY
  (4)                                   ;*KNOW IT CLEARS THE STATUS REG.
  (4)                                   ;*
  (3)                                   ;*****************************************************************
  (2) 004610  000004                    TST24:  SCOPE
  (2)
  (1) 004612  012737  000005  001160            MOV     #5,$TIMES       ;;DO 5 ITERATIONS
 6239
 6240 004620  005037  001124            CLR     $GDDAT          ;CLEAR EXPECTED DATA.
 6241 004624  012777  177777  174546    MOV     #177777,@ABR    ;SET ALL BITS IN THE BUFFER REGISTER.
 6242
 6243 004632  000005                    RESET                   ;ISSUE SYSTEM INITIALIZE.
 6244
 6245 004634  017737  174540  001126    MOV     @ABR,$BDDAT     ;READ THE BUFFER REGISTER, ALL BITS
 6246                                                            ;SHOULD HAVE BEEN CLEARED BY INIT.
 6247 004642  001401                    BEQ     1$              ;BR IF YES TO NEXT SUBTEST.
 6248
 6249

                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6250 004644  104003                    ERROR   3               ;ERROR - SYSTEM INIT FAILED
 6251                                                           ;TO CLEAR BUFFER REGISTER A.
 6252
 6253

                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6254 004646                            1$:
 6255 004646  005737  001440                    TST     EXS             ;TEST EXTERNAL SIGS?
 6256 004652  001403                            BEQ     TST25           ;;
 6257 004654  052777  016000  174534            BIS     #BIT11!BIT12!BIT10,@DR2 ;ENABLE THEM
 6258
```

```
 6260
 6261                                   ;;*********************************************************************
  (3)                                   ;*TEST 25       *TEST THE SETTING OF MAINTENANCE ST2 IN CLOCK BIT 15 TO SET
  (3)                                   ;;*********************************************************************
  (2)  004662  000004                   TST25:  SCOPE
  (2)
 6262
 6263  004664  012777  000001  174504          MOV     #1,@ASR         ;SET THE GO BIT(ENABLES ST2 TO SET FLG).
 6264  004672  052777  001000  174476          BIS     #BIT9,@ASR      ;SET MAINTENANCE ST2.
 6265
 6266  004700  005777  174472                  TST     @ASR            ;DID BIT15 (ST2 FLAG) SET?
 6267  004704  100402                          BMI     1$              ;BR IF YES - NEXT TEST
 6268
 6269
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6270  004706  104001                          ERROR   1               ;ERROR - MAINTENANCE ST2 (BIT9)
 6271                                                                  ;DID NOT SET BIT15 (ST2 FLAG).
 6272
 6273
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6274  004710  000410                          BR      TST26           ;;
 6275  004712  005737  001440          1$:     TST     EXS             ;TEST EXTERNAL SIGNALS?
 6276  004716  001405                          BEQ     TST26           ;;
 6277  004720  032777  000004  174466          BIT     #BIT2,@DR       ;DID EXTERNAL ST2 GET SET?
 6278  004726  001001                          BNE     TST26           ;;
 6279
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6280  004730  104006                          ERROR   6               ;ST2 OUT NOT DETECTED
 6281                                                                  ;BY TESTOR
 6282
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6283
 6284
```

```
                                        ;;*****************************************************************
                                        ;*TEST 26       *TEST THAT BIT00 IN CLOCK STATUS REG. WILL SET WHEN BIT13 AND MAIN. ST2
                                        ;;*****************************************************************
 (2)  004732  000004                    TST26:  SCOPE
 (2)
6288
6289  004734  012777  020000  174434            MOV     #BIT13,@ASR     ;SET "ST2 ENB COUNTER" IN CLK  STATUS REG.
6290  004742  052777  001000  174426            BIS     #BIT9,@ASR      ;GENERATE A MAINTENANCE ST2.
6291  004750  032777  000001  174420            BIT     #BIT00,@ASR     ;DID BIT00 (GO) SET?
6292  004756  001001                            BNE     1$              ;BR IF YES - NEXT TEST.
6293
6294
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

6295  004760  104001                            ERROR   1               ;ERROR - BIT00 OF CLOCK'S STATUS REGISTER
6296                                                                    ;FAILED TO SET WHEN BIT13 WAS SET
6297                                                                    ;AND A MAINTENANCE ST2 GENERATED.
6298
6299
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

6300  004762  005077  174410          1$:     CLR     @ASR            ;LEAVE SUBTEST WITH CLOCK CLEAR.
6301
6302                                            .SBTTL  *
6303                                            .SBTTL  *PHASE 3 COUNT TESTS
6304                                            .SBTTL  *
6305
```

```
6307
6308                                    ;;*******************************************************************
 (3)                                    ;*TEST 27       *TEST TO SEE IF THE COUNTER WILL INCREMENT
 (3)                                    ;;*******************************************************************
 (2)  004766  000004                    TST27:  SCOPE
 (2)
6309
6310  004770  005077  174402                    CLR     @ASR            ;CLEAR THE CSR.
6311  004774  005077  174400                    CLR     @ABR            ;CLEAR THE BUFFER
6312  005000  052777  000061  174370            BIS     #BIT5!BIT4!BIT0,@ASR ;SET RATE:ST1, GO.
6313  005006  052777  000400  174362            BIS     #BIT8,@ASR      ;GENERATE A MAINTENANCE ST1.
6314                                                                    ;DID THE CLOCK COUNT?
6315  005014  017746  174356                    MOV     @ASR,-(6)       ;/SAVE CSR
 (1)  005020  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
 (1)  005024  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
 (1)  005032  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
 (1)  005040  013777  001424  174330            MOV     $TMP3,@ASR              ;/LOAD CSR.
 (1)                                                                    ;/THIS MUST BE DONE IN
 (1)                                                                    ;/ORDER TO XFERR COUNTER
 (1)                                                                    ;/TO BUFFER ON ST2.
 (1)  005046  052777  001000  174322            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
 (1)  005054  017737  174320  001126            MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
 (1)                                                                    ;/PREVIOUS COUNTER
 (1)  005062  012677  174310                    MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
 (1)  005066  005737  001126                    TST     $BDDAT          ;/RESTORE CSR
6316  005072  001001                          BNE     1$              ;YES, NEXT TEST.
6317
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
6318  005074  104006                          ERROR   6               ;CLOCK FAILED TO INCREMENT.
6319
6320
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
6321  005076                            1$:
6322
```

```
 6332
 6333                                   ;;*****************************************************************
  (3)                                   ;*TEST 30       *SEE IF CLOCK WILL COUNT UP FROM A ZERO BASE, RATE:ST1
  (4)                                   ;*
  (4)                                   ;*      NOTE:   IN THIS TEST, LOOP ON ERROR WILL CAUSE A LOOP
  (4)                                   ;*              ON THE FAILING COUT PATTERN;
  (4)                                   ;*              WHILE LOOP ON TEST WILL START THE TEST
  (4)                                   ;*              AND THE CLOCK FROM ZERO TO THE FAILING COUNT.
  (4)                                   ;*
  (3)                                   ;*****************************************************************
  (2)  005076  000004                   TST30:  SCOPE
  (2)
  (1)  005100  012737  000010  001160           MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 6334
 6335  005106  005077  174264                   CLR     @ASR            ;CLEAR THE CSR.
 6336  005112  005077  174262                   CLR     @ABR            ;CLEAR THE BUFFER REG
 6337  005116  012737  000000  001124           MOV     #0,$GDDAT       ;CLEAR EXPECTED.
 6338  005124  012737  005266  001110           MOV     #2$,$LPERR
 6339
 6340  005132  052777  000061  174236  1$:      BIS     #BIT5!BIT4!BIT0,@ASR ;START CLOCK, RATE:ST1.
 6341
 6342  005140  052777  000400  174230           BIS     #BIT8,@ASR      ;GENERATE A MAINTENANCE ST1
 6343                                                                   ;CLOCK SHOULD COUNT ONCE.
 6344  005146  005737  001440                   TST     EXS             ;TEST EXTERNAL SIGNALS?
 6345  005152  001406                           BEQ     10$             ;NO
 6346  005154  032777  000002  174232           BIT     #BIT1,@DR       ;YES - DID ST1 GET SET?
 6347  005162  001002                           BNE     10$             ;YES
 6348
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
 6349  005164  104006                           ERROR   6               ;ST1 OUT NOT DETECTED
 6350                                                                   ;BY TESTOR
 6351
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6352  005166  000447                           BR      TST31           ;;
 6353  005170                         10$:
 6354  005170  017746  174202                   MOV     @ASR,-(6)       ;/SAVE CSR
  (1)  005174  011637  001424                   MOV     (6),$TMP3       ;/GET CSR.
  (1)  005200  042737  177707  001424           BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (1)  005206  052737  004005  001424           BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1)  005214  013777  001424  174154           MOV     $TMP3,@ASR              ;/LOAD CSR.
  (1)                                                                   ;/THIS MUST BE DONE IN
  (1)                                                                   ;/ORDER TO XFERR COUNTER
  (1)                                                                   ;/TO BUFFER ON ST2.
  (1)  005222  052777  001000  174146           BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (1)  005230  017737  174144  001126           MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (1)                                                                   ;/PREVIOUS COUNTER
  (1)  005236  012677  174134                   MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (1)  005242  005737  001126                   TST     $BDDAT          ;/RESTORE CSR
 6355  005246  005237  001124                   INC     $GDDAT          ;COUNT=OLD COUNT+1
 6356  005252  023737  001124  001126           CMP     $GDDAT,$BDDAT   ;COUNT READ=COUNT EXP'ED?
 6357  005260  001402                           BEQ     2$              ;YES - SEE IF WE'RE THROUGH.
 6358
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
  6359  005262  104011                          ERROR   11              ;CLOCK FAILED TO COUNT UP PROPERLY.
  6360
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  6361  005264  000410                          BR      3$              ;GOTO SCOPE LOOP.
  6362
  6363  005266  005077  174104                2$:     CLR     @ASR
  6364  005272  013777  001124  174100              MOV     $GDDAT,@ABR
  6365  005300  005737  001124                      TST     $GDDAT          ;ALL DONE?
  6366  005304  001312                          BNE     1$              ;NO DO NEXT INCREMENT.
  6367  005306                                3$:
  6368
  6369                                          ;;**************************************************************
   (3)                                          ;*TEST 31        *TEST THAT OVERFLOW (CSR BIT07) WILL SET ON OVERFLOW
   (3)                                          ;;**************************************************************
   (2)  005306  000004                          TST31:  SCOPE
   (2)
  6370
  6371  005310  005737  001440                      TST     EXS             ;TESTOR MODE ENABLED??
  6372  005314  001411                          BEQ     2$              ;NO-THEN SKIP NEXT SECTION OF CODE.
  6373
  6374  005316  005037  170500                      CLR     @#170500        ;CLEAR TESTOR A/D
  6375  005322  005737  170502                      TST     @#170502        ;DUMB READ OF A/D BUFFER.
  6376  005326  052737  000040  170500              BIS     #BIT5,@#170500  ;ENABLE EXTRENAL START OF A/D.
  6377  005334  012700  000010                      MOV     #8.,R0          ;SET TIME OUT NUMBER.
  6378  005340                                2$:
  6379  005340  005077  174032                      CLR     @ASR            ;CLEAR THE CSR
  6380  005344  012777  177777  174026              MOV     #-1,@ABR        ;SET PRESET BUFFER TO ALL ONES.
  6381
  6382  005352  052777  000061  174016              BIS     #BIT5!BIT4!BIT0,@ASR ;START CLOCK, RATE ST1.
  6383
  6384  005360  052777  000400  174010              BIS     #BIT8,@ASR      ;COUNT CLOCK ONCE, OVERFLOW
  6385                                                                    ;SHOULD OCCUR.
  6386  005366  105777  174004                      TSTB    @ASR            ;DID OVERFLOW SET?
  6387  005372  100402                          BMI     1$              ;YES - THEN NEXT TEST
  6388
  6389
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  6390  005374  104006                          ERROR   6               ;ERROR - OVERFLOW, CSR BIT0?
  6391                                                                    ;FAILED TO SET ON OVERFLOW
  6392  005376  000411                          BR      TST32           ;;
  6393  005400  005737  001440                1$:     TST     EXS             ;TEST EXTERNAL SIGNALS?
  6394  005404  001406                          BEQ     TST32           ;;
  6395  005406  105737  170500                      TSTB    @#170500        ;IF  OVERFLOW GOT OUT,IT GAVE A/D START,
  6396                                                                    ;WE'RE LOOKING FOR A/D DONE-DID IT GET SET?
  6397  005412  100403                          BMI     TST32           ;;
  6398  005414  005300                          DEC     R0              ;DID WE ALLOW ENOUGH TIME??
  6399  005416  001370                          BNE     1$              ;NO-THEN WAIT.
  6400
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  6401  005420  104006                          ERROR   6               ;OVERFLOW OUT NOT DETECTED
  6402                                                                    ;BY TESTOR
  6403
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
                                ;;*****************************************************************
                                ;*TEST 32        *TEST THAT OVERFLOW WILL CLEAR THE GO BIT
                                ;;*****************************************************************
005422  000004                  TST32:  SCOPE

005424  005077  173746                  CLR     @ASR            ;CLEAR THE CSR.

005430  012777  177777  173742          MOV     #-1,@ABR        ;PRESET CLOCK TO -1.

005436  052777  000061  173732          BIS     #BIT5!BIT4!BIT0,@ASR ;START CLOCK, RATE:ST1

005444  052777  000400  173724          BIS     #BIT8,@ASR      ;COUNT ONCE, OVERFLOW
                                                                ;SHOULD OCCUR CLEARING
                                                                ;ENABLE (CSR BIT00)

005452  032777  000001  173716          BIT     #BIT0,@ASR      ;DID THE ENABLE CLEAR?
005460  001401                          BEQ     1$              ;YES - NEXT TEST.


                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

005462  104006                          ERROR   6               ;ERROR - OVERFLOW FAILED
                                                                ;TO CLEAR ENABLE (CSR BIT00)

005464                          1$:

                                ;;*****************************************************************
                                ;*TEST 33        *TEST THAT GO BIT DOES NOT CLEAR ON OVERFLOW, IF MODE 1
                                ;;*****************************************************************
005464  000004                  TST33:  SCOPE

005466  005077  173704                  CLR     @ASR            ;CLEAR THE CSR.
005472  012777  177777  173700          MOV     #-1,@ABR        ;PRESET BUFFER=ONE COUNT FROM OVERFLOW.
005500  052777  000063  173670          BIS     #63,@ASR        ;MODE 1, RATE:ST1, GO.

005506  052777  000400  173662          BIS     #BIT8,@ASR      ;GENERATE MAINTENANCE ST1.

005514  032777  000001  173654          BIT     #BIT0,@ASR      ;DID ENABLE (GO BIT) CLEAR?
005522  001001                          BNE     1$              ;NO (GOOD) NEXT TEST.


                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

005524  104006                          ERROR   6               ;GO BIT CLEARED ON OVERFLOW
                                                                ;WHEN MODE 1 WAS SELECTED


                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


005526  005077  173644          1$:     CLR     @ASR            ;CLEAR THE CLOCK.


```

```
6490
```

```
(5)                                        ;;*****************************************************************
(4)                                        ;*TEST 34        *TEST THE ABILITY OF CLOCK TO COUNT AT 1MHZ RATE
(5)
(5)                                        ;*
(5)                                        ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
(5)                                        ;*TO COUNT AT 1MHZ RATE.
(5)                                        ;*
(4)                                        ;;*****************************************************************
(3)  005532 000004                         TST34:  SCOPE
(3)
(2)  005534 012737 000005 001160                   MOV     #5,$TIMES          ;;DO 5 ITERATIONS
(1)
(1)
(1)  005542 005077 173630                          CLR     @ASR               ;/CLEAR CLOCK
(1)  005546 005077 173626                          CLR     @ABR               ;/CLEAR PRESET BUFFER
(1)  005552 012777 000011 173616                   MOV     #BIT0!10,@ASR      ;/START CLOCK, MODE0, RATE:1MHZ
(1)  005560 005000                                 CLR     R0                 ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
(1)
(1)  005562 005200                         1$:     INC     R0                 ;/WILL AMOUNT TO APPROXIMATELY
(1)  005564 001376                                 BNE     1$                 ;/369 MS.
(1)
(2)  005566 017746 173604                          MOV     @ASR,-(6)          ;/SAVE CSR
(2)  005572 011637 001424                          MOV     (6),$TMP3          ;/GET CSR.
(2)  005576 042737 177707 001424                   BIC     #177707,$TMP3      ;/SAVE RATE BITS.
(2)  005604 052737 004005 001424                   BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  005612 013777 001424 173556                   MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                           ;/THIS MUST BE DONE IN
(2)                                                                           ;/ORDER TO XFERR COUNTER
(2)                                                                           ;/TO BUFFER ON ST2.
(2)  005620 052777 001000 173550                   BIS     #BIT9,@ASR         ;/GENERATE ON ST2 PULSE
(2)  005626 017737 173546 001126                   MOV     @ABR,$BDDAT        ;/READ THE PRESET BUFFER,
(2)                                                                           ;/PREVIOUS COUNTER
(2)  005634 012677 173536                          MOV     (6)+,@ASR          ;/CONTENTS ARE IN $BDDAT.
(2)  005640 005737 001126                          TST     $BDDAT             ;/RESTORE CSR
(1)  005644 001004                                 BNE     2$                 ;/YES - NEXT TEST.
(1)  005646 105766 177776                          TSTB    -2(6)              ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
(1)                                                                           ;/NOTE: CSR HAD BEEN PUT ON STACK.
(1)  005652 100401                                 BMI     2$                 ;/NEXT TEST IF OVERFLOW.
(1)
(2)
                                           ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  005654 104006                                 ERROR   6                  ;/CLOCK FAILED TO COUNT AT
(1)                                                                           ;/RATE:1MHZ
(2)
                                           ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)
(1)  [illegible]
(1)
(1)
```

```
 6495
 6496
  (5)                                   ;;*****************************************************************
  (4)                                   ;*TEST 35       *TEST THE ABILITY OF CLOCK TO COUNT AT 100KHZ RATE
  (5)                                   ;*
  (5)                                   ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
  (5)                                   ;*TO COUNT AT 100KHZ RATE.
  (5)                                   ;*
  (4)                                   ;;*****************************************************************
  (3)  005662  000004                   T$T35:  SCOPE
  (3)
  (2)  005664  012737  000005  001160           MOV     #5,$TIMES       ;;DO 5 ITERATIONS
  (1)
  (1)
  (1)  005672  005077  173500           CLR     @ASR            ;/CLEAR CLOCK
  (1)  005676  005077  173476           CLR     @ABR            ;/CLEAR PRESET BUFFER
  (1)  005702  012777  000021  173466   MOV     #BIT0!20,@ASR   ;/START CLOCK, MODE0, RATE:100KHZ
  (1)  005710  005000                   CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
  (1)
  (1)  005712  005200           1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
  (1)  005714  001376                   BNE     1$              ;/369 MS.
  (1)
  (2)  005716  017746  173454           MOV     @ASR,-(6)       ;/SAVE CSR
  (2)  005722  011637  001424           MOV     (6),$TMP3       ;/GET CSR.
  (2)  005726  042737  177707  001424   BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (2)  005734  052737  004005  001424   BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (2)  005742  013777  001424  173426   MOV     $TMP3,@ASR              ;/LOAD CSR.
  (2)                                                   ;/THIS MUST BE DONE IN
  (2)                                                   ;/ORDER TO XFERR COUNTER
  (2)                                                   ;/TO BUFFER ON ST2.
  (2)  005750  052777  001000  173420   BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (2)  005756  017737  173416  001126   MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (2)                                                   ;/PREVIOUS COUNTER
  (2)  005764  012677  173406           MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (2)  005770  005737  001126           TST     $BDDAT          ;/RESTORE CSR
  (1)  005774  001004                   BNE     2$              ;/YES - NEXT TEST.
  (1)  005776  105766  177776           TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
  (1)                                                   ;/NOTE: CSR HAD BEEN PUT ON STACK.
  (1)  006002  100401                   BMI     2$              ;/NEXT TEST IF OVERFLOW.
  (1)
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  006004  104006                   ERROR   6               ;/CLOCK FAILED TO COUNT AT
  (1)                                                   ;/RATE:100KHZ
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)
  (1)  006006  005077  173364   2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
  (1)
  (1)
```

```
                                        ;;*****************************************************************
                                        ;*TEST 36       *TEST THE ABILITY OF CLOCK TO COUNT AT 10KHZ RATE
                                        ;*
                                        ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
                                        ;*TO COUNT AT 10KHZ RATE.
                                        ;*
                                        ;;*****************************************************************
006012  000004                  TST36:  SCOPE

006014  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS

006022  005077  173350                  CLR     @ASR            ;/CLEAR CLOCK
006026  005077  173346                  CLR     @ABR            ;/CLEAR PRESET BUFFER
006032  012777  000031  173336          MOV     #BIT0!30,@ASR   ;/START CLOCK, MODE0, RATE:10KHZ
006040  005000                          CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
006042  005200                  1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
006044  001376                          BNE     1$              ;/369 MS.

006046  017746  173324                  MOV     @ASR,-(6)       ;/SAVE CSR
006052  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
006056  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
006064  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
006072  013777  001424  173276          MOV     $TMP3,@ASR              ;/LOAD CSR.
                                                                ;/THIS MUST BE DONE IN
                                                                ;/ORDER TO XFERR COUNTER
                                                                ;/TO BUFFER ON ST2.
006100  052777  001000  173270          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
006106  017737  173266  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                                ;/PREVIOUS COUNTER
006114  012677  173256                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
006120  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
006124  001004                          BNE     2$              ;/YES - NEXT TEST.
006126  105766  177776                  TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
                                                                ;/NOTE: CSR HAD BEEN PUT ON STACK.
006132  100401                          BMI     2$              ;/NEXT TEST IF OVERFLOW.

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

006134  104006                          ERROR   6               ;/CLOCK FAILED TO COUNT AT
                                                                ;/RATE:10KHZ

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

006136  005077  173234          2$:    CLR     @ASR            ;/CLEAR THE CLOCK.
```

```
 6501
 6502
  (5)                                ;;*****************************************************************
  (4)                                ;*TEST 37       *TEST THE ABILITY OF CLOCK TO COUNT AT 1KHZ RATE
  (5)
  (5)                                ;*
  (5)                                ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
  (5)                                ;*TO COUNT AT 1KHZ RATE.
  (5)                                ;*
  (4)                                ;;*****************************************************************
  (3)  006142  000004                TST37:  SCOPE
  (3)
  (2)  006144  012737  000005  001160        MOV     #5,$TIMES       ;;DO 5 ITERATIONS
  (1)
  (1)
  (1)  006152  005077  173220                CLR     @ASR            ;/CLEAR CLOCK
  (1)  006156  005077  173216                CLR     @ABR            ;/CLEAR PRESET BUFFER
  (1)  006162  012777  000041  173206        MOV     #BIT0!40,@ASR   ;/START CLOCK, MODE0, RATE:1KHZ
  (1)  006170  005000                        CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
  (1)
  (1)  006172  005200                1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
  (1)  006174  001376                        BNE     1$              ;/369 MS.
  (1)
  (2)  006176  017746  173174                MOV     @ASR,-(6)       ;/SAVE CSR
  (2)  006202  011637  001424                MOV     (6),$TMP3       ;/GET CSR.
  (2)  006206  042737  177707  001424        BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (2)  006214  052737  004005  001424        BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (2)  006222  013777  001424  173146        MOV     $TMP3,@ASR              ;/LOAD CSR.
  (2)                                                                ;/THIS MUST BE DONE IN
  (2)                                                                ;/ORDER TO XFERR COUNTER
  (2)                                                                ;/TO BUFFER ON ST2.
  (2)  006230  052777  001000  173140        BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (2)  006236  017737  173136  001126        MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (2)                                                                ;/PREVIOUS COUNTER
  (2)  006244  012677  173126                MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (2)  006250  005737  001126                TST     $BDDAT          ;/RESTORE CSR
  (1)  006254  001004                        BNE     2$              ;/YES - NEXT TEST.
  (1)  006256  105766  177776                TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
  (1)                                                                ;/NOTE: CSR HAD BEEN PUT ON STACK.
  (1)  006262  100401                        BMI     2$              ;/NEXT TEST IF OVERFLOW.
  (1)
  (2)
                                     ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  006264  104006                        ERROR   6               ;/CLOCK FAILED TO COUNT AT
  (1)                                                                ;/RATE:1KHZ
  (2)
                                     ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)
  (1)  006266  005077  173104        2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
  (1)
  (1)
```

```
 6504
 6505
  (5)                                ;;*****************************************************************
  (4)                                ;*TEST 40       *TEST THE ABILITY OF CLOCK TO COUNT AT 100HZ RATE
  (5)                                ;*
  (5)                                ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
  (5)                                ;*TO COUNT AT 100HZ RATE.
  (5)                                ;*
  (4)                                ;;*****************************************************************
  (3)  006272  000004                TST40:  SCOPE
  (3)
  (2)  006274  012737  000005  001160        MOV     #5,$TIMES       ;;DO 5 ITERATIONS
  (1)
  (1)
  (1)  006302  005077  173070                 CLR     @ASR            ;/CLEAR CLOCK
  (1)  006306  005077  173066                 CLR     @ABR            ;/CLEAR PRESET BUFFER
  (1)  006312  012777  000051  173056        MOV     #BIT0!50,@ASR   ;/START CLOCK, MODE0, RATE:100HZ
  (1)  006320  005000                         CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
  (1)
  (1)  006322  005200                1$:      INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
  (1)  006324  001376                         BNE     1$              ;/369 MS.
  (1)
  (2)  006326  017746  173044                 MOV     @ASR,-(6)       ;/SAVE CSR
  (2)  006332  011637  001424                 MOV     (6),$TMP3       ;/GET CSR.
  (2)  006336  042737  177707  001424        BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (2)  006344  052737  004005  001424        BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (2)  006352  013777  001424  173016        MOV     $TMP3,@ASR              ;/LOAD CSR.
  (2)                                                                 ;/THIS MUST BE DONE IN
  (2)                                                                 ;/ORDER TO XFERR COUNTER
  (2)                                                                 ;/TO BUFFER ON ST2.
  (2)  006360  052777  001000  173010        BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (2)  006366  017737  173006  001126        MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (2)                                                                 ;/PREVIOUS COUNTER
  (2)  006374  012677  172776                 MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (2)  006400  005737  001126                 TST     $BDDAT          ;/RESTORE CSR
  (1)  006404  001004                         BNE     2$              ;/YES - NEXT TEST.
  (1)  006406  105766  177776                 TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
  (1)                                                                 ;/NOTE: CSR HAD BEEN PUT ON STACK.
  (1)  006412  100401                         BMI     2$              ;/NEXT TEST IF OVERFLOW.
  (1)
  (2)
                                     ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  006414  104006                         ERROR   6               ;/CLOCK FAILED TO COUNT AT
  (1)                                                                 ;/RATE:100HZ
  (2)
                                     ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)
  (1)  006416  005077  172754        2$:      CLR     @ASR            ;/CLEAR THE CLOCK.
  (1)
  (1)
```

```
 6507
 6508
  (5)                                   ;;*****************************************************************
  (4)                                   ;*TEST 41        *TEST THE ABILITY OF CLOCK TO COUNT AT LINEFREQ RATE
  (5)                                   ;
  (5)                                   ;*
  (5)                                   ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
  (5)                                   ;*TO COUNT AT LINEFREQ RATE.
  (5)                                   ;*
  (4)                                   ;*****************************************************************
  (3)  006422  000004                   TST41:  SCOPE
  (3)
  (2)  006424  012737  000005  001160           MOV     #5,$TIMES       ;;DO 5 ITERATIONS
  (1)
  (1)  006432  032777  010000  172500           BIT     #BIT12,@SWR     ;SHALL WE TEST
  (1)                                                                   ;LINE FREQ?
  (1)                                                                   ;SW12=1
  (3)  006440  001450                           BEQ     TST42           ;;
  (1)
  (1)  006442  005077  172730                   CLR     @ASR            ;/CLEAR CLOCK
  (1)  006446  005077  172726                   CLR     @ABR            ;/CLEAR PRESET BUFFER
  (1)  006452  012777  000071  172716           MOV     #BIT0!70,@ASR   ;/START CLOCK, MODE0, RATE:LINEFREQ
  (1)  006460  005000                           CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
  (1)
  (1)  006462  005200                   1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
  (1)  006464  001376                           BNE     1$              ;/369 MS.
  (1)
  (2)  006466  017746  172704                   MOV     @ASR,-(6)       ;/SAVE CSR
  (2)  006472  011637  001424                   MOV     (6),$TMP3       ;/GET CSR.
  (2)  006476  042737  177707  001424           BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (2)  006504  052737  004005  001424           BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL CSC
  (2)  006512  013777  001424  172656           MOV     $TMP3,@ASR              ;/LOAD CSR.
  (2)                                                                   ;/THIS MUST BE DONE IN
  (2)                                                                   ;/ORDER TO XFERR COUNTER
  (2)                                                                   ;/TO BUFFER ON ST2.
  (2)  006520  052777  001000  172650           BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (2)  006526  017737  172646  001126           MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (2)                                                                   ;/PREVIOUS COUNTER
  (2)  006534  012677  172636                   MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (2)  006540  005737  001126                   TST     $BDDAT          ;/RESTORE CSR
  (1)  006544  001004                           BNE     2$              ;/YES - NEXT TEST.
  (1)  006546  105766  177776                   TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
  (1)                                                                   ;/NOTE: CSR HAD BEEN PUT ON STACK.
  (1)  006552  100401                           BMI     2$              ;/NEXT TEST IF OVERFLOW.
  (1)
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  006554  104006                           ERROR   6               ;/CLOCK FAILED TO COUNT AT
  (1)                                                                   ;/RATE:LINEFREQ
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)
  (1)  006556  005077  172614           2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
  (1)
```

```
   (1)
  6509
```

```
                                ;;*****************************************************************
                                ;*TEST 42        *TEST THAT COUNTER DOESN'T COUNT WHEN "SLAVE IN" RATE IS SELECTED
                                ;;*****************************************************************
006562  000004                TST42:  SCOPE

006564  012737  000020  001160        MOV     #20,$TIMES      ;;DO 20 ITERATIONS

006572  005077  172600                CLR     @ASR            ;CLEAR CSR.
006576  005077  172576                CLR     @ABR            ;CLEAR PRESET BUFFER.

006602  052777  000001  172566        BIS     #BIT0,@ASR      ;SET GO BIT, RATE:"SLAVE IN"

006610  052777  000400  172560        BIS     #BIT8,@ASR      ;GENERATE A MAINTENANCE ST1.
006616  005000                        CLR     R0

006620  005200                1$:     INC     R0              ;NOW DO A SHORT DELAY.
006622  001376                        BNE     1$

006624  017746  172546                MOV     @ASR,-(6)       ;/SAVE CSR
006630  011637  001424                MOV     (6),$TMP3       ;/GET CSR.
006634  042737  177707  001424        BIC     #177707,$TMP3   ;/SAVE RATE BITS.
006642  052737  004005  001424        BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
006650  013777  001424  172520        MOV     $TMP3,@ASR              ;/LOAD CSR.
                                                        ;/THIS MUST BE DONE IN
                                                        ;/ORDER TO XFERR COUNTER
                                                        ;/TO BUFFER ON ST2.
006656  052777  001000  172512        BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
006664  017737  172510  001126        MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                        ;/PREVIOUS COUNTER
006672  012677  172500                MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
006676  005737  001126                TST     $BDDAT          ;/RESTORE CSR
006702  001003                        BNE     2$              ;IF ANY COUNT, ERROR.
006704  105777  172466                TSTB    @ASR            ;CHECK FOR OVERFLOW.
006710  100003                        BPL     3$              ;IF NO OVERFLOW, NEXT TEST

006712  005037  001124        2$:     CLR     $GDDAT          ;CLEAR FOR TYPEOUT, EXP'D ZERO.

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

006716  104011                        ERROR   11              ;CLOCK COUNTED WHEN RATE:"SLAVE IN"
                                                        ;WAS SELECTED.

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

006720  005077  172452        3$:     CLR     @ASR            ;CLEAR CLOCK'S CSR.


                                ;;*****************************************************************
                                ;*TEST 43        *TEST THAT THE CLOCK WILL COUNT IN MODE 1
                                ;;*****************************************************************
006724  000004                TST43:  SCOPE

006726  005077  172444                CLR     @ASR            ;CLEAR THE CSR.
006732  012777  177777  172440        MOV     #-1,@ABR        ;PRESET BUFFER REG.
```

```
 6542  006740  052777  000013  172430          BIS     #13,@ASR        ;START CLOCK,RATE:1 MHZ,MODE 1.
 6543  006746  005000                          CLR     R0
 6544  006750  105200                  1$:     INCB    R0              ;NOW DO A SHORT DELAY SO THAT THE
 6545  006752  001376                          BNE     1$              ;CLOCK CAN COUNT TO OVERFLOW.
 6546
 6547  006754  105777  172416                  TSTB    @ASR            ;OVERFLOW SHOULD HAVE SET.
 6548  006760  100401                          BMI     2$              ;GOTO NEXT TEST IF SO.
 6549
 6550
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6551  006762  104006                          ERROR   6               ;CLOCK FAILED TO COUNT UP AND SET
 6552                                          ;OVERFLOW IN MODE 1.
 6553
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6554  006764                          2$:
 6555
 6556
 6557                                           .SBTTL  *
 6558                                           .SBTTL  *PHASE 4 CLOCK INTERRUPT TEST.
 6559                                           .SBTTL  *
 6560
 6568
 6569                                   ;;*****************************************************************
  (3)                                   ;*TEST 44       *TEST THAT THE CLOCK WILL INTERRUPT ON OVERFLOW
  (3)                                   ;;*****************************************************************
  (2)  006764  000004                   TST44:  SCOPE
  (2)
  (1)  006766  012737  000020  001160           MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 6570
 6571
  (1)  006774  012746  000340                   MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
  (1)  007000  012746  007006                   MOV     #64$,-(SP)      ;PUT RETURN ADDRESS ON STACK
  (1)  007004  000002                           RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
  (1)  007006                           64$:
 6572
 6573  007006  005077  172364                  CLR     @ASR            ;CLEAR CLOCK'S CSR.
 6574  007012  012777  177777  172360          MOV     #-1,@ABR        ;SET PRESET BUFFER TO ALL ONES.
 6575
 6576  007020  012777  000161  172350          MOV     #161,@ASR       ;START CLOCK, RATE:ST1.
 6577  007026  052777  000400  172342          BIS     #BIT8,@ASR      ;GENERATE A MAINTENANCE ST1.
 6578  007034  012777  007074  172340          MOV     #1$,@VECT1      ;SET INTERRUPT ADDR.
 6579
  (1)  007042  012746  000000                   MOV     #0,-(SP)        ;PUT PRIORITY ON STACK.
  (1)  007046  012746  007054                   MOV     #65$,-(SP)      ;PUT RETURN ADDRESS ON STACK
  (1)  007052  000002                           RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
  (1)  007054                           65$:
 6580
 6581  007054  000240                          NOP                     ;STALL TIME
 6582
  (1)  007056  012746  000340                   MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
  (1)  007062  012746  007070                   MOV     #66$,-(SP)      ;PUT RETURN ADDRESS ON STACK
  (1)  007066  000002                           RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
  (1)  007070                           66$:
 65R3
```

```
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
 6584  007070  104007                             ERROR   7               ;CLOCK FAILED TO INTERRUPT.
 6585
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6586  007072  000402                             BR      2$
 6587  007074                          1$:
  (1)  007074  062706  000004                     ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
 6588  007100  005077  172272          2$:       CLR     @ASR            ;CLEAR THE CLOCK.
 6589
 6590                                   ;;=====================================================================
  (3)                                   ;*TEST 45       *TEST THAT ST2 WILL CAUSE AN INTERRUPT
  (3)                                   ;;=====================================================================
  (2)  007104  000004                   TST45:  SCOPE
  (2)
 6591
 6592
  (1)  007106  012746  000340                     MOV     #340,-(SP)      ;PU: PRIORITY ON STACK.
  (1)  007112  012746  007120                     MOV     #64$,-(SP)      ;PUT RETURN ADDRESS ON STACK
  (1)  007116  000002                             RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
  (1)  007120                           64$:
 6593
 6594  007120  005077  172252                     CLR     @ASR            ;CLEAR CLOCKS CSR.
 6595  007124  012777  007200  172254             MOV     #1$,@VECT2      ;SET UP INTERRUPT VECTOR.
 6596  007132  012777  040001  172236             MOV     #BIT14!BIT0,@ASR;SET "INT2",AND GO BIT.
 6597  007140  052777  001000  172230             BIS     #BIT9,@ASR      ;GENERATE A MAINTENANCE ST2.
 6598
  (1)  007146  012746  000000                     MOV     #0,-(SP)        ;PUT PRIORITY ON STACK.
  (1)  007152  012746  007160                     MOV     #65$,-(SP)      ;PUT RETURN ADDRESS ON STACK
  (1)  007156  000002                             RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
  (1)  007160                           65$:
 6599
 6600  007160  000240                             NOP                     ;STALL TIME
 6601
  (1)  007162  012746  000340                     MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
  (1)  007166  012746  007174                     MOV     #66$,-(SP)      ;PUT RETURN ADDRESS ON STACK
  (1)  007172  000002                             RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
  (1)  007174                           66$:
 6602
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6603  007174  104007                             ERROR   7               ;CLOCK FAILED TO INTERRUPT ON ST2.
 6604
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6605  007176  000402                             BR      2$
 6606  007200                          1$:
  (1)  007200  062706  000004                     ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
 6607  007204  005077  172166          2$:       CLR     @ASR            ;CLEAR CLOCK'S CSR.
 6608
 6609                                           .SBTTL  *
 6610                                           .SBTTL  *PHASE 5 ADVANCED TESTING
 6611                                           .SBTTL  *
 6612
```

```
                                        ;;*****************************************************************
                                        ;*TEST 46       *TEST THAT THE "FOR" BIT WILL SET ON 2 ST2'S
                                        ;;*****************************************************************
 007210  000004                 TST46:  SCOPE

 007212  005077  172160                 CLR     @ASR            ;START WITH CSR CLEAR.
 007216  005277  172154                 INC     @ASR            ;SET GO BIT.
 007222  052777  001000  172146         BIS #BIT9,@ASR          ; GENERATE THE 1ST ST2 PULSE.
 007230  052777  001000  172140         BIS     #BIT9,@ASR      ;GENERATE 2ND ST2 PULSE.
                                                                ;THIS SHOULD CAUSE FOR BIT TO SET.
 007236  032777  010000  172132         BIT     #BIT12,@ASR     ;DID FOR BIT SET?
 007244  001007                         BNE     1$              ;YES-THEN NEXT TEST.

 007246  017737  172124  001126         MOV     @ASR,$BDDAT     ;RECORD CSR.
 007254  012737  110001  001124         MOV     #BIT15!BIT12!BIT0,$GDDAT        ;RECORD S/B.

                                 ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 007262  104001                         ERROR   1               ;ERROR-"FOR" BIT FAILED TO SET ON
                                                                ;ON TWO SUCCESIVE ST2 PULSES.

                                 ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 007264                          1$:

                                        ;;*****************************************************************
                                        ;*TEST 47       *TEST THAT FOR BIT WILL SET ON TWO OVERFLOWS
                                        ;;*****************************************************************
 007264  000004                 TST47:  SCOPE

 007266  012737  000002  001160         MOV     #2,$TIMES       ;;DO 2 ITERATIONS

 007274  005077  172076                 CLR     @ASR            ;START WITH CSR CLEAR.
 007300  012777  177777  172072         MOV     #-1,@ABR        ;PRELOAD BUFFER REG.
 007306  052777  000013  172062         BIS     #13,@ASR        ;START CLOCK,MODE 1,1MHZ.
 007314  005000                         CLR     R0              ;NOW DO A SHORT DELAY.
 007316  105200                 1$:     INCB    R0              ;DURING THIS DELAY, THE CLOCK WILL
 007320  001376                         BNE     1$              ;HAVE OVERFLOWED TWICE-
                                                                ;THIS SHOULD CAUSE THE FOR BIT TO SET.
 007322  032777  010000  172046         BIT     #BIT12,@ASR     ;DID FOR SET?
 007330  001007                         BNE     2$              ;YES-NEXT TEST.

 007332  017737  172040  001126         MOV     @ASR,$BDDAT     ;NO-RECORD THE CSR.
 007340  012737  010213  001124         MOV     #010213,$GDDAT  ;RECORD S/B.

                                 ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 007346  104001                         ERROR   1               ;ERROR FOR BIT FAILED TO SET ON
                                                                ; TWO SUCCESSIVE OVERFLOWS.

                                 ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 007350                          2$:
                                 ;;*****************************************************************
```

```
  (3)                                   ;*TEST 50       *TEST THAT FOR BIT WILL CLEAR IF GO BIT IS SET
  (3)                                   ;;*************************************************************
  (2)  007350  000004                   TST50:  SCOPE
  (2)
6733
6734  007352  012777  000001  172016            MOV     #BIT0,@ASR      ;CLEAR CSR,SET GO BIT.
6735  007360  052777  001000  172010            BIS     #BIT9,@ASR      ;SET 1ST ST2 PULSE.
6736  007366  052777  001000  172002            BIS     #BIT9,@ASR      ;GENERATE 2ND ST2 PULSE.
6737                                                                    ;FOR BIT SETS HERE.
6738  007374  042777  000001  171774            BIC     #BIT0,@ASR      ;CLEAR GO BIT.
6739
6740  007402  052777  000001  171766            BIS     #BIT0,@ASR      ;SET THE "GO" BIT AGAIN -
6741                                                                    ;SHOULD CLEAR FOR BIT.
6742
6743  007410  017737  171762  001126            MOV     @ASR,$BDDAT     ;READ THE CSR.
6744
6745  007416  012737  100001  001124            MOV     #100001,$GDDAT  ;RECORD WHAT CSR S/B.
6746  007424  032737  010000  001126            BIT     #BIT12,$BDDAT   ;DID FOR BIT CLEAR?
6747  007432  001401                            BEQ     1$              ;YES NEXT TEST.
6748
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

6749  007434  104001                            ERROR   1               ;FOR BIT FALED TO CLEAR
6750                                                                    ;WHEN "GO" BIT WAS SET.
6751
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

6752  007436  005077  171734            1$:     CLR     @ASR            ;CLEAR THE CSR.
6753
6754                                    ;;*************************************************************
  (3)                                   ;*TEST 51       *TEST THAT WE CAN DISABLE THE INTERNAL OSC
  (3)                                   ;;*************************************************************
  (2)  007442  000004                   TST51:  SCOPE
  (2)
  (1)  007444  012737  000005  001160            MOV     #5,$TIMES       ;;DO 5 ITERATIONS
6755
6756  007452  005077  171720                    CLR     @ASR            ;CLEAR THE CSR
6757  007456  005077  171716                    CLR     @ABR            ;CLEAR THE PRESET BUFFER
6758  007462  005037  001124                    CLR     $GDDAT          ;CLEAR EXPED.
6759
6760  007466  012777  004000  171702            MOV     #BIT11,@ASR     ;DISABLE THE INTERNAL OSC.
6761  007474  052777  000011  171674            BIS     #BIT3!BIT0,@ASR ;START CLOCK:RATE 1MHZ.
6762  007502  005000                            CLR     R0
6763  007504  105200                    1$:     INCB    R0              ;DELAY A SHORT TIME.
6764  007506  001376                            BNE     1$
6765  007510  017746  171662                    MOV     @ASR,-(6)       ;/SAVE CSR
  (1)  007514  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
  (1)  007520  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (1)  007526  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1)  007534  013777  001424  171634            MOV     $TMP3,@ASR              ;/LOAD CSR.
  (1)                                                                   ;/THIS MUST BE DONE IN
  (1)                                                                   ;/ORDER TO XFERR COUNTER
  (1)                                                                   ;/TO BUFFER ON ST2.
  (1)  007542  052777  001000  171626            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (1)  007550  017737  171624  001126            MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (1)                                                                   ;/PREVIOUS COUNTER
```

```
   (1)  007556  012677  171614                 MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
   (1)  007562  005737  001126                 TST     $BDDAT          ;/RESTORE CSR
 6766   007566  001401                         BEQ     2$              ;NO - GOOD - NEXT TEST.
 6767
                                              ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6768   007570  104011                         ERROR   11              ;CLOCK DISABLE INTERNAL
 6769                                                                 ;OSC. DID NOT WORK.
 6770
                                              ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6771   007572  005077  171600         2$:     CLR     @ASR            ;CLEAR THE CSR.
 6772
 6773                                         ;;*****************************************************************
   (3)                                        ;*TEST 52       *TEST THAT CLOCK CAN BE COUNTED USING MAINTENANCE OSC
   (3)                                        ;;*****************************************************************
   (2)  007576  000004                 TST52:  SCOPE
   (2)
 6774
 6775   007600  005077  171572                 CLR     @ASR            ;CLEAR THE CSR.
 6776   007604  005077  171570                 CLR     @ABR            ;CLEAR THE PRESET BUFFER.
 6777   007610  052777  004000  171560         BIS     #BIT11,@ASR     ;DISABLE THE INTERNAL OSC.
 6778   007616  052777  000011  171552         BIS     #BIT3!BIT0,@ASR ;START CLOCK, 1MHZ, GO.
 6779   007624  012700  177754                 MOV     #-20.,R0        ;SET TO COUNT 20 TIMES
 6780
 6781   007630  052777  002000  171540  1$:    BIS     #BIT10,@ASR     ;GENERATE 1 MAINTENANCE OSC.
 6782                                                                 ;NOTE: AT 1MHZ, IT TAKES 10
 6783                                                                 ;MAINT. OSC TO EQUAL 1 COUNT
 6784   007636  005200                         INC     R0              ;DO 20 MAINTENANCE OSC.
 6785   007640  001373                         BNE     1$
 6786
 6787   007642  017746  171530                 MOV     @ASR,-(6)       ;/SAVE CSR
   (1)  007646  011637  001424                 MOV     (6),$TMP3       ;/GET CSR.
   (1)  007652  042737  177707  001424         BIC     #177707,$TMP3   ;/SAVE RATE BITS.
   (1)  007660  052737  004005  001424         BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
   (1)  007666  013777  001424  171502         MOV     $TMP3,@ASR              ;/LOAD CSR.
   (1)                                                                 ;/THIS MUST BE DONE IN
   (1)                                                                 ;/ORDER TO XFERR COUNTER
   (1)                                                                 ;/TO BUFFER ON ST2.
   (1)  007674  052777  001000  171474         BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
   (1)  007702  017737  171472  001126         MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
   (1)                                                                 ;/PREVIOUS COUNTER
   (1)  007710  012677  171462                 MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
   (1)  007714  005737  001126                 TST     $BDDAT          ;/RESTORE CSR
 6788   007720  001001                         BNE     2$              ;YES - NEXT TEST.
 6789
 6790
                                              ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6791   007722  104011                         ERROR   11              ;ERROR COULD NOT COUNT USING
 6792                                                                 ;MAINTENANCE OSC.
 6793
                                              ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6794   007724  005077  171446         2$:     CLR     @ASR            ;CLEAR THE CSR.
 6795
```

```
 6866
 6867
  (1)
  (1)
  (5)                                   ;;*****************************************************************
  (4)                                   ;*TEST 53       *TEST THE CLOCK'S 1MHZ DIVIDER
  (4)                                   ;;*****************************************************************
  (3)  007730  000004                   TST53:  SCOPE
  (3)
  (2)  007732  012737  000005  001160           MOV     #5,$TIMES       ;;DO 5 ITERATIONS
  (1)
  (1)  007740  005077  171432                   CLR     @ASR            ;/CLEAR THE CSR.
  (1)  007744  005077  171430                   CLR     @ABR            ;/CLEAR THE PRESET BUFFER.
  (1)  007750  052777  004000  171420           BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
  (1)  007756  052777  000011  171412           BIS     #1!10,@ASR      ;/ENABLE CLOCK, RATE:1MHZ
  (1)
  (1)
  (1)  007764  012700  177766           10$:    MOV     #-10.,R0        ;/SET TO GENERATE 10 OSC PULSES.
  (1)
  (1)  007770  052777  002000  171400   1$:     BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
  (1)  007776  005200                           INC     R0              ;/DONE 10 OSC PULSES?
  (1)  010000  001373                           BNE     1$              ;/NO - DO ANOTHER ONE.
  (1)
  (1)
  (1)  010002  012737  000001  001124   2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
  (2)  010010  017746  171362                   MOV     @ASR,-(6)       ;/SAVE CSR
  (2)  010014  011637  001424                   MOV     (6),$TMP3       ;/GET CSR.
  (2)  010020  042737  177707  001424           BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (2)  010026  052737  004005  001424           BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (2)  010034  013777  001424  171334           MOV     $TMP3,@ASR              ;/LOAD CSR.
  (2)                                                                   ;/THIS MUST BE DONE IN
  (2)                                                                   ;/ORDER TO XFERR COUNTER
  (2)                                                                   ;/TO BUFFER ON ST2.
  (2)  010042  052777  001000  171326           BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (2)  010050  017737  171324  001126           MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (2)                                                                   ;/PREVIOUS COUNTER
  (2)  010056  012677  171314                   MOV     (6)+,@ASR       ;/CONTENTS ARE '+ $BDDAT.
  (2)  010062  005737  001126                   TST     $BDDAT          ;/RESTORE CSR
  (1)  010066  013737  001124  001420           MOV     $GDDAT,$TMP0    ;/$TMP0 USED IN ERROR TYPEOUT.
  (1)
  (1)  010074  023737  001124  001126           CMP     $GDDAT,$BDDAT   ;/DID CLOCK ADVANCE ONCE?
  (1)
  (1)  010102  001402                           BEQ     3$              ;/YES - NEXT TEST.
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  010104  104011                           ERROR   11              ;/ERROR ON CLOCK1MHZ PULSE
  (1)                                                                   ;/NOT GENERATED WHEN 10
  (1)                                                                   ;/OSC PULSES GENERATED.
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  010106  000443                           BR      5$
  (1)  010110  012700  000011           3$:     MOV     #9.,R0  ;/GET THE NUMBER OF MORE OSC PULSES
  (1)                                                           ;/TO BE GENERATED.
  (1)
```

```
   (1)  010114  052777  002000  171254  4$:     BIS     #BIT10,@ASR     ;/GENERATE ANOTHER OSC PULSE.
   (1)  010122  005300                          DEC     R0              ;/WHAT WE WANT TO CHECK
   (1)  010124  001373                          BNE     4$              ;/1MHZ PULSE ON 9 OSC PULSES.
   (1)
   (1)
   (2)  010126  017746  171244                  MOV     @ASR,-(6)       ;/SAVE CSR
   (2)  010132  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
   (2)  010136  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
   (2)  010144  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
   (2)  010152  013777  001424  171216          MOV     $TMP3,@ASR              ;/LOAD CSR.
   (2)                                                  ;/THIS MUST BE DONE IN
   (2)                                                  ;/ORDER TO XFERR COUNTER
   (2)                                                  ;/TO BUFFER ON ST2.
   (2)  010160  052777  001000  171210          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
   (2)  010166  017737  171206  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
   (2)                                                  ;/PREVIOUS COUNTER
   (2)  010174  012677  171176                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
   (2)  010200  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
   (1)  010204  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/WAS ANOTHER 1MHZ PULSE GENERATED?
   (1)  010212  001401                          BEQ     5$              ;/NO - NEXT TEST.
   (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

   (1)  010214  104011                          ERROR   11              ;/WE SEEM TO HAVE GENERATED
   (1)                                                  ;/ANOTHER 1MHZ PULSE ON
   (1)                                                  ;/ONLY 9 MAINTENANCE
   (1)                                                  ;/OSC PULSES.
   (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

   (1)  010216  005077  171154          5$:     CLR     @ASR            ;/CLEAR THE CSR.
   (1)
6868
   (1)
   (1)
   (5)                                  ;;*****************************************************************
   (4)                                  ;*TEST 54        *TEST THE CLOCK'S 100KHZ DIVIDER
   (4)                                  ;;*****************************************************************
   (3)  010222  000004                  TST54:  SCOPE
   (3)
   (2)  010224  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
   (1)
   (1)  010232  005077  171140                  CLR     @ASR            ;/CLEAR THE CSR.
   (1)  010236  005077  171136                  CLR     @ABR            ;/CLEAR THE PRESET BUFFER.
   (1)  010242  052777  004000  171126          BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
   (1)  010250  052777  000021  171120          BIS     #1!20,@ASR      ;/ENABLE CLOCK, RATE:100KHZ
   (1)
   (1)
   (1)  010256  012700  177634          10$:    MOV     #-100.,R0       ;/SET TO GENERATE 100 OSC PULSES.
   (1)
   (1)  010262  052777  002000  171106  1$:     BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
   (1)  010270  005200                          INC     R0              ;/DONE 100 OSC PULSES?
   (1)  010272  001373                          BNE     1$              ;/NO - DO ANOTHER ONE.
   (1)
   (1)
   (1)  010274  012737  000001  001124  2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
```

```
(2)  010302  017746  171070                    MOV     @ASR,-(6)        ;/SAVE CSR
(2)  010306  011637  001424                    MOV     (6),$TMP3        ;/GET CSR.
(2)  010312  042737  177707  001424            BIC     #177707,$TMP3    ;/SAVE RATE BITS.
(2)  010320  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  010326  013777  001424  171042            MOV     $TMP3,@ASR               ;/LOAD CSR.
(2)                                                                     ;/THIS MUST BE DONE IN
(2)                                                                     ;/ORDER TO XFERR COUNTER
(2)                                                                     ;/TO BUFFER ON ST2.
(2)  010334  052777  001000  171034            BIS     #BIT9,@ASR       ;/GENERATE ON ST2 PULSE
(2)  010342  017737  171032  001126            MOV     @ABR,$BDDAT      ;/READ THE PRESET BUFFER,
(2)                                                                     ;/PREVIOUS COUNTER
(2)  010350  012677  171022                    MOV     (6)+,@ASR        ;/CONTENTS ARE IN $BDDAT.
(2)  010354  005737  001126                    TST     $BDDAT           ;/RESTORE CSR
(1)  010360  013737  001124  001420            MOV     $GDDAT,$TMP0     ;/$TMP0 USED IN ERROR TYPEOUT.
(1)
(1)  010366  023737  001124  001126            CMP     $GDDAT,$BDDAT    ;/DID CLOCK ADVANCE ONCE?
(1)
(1)  010374  001402                            BEQ     3$               ;/YES - NEXT TEST.
(2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  010376  104011                            ERROR   11               ;/ERROR ON CLOCK100KHZ PULSE
(1)                                                                     ;/NOT GENERATED WHEN 100
(1)                                                                     ;/OSC PULSES GENERATED.
(2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  010400  000443                            BR      5$
(1)  010402  012700  000143            3$:     MOV     #99.,R0 ;/GET THE NUMBER OF MORE OSC PULSES
(1)                                                                     ;/TO BE GENERATED.
(1)
(1)  010406  052777  002000  170762    4$:     BIS     #BIT10,@ASR      ;/GENERATE ANOTHER OSC PULSE.
(1)  010414  005300                            DEC     R0               ;/WHAT WE WANT TO CHECK
(1)  010416  001373                            BNE     4$               ;/100KHZ PULSE ON 99 OSC PULSES.
(1)
(1)
(2)  010420  017746  170752                    MOV     @ASR,-(6)        ;/SAVE CSR
(2)  010424  011637  001424                    MOV     (6),$TMP3        ;/GET CSR.
(2)  010430  042737  177707  001424            BIC     #177707,$TMP3    ;/SAVE RATE BITS.
(2)  010436  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  010444  013777  001424  170724            MOV     $TMP3,@ASR               ;/LOAD CSR.
(2)                                                                     ;/THIS MUST BE DONE IN
(2)                                                                     ;/ORDER TO XFERR COUNTER
(2)                                                                     ;/TO BUFFER ON ST2.
(2)  010452  052777  001000  170716            BIS     #BIT9,@ASR       ;/GENERATE ON ST2 PULSE
(2)  010460  017737  170714  001126            MOV     @ABR,$BDDAT      ;/READ THE PRESET BUFFER,
(2)                                                                     ;/PREVIOUS COUNTER
(2)  010466  012677  170704                    MOV     (6)+,@ASR        ;/CONTENTS ARE IN $BDDAT.
(2)  010472  005737  001126                    TST     $BDDAT           ;/RESTORE CSR
(1)  010476  023737  001124  001126            CMP     $GDDAT,$BDDAT    ;/WAS ANOTHER 100KHZ PULSE GENERATED?
(1)  010504  001401                            BEQ     5$               ;/NO - NEXT TEST.
(2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  010506  104011                            ERROR   11               ;/WE SEEM TO HAVE GENERATED
(1)                                                                     ;/ANOTHER 100KHZ PULSE ON
```

```
(1)                                                          ;/ONLY 99 MAINTENANCE
(1)                                                          ;/OSC PULSES.
(2)
                                  ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  010510  005077  170662          5$:     CLR     @ASR            ;/CLEAR THE CSR.
(1)

(1)
(1)
(5)                               ;;*************************************************************
(4)                               ;*TEST 55       *TEST THE CLOCK'S 10KHZ DIVIDER
(4)                               ;;*************************************************************
(3)  010514  000004               TST55:  SCOPE
(3)
(2)  010516  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)
(1)  010524  005077  170646                  CLR     @ASR            ;/CLEAR THE CSR.
(1)  010530  005077  170644                  CLR     @ABR            ;/CLEAR THE PRESET BUFFER.
(1)  010534  052777  004000  170634          BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
(1)  010542  052777  000031  170626          BIS     #1!30,@ASR      ;/ENABLE CLOCK, RATE:10KHZ
(1)
(1)
(1)  010550  012700  176030          10$:    MOV     #-1000.,R0      ;/SET TO GENERATE 1000 OSC PULSES.
(1)
(1)  010554  052777  002000  170614  1$:     BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
(1)  010562  005200                          INC     R0              ;/DONE 1000 OSC PULSES?
(1)  010564  001373                          BNE     1$              ;/NO - DO ANOTHER ONE.
(1)
(1)
(1)  010566  012737  000001  001124  2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
(2)  010574  017746  170576                  MOV     @ASR,-(6)       ;/SAVE CSR
(2)  010600  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
(2)  010604  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  010612  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  010620  013777  001424  170550          MOV     $TMP3,@ASR               ;/LOAD CSR.
(2)                                                          ;/THIS MUST BE DONE IN
(2)                                                          ;/ORDER TO XFERR COUNTER
(2)                                                          ;/TO BUFFER ON ST2.
(2)  010626  052777  001000  170542          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  010634  017737  170540  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                          ;/PREVIOUS COUNTER
(2)  010642  012677  170530                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  010646  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
(1)  010652  013737  001124  001420          MOV     $GDDAT,$TMP0    ;/$TMP0 USED IN ERROR TYPEOUT.
(1)
(1)  010660  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID CLOCK ADVANCE ONCE?
(1)
(1)  010666  001402                          BEQ     3$              ;/YES - NEXT TEST.
(2)
                                  ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  010670  104011                          ERROR   11              ;/ERROR ON CLOCK10KHZ PULSE
(1)                                                          ;/NOT GENERATED WHEN 1000
(1)                                                          ;/OSC PULSES GENERATED.
(2)
```

```
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  010672  000443                             BR      5$
(1)  010674  012700  001747          3$:       MOV     #999.,R0        ;/GET THE NUMBER OF MORE OSC PULSES
(1)                                                                    ;/TO BE GENERATED.
(1)
(1)  010700  052777  002000  170470  4$:       BIS     #BIT10,@ASR     ;/GENERATE ANOTHER OSC PULSE.
(1)  010706  005300                             DEC     R0              ;/WHAT WE WANT TO CHECK
(1)  010710  001373                             BNE     4$              ;/10KHZ PULSE ON 999 OSC PULSES.
(1)
(1)
(2)  010712  017746  170460                     MOV     @ASR,-(6)       ;/SAVE CSR
(2)  010716  011637  001424                     MOV     (6),$TMP3       ;/GET CSR.
(2)  010722  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  010730  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  010736  013777  001424  170432            MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                    ;/THIS MUST BE DONE IN
(2)                                                                    ;/ORDER TO XFERR COUNTER
(2)                                                                    ;/TO BUFFER ON ST2.
(2)  010744  052777  001000  170424            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  010752  017737  170422  001126            MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                                    ;/PREVIOUS COUNTER
(2)  010760  012677  170412                     MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  010764  005737  001126                     TST     $BDDAT          ;/RESTORE CSR
(1)  010770  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;/WAS ANOTHER 10KHZ PULSE GENERATED?
(1)  010776  001401                             BEQ     5$              ;/NO - NEXT TEST.
(2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011000  104011                             ERROR   11              ;/WE SEEM TO HAVE GENERATED
(1)                                                                    ;/ANOTHER 10KHZ PULSE ON
(1)                                                                    ;/ONLY 999 MAINTENANCE
(1)                                                                    ;/OSC PULSES.
(2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011002  005077  170370          5$:       CLR     @ASR            ;/CLEAR THE CSR.
(1)
6870
6871
(1)
(1)
(5)                                     ;;*************************************************************
(4)                                     ;*TEST 56        *TEST THE CLOCK'S 1KHZ DIVIDER
(4)                                     ;;*************************************************************
(3)  011006  000004                  TST56:  SCOPE
(3)
(2)  011010  012737  000005  001160            MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)
(1)  011016  005077  170354                     CLR     @ASR            ;/CLEAR THE CSR.
(1)  011022  005077  170352                     CLR     @ABR            ;/CLEAR THE PRESET BUFFER.
(1)  011026  052777  004000  170342            BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
(1)  011034  052777  000041  170334            BIS     #1!40,@ASR      ;/ENABLE CLOCK, RATE:1KHZ
(1)
(1)
(1)  011042  012700  154360          10$:      MOV     #-10000.,R0     ;/SET TO GENERATE 10000 OSC PULSES.
```

```
(1)
(1)  011046  052777  002000  170322  1$:     BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
(1)  011054  005200                          INC     R0              ;/DONE 10000 OSC PULSES?
(1)  011056  001373                          BNE     1$              ;/NO - DO ANOTHER ONE.
(1)
(1)
(1)  011060  012737  000001  001124  2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
(2)  011066  017746  170304                  MOV     @ASR,-(6)       ;/SAVE CSR
(2)  011072  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
(2)  011076  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  011104  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  011112  013777  001424  170256          MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                  ;/THIS MUST BE DONE IN
(2)                                                                  ;/ORDER TO XFERR COUNTER
(2)                                                                  ;/TO BUFFER ON ST2.
(2)  011120  052777  001000  170250          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  011126  017737  170246  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                                  ;/PREVIOUS COUNTER
(2)  011134  012677  170236                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  011140  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
(1)  011144  013737  001124  001420          MOV     $GDDAT,$TMP0    ;/$TMP0 USED IN ERROR TYPEOUT.
(1)
(1)  011152  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID CLOCK ADVANCE ONCE?
(1)
(1)  011160  001402                          BEQ     3$              ;/YES - NEXT TEST.
(2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011162  104011                          ERROR   11              ;/ERROR ON CLOCK1KHZ PULSE
(1)                                                                  ;/NOT GENERATED WHEN 10000
(1)                                                                  ;/OSC PULSES GENERATED.
(2)

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011164  000443                          BR      5$
(1)  011166  012700  023417          3$:     MOV     #9999.,R0       ;/GET THE NUMBER OF MORE OSC PULSES
(1)                                                                  ;/TO BE GENERATED.
(1)
(1)  011172  052777  002000  170176  4$:     BIS     #BIT10,@ASR     ;/GENERATE ANOTHER OSC PULSE.
(1)  011200  005300                          DEC     R0              ;/WHAT WE WANT TO CHECK
(1)  011202  001373                          BNE     4$              ;/1KHZ PULSE ON 9999 OSC PULSES.
(1)
(1)
(2)  011204  017746  170166                  MOV     @ASR,-(6)       ;/SAVE CSR
(2)  011210  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
(2)  011214  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  011222  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  011230  013777  001424  170140          MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                  ;/THIS MUST BE DONE IN
(2)                                                                  ;/ORDER TO XFERR COUNTER
(2)                                                                  ;/TO BUFFER ON ST2.
(2)  011236  052777  001000  170132          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  011244  017737  170130  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                                  ;/PREVIOUS COUNTER
(2)  011252  012677  170120                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  011256  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
```

```
(1)  011262  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/WAS ANOTHER 1KHZ PULSE GENERATED?
(1)  011270  001401                           BEQ     5$              ;/NO - NEXT TEST.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011272  104011                           ERROR   11              ;/WE SEEM TO HAVE GENERATED
(1)                                                                   ;/ANOTHER 1KHZ PULSE ON
(1)                                                                   ;/ONLY 9999 MAINTENANCE
(1)                                                                   ;/OSC PULSES.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011274  005077  170076           5$:     CLR     @ASR            ;/CLEAR THE CSR.
(1)
6872
(1)
(1)
(5)                                     ;;*****************************************************************
(4)                                     ;*TEST 57        *TEST THE CLOCK'S 100HZ DIVIDER
(4)                                     ;;*****************************************************************
(3)  011300  000004                    TST57:  SCOPE
(3)
(2)  011302  012737  000005  001160           MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)
(1)  011310  005077  170062                   CLR     @ASR            ;/CLEAR THE CSR.
(1)  011314  005077  170060                   CLR     @ABR            ;/CLEAR THE PRESET BUFFER.
(1)  011320  052777  004000  170050           BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
(1)  011326  052777  000051  170042           BIS     #1!50,@ASR      ;/ENABLE CLOCK, RATE:100HZ
(1)
(1)  011334  012701  000012                   MOV     #10.,R1
(1)
(1)  011340  012700  154360           10$:    MOV     #-10000.,R0     ;/SET TO GENERATE 10000 OSC PULSES.
(1)
(1)  011344  052777  002000  170024   1$:     BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
(1)  011352  005200                           INC     R0              ;/DONE 10000 OSC PULSES?
(1)  011354  001373                           BNE     1$              ;/NO - DO ANOTHER ONE.
(1)
(1)  011356  005301                           DEC     R1
(1)  011360  001367                           BNE     10$
(1)
(1)  011362  012737  000001  001124   2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
(2)  011370  017746  170002                   MOV     @ASR,-(6)       ;/SAVE CSR
(2)  011374  011637  001424                   MOV     (6),$TMP3       ;/GET CSR.
(2)  011400  042737  177707  001424           BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  011406  052737  004005  001424           BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  011414  013777  001424  167754           MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                   ;/THIS MUST BE DONE IN
(2)                                                                   ;/ORDER TO XFERR COUNTER
(2)                                                                   ;/TO BUFFER ON ST2.
(2)  011422  052777  001000  167746           BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  011430  017737  167744  001126           MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                                   ;/PREVIOUS COUNTER
(2)  011436  012677  167734                   MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  011442  005737  001126                   TST     $BDDAT          ;/RESTORE CSR
(1)  011446  013737  001124  001420           MOV     $GDDAT,$TMP0    ;/$TMP0 USED IN ERROR TYPEOUT.
(1)
```

```
    (1)  011454  023737  001124  001126          CMP     $GDDAT,$BDDAT     ;/DID CLOCK ADVANCE ONCE?
    (1)
    (1)  011462  001402                          BEQ     66$               ;/YES - NEXT TEST
    (2)
                                   ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  011464  104011                          ERROR   11                ;/ERROR ON CLOCK100HZ PULSE
    (1)                                                                    ;/NOT GENERATED WHEN 10000
    (1)                                                                    ;/OSC PULSES GENERATED.
    (2)
                                   ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  011466  000447                          BR      5$
    (1)  011470  012701  000012          66$:    MOV     #10.,R1
    (1)  011474  012700  023417          3$:     MOV     #9999.,R0         ;/GET THE NUMBER OF MORE OSC PULSES
    (1)                                                                    ;/TO BE GENERATED.
    (1)
    (1)  011500  052777  002000  167670  4$:     BIS     #BIT10,@ASR       ;/GENERATE ANOTHER OSC PULSE.
    (1)  011506  005300                          DEC     R0                ;/WHAT WE WANT TO CHECK
    (1)  011510  001373                          BNE     4$                ;/100HZ PULSE ON 9999 OSC PULSES.
    (1)  011512  005301                          DEC     R1
    (1)  011514  001367                          BNE     3$
    (1)
    (1)
    (2)  011516  017746  167654                  MOV     @ASR,-(6)         ;/SAVE CSR
    (2)  011522  011637  001424                  MOV     (6),$TMP3         ;/GET CSR.
    (2)  011526  042737  177707  001424          BIC     #177707,$TMP3     ;/SAVE RATE BITS.
    (2)  011534  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
    (2)  011542  013777  001424  167626          MOV     $TMP3,@ASR            ;/LOAD CSR.
    (2)                                                                    ;/THIS MUST BE DONE IN
    (2)                                                                    ;/ORDER TO XFERR COUNTER
    (2)                                                                    ;/TO BUFFER ON ST2.
    (2)  011550  052777  001000  167620          BIS     #BIT9,@ASR        ;/GENERATE ON ST2 PULSE
    (2)  011556  017737  167616  001126          MOV     @ABR,$BDDAT       ;/READ THE PRESET BUFFER,
    (2)                                                                    ;/PREVIOUS COUNTER
    (2)  011564  012677  167606                  MOV     (6)+,@ASR         ;/CONTENTS ARE IN $BDDAT.
    (2)  011570  005737  001126                  TST     $BDDAT            ;/RESTORE CSR
    (1)  011574  023737  001124  001126          CMP     $GDDAT,$BDDAT     ;/WAS ANOTHER 100HZ PULSE GENERATED?
    (1)  011602  001401                          BEQ     5$                ;/NO - NEXT TEST.
    (2)
                                   ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  011604  104011                          ERROR   11                ;/WE SEEM TO HAVE GENERATED
    (1)                                                                    ;/ANOTHER 100HZ PULSE ON
    (1)                                                                    ;/ONLY 9999 MAINTENANCE
    (1)                                                                    ;/OSC PULSES.
    (2)
                                   ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    (1)  011606  005077  167564          5$:     CLR     @ASR              ;/CLEAR THE CSR.
    (1)
   6873
   6891
```

```
 6893
 6894                                   ;;*****************************************************************
  (3)                                   ;*TEST 60       *TEST THE CLOCK'S MODE 2 OPERATION
  (4)                                           ;*
  (4)                                           ;*IN THIS TEST WE'LL CHECK MODE 2 OPERATION
  (4)                                           ;*MODE 2: EXTERNAL EVENTS TIMING MODE
  (4)                                           ;*SETTING THE GO BIT CAUSES THE COUNTER TO BEGIN COUNTING FROM
  (4)                                           ;*ZERO AND TO FREE-RUN UNTIL  THE GO BIT IS WRITTEN
  (4)                                           ;*TO A ZERO THE COUNTER WILL CONTINUE COUNTING AFTER
  (4)                                           ;*OVERFLOW. AN EXTERNAL PULSE FROM SCHMITZ TRIGGER 2
  (4)                                           ;*(WHEN ST2 GO ENABLE IS A '0') CAUSES DATA TO
  (4)                                           ;*TRANSFER FROM THE COUNTER TO THE BUFFER/PRESET REG.
  (4)                                           ;*WHILE THE COUNTER CONTINUES TO RUN.
  (4)                                           ;*
  (4)                                           ;*TO TEST THIS MODE, WE'LL DISABLE THE INTERNAL OSC AND USE
  (4)                                           ;*MAINTENANCE OSC PULSES AS WELL AS A MAINTENANCE
  (4)                                           ;*ST2.
  (4)                                           ;*
  (3)                                   ;*****************************************************************
  (2) 011612  000004                    TST60:  SCOPE
  (2)
  (1) 011614  012737  000020  001160            MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 6895
 6896 011622  005077  167550                    CLR     @ASR            ;CLEAR THE CSR.
 6897 011626  005077  167546                    CLR     @ABR            ;CLEAR THE PRESET REG.
 6898 011632  012777  004015  167536            MOV     #004015,@ASR    ;START CLOCK.
 6899
 6900 011640  012700  177754            1$:     MOV     #-20.,R0        ;SET TO GIVE 20 MAINTENANCE OSC.
 6901 011644  052777  002000  167524    2$:     BIS     #BIT10,@ASR     ;GENERATE A MAINTENANCE OSC.
 6902 011652  005200                            INC     R0
 6903 011654  001373                            BNE     2$              ;IF NOT DONE 20 TIMES, LOOP.
 6904
 6905 011656  052777  001000  167512  3$:     BIS     #BIT9,@ASR      ;HERE'S THE BIGGIE! AN ST2 HAS BEEN GENERATED
 6906 011664  012737  000002  001124            MOV     #2,$GDDAT       ;THE PRESET BUFFER SHOULD BE 2.
 6907 011672  017737  167502  001126            MOV     @ABR,$BDDAT     ;READ THE PRESET BUFFER.
 6908 011700  023737  001126  001124            CMP     $BDDAT,$GDDAT   ;DID A COUNTER TO PRESET BUFFER  CCUR?
 6909 011706  001402                            BEQ     4$              ;YES - NEXT SUBTEST.
 6910
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6911 011710  104005                            ERROR   5               ;A COUNTER TO PRESET BUFFER DID NOT
 6912                                           ;HAPPEN PROPERLY.
 6913
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6914 011712  000434                            BR      5$
 6915
 6916 011714                            4$:
  (1) 011714  017746  167456                    MOV     @ASR,-(6)       ;/SAVE CSR
  (1) 011720  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
  (1) 011724  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (1) 011732  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1) 011740  013777  001424  167430            MOV     $TMP3,@ASR              ;/LOAD CSR.
  (1)                                           ;/THIS MUST BE DONE IN
  (1)                                           ;/ORDER TO XFERR COUNTER
  (1)                                           ;/TO BUFFER ON ST2.
```

```
   (1)  011746  052777  001000  167422          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
   (1)  011754  017737  167420  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
   (1)                                                                  ;/PREVIOUS COUNTER
   (1)  011762  012677  167410                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
   (1)  011766  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
 6917   011772  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;WAS THE COUNTER ACCIDENTLY ZEROED?
 6918   012000  001401                          BEQ     5$              ;NO - NEXT TEST.
 6919
                                              ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6920   012002  104005                          ERROR   5               ;THE COUNT REGISTER SHOULD NOT
 6921                                                                  ;HAVE BEEN EFFECTED BY THE ST2
 6922                                                                  ;IN MODE 2.
 6923
                                              ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6924   012004                               5$:
 6925
 6933
 6934                                         ;;*****************************************************************
  (3)                                         ;*TEST 61       *TEST THE CLOCK'S MODE 3 OPERATION
  (4)                                         ;*
  (4)                                         ;*IN THIS TEST WE'LL CHECK MODE 3 OPERATION.
  (4)                                         ;*MODE 3 IS JUST LIKE MODE 2 EXCEPT THAT THE COUNT
  (4)                                         ;*REG IS ZEROED AFTER AN ST2.
  (4)                                         ;*
  (3)                                         ;*****************************************************************
  (2)  012004  000004                         TST61:  SCOPE
  (2)
  (1)  012006  012737  000020  001160          MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 6935
 6936   012014  005077  167356                  CLR     @ASR            ;CLEAR THE CSR.
 6937   012020  005077  167354                  CLR     @ABR            ;CLEAR THE BUFFER REG.
 6938   012024  012777  004017  167344          MOV     #4017,@ASR      ;START CLOCK.
 6939
 6940   012032  012700  177754          1$:     MOV     #-20.,R0        ;SET TO GIVE 20 MAINTENANCE OSC.
 6941   012036  052777  002000  167332  2$:     BIS     #BIT10,@ASR     ;GENERATE A MAINTENANCE OSC.
 6942   012044  005200                          INC     R0
 6943   012046  001373                          BNE     2$              ;IF NOT DONE 20 TIMES, LOOP.
 6944
 6945   012050  052777  001000  167320  3$:     BIS     #BIT9,@ASR      ;HERE'S THE BIGGIE! AN ST2 HAS BEEN GENERATED
 6946   012056  012737  000002  001124          MOV     #2,$GDDAT       ;THE PRESET BUFFER SHOULD BE 2.
 6947   012064  017737  16731C  001126          MOV     @ABR,$BDDAT     ;READ THE PRESET BUFFER.
 6948   012072  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;DID A COUNTER TO PRESET BUFFER OCCUR?
 6949   012100  001402                          BEQ     4$              ;YES - NEXT SUBTEST.
 6950
                                              ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6951   012102  104005                          ERROR   5               ;A COUNTER TO PRESET BUFFER DID NOT
 6952                                                                  ;HAPPEN PROPERLY.
 6953
                                              ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6954   012104  000445                          BR      TST62           ;;
 6955
 6956   012106  005037  001124          4$:     CLR     $GDDAT          ;EXPECT ZERO BACK FROM COUNT REG.
```

```
 6957  012112  017746  167260                 MOV     @ASR,-(6)       ;/SAVE CSR
  (1)  012116  011637  001424                 MOV     (6),$TMP3       ;/GET CSR.
  (1)  012122  042737  177707  001424         BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (1)  012130  052737  004005  001424         BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1)  012136  013777  001424  167232         MOV     $TMP3,@ASR              ;/LOAD CSR.
  (1)                                                                 ;/THIS MUST BE DONE IN
  (1)                                                                 ;/ORDER TO XFERR COUNTER
  (1)                                                                 ;/TO BUFFER ON ST2.
  (1)  012144  052777  001000  167224         BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (1)  012152  017737  167222  001126         MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (1)                                                                 ;/PREVIOUS COUNTER
  (1)  012160  012677  167212                 MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (1)  012164  005737  001126                 TST     $BDDAT          ;/RESTORE CSR
 6958  012170  001402                         BEQ     5$              ;IF SO - NEXT TEST.
 6959

                                    ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6960  012172  104005                         ERROR   5               ;THE CLOCK FORGOT TO ZERO THE COUNT
 6961                                                                ;REG. AFTER AN ST2 OCCURRED ON
 6962                                                                ;A MODE 3 COUNT.
 6963

                                    ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6964  012174  000411                         BR      TST62           ;;
 6965  012176                             5$:
 6966  012176  005077  167174                 CLR     @ASR            ;NOW TRY CLEARING THE CSR.
 6967  012202  017737  167170  001126         MOV     @ASR,$BDDAT     ;READ THE CSR - DID IT CLEAR?
 6968  012210  001403                         BEQ     TST62           ;;
 6969  012212  005037  001124                 CLR     $GDDAT          ;NO - RECORD S/B.
 6970

                                    ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6971  012216  104002                         ERROR   2               ;CSR FAILED TO CLEAR
 6972

                                    ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6973
 6974                               ;;*****************************************************************
  (3)                               ;*TEST 62       *IF ENABLED,CHECK THRESHOLD ST1 FROM TESTOR
  (3)                               ;;*****************************************************************
  (2)  012220  000004               TST62:  SCOPE
  (2)
 6975
 6976  012222  012737  000002  001160         MOV     #2,$TIMES       ;;DO 2 ITERATIONS.
 6977  012230  005737  001440                 TST     EXS             ;OPERATING IN TESTOR MODE?
 6978  012234  001002                         BNE     4$              ;YES DO THIS TEST.
 6979  012236  000137  013056                 JMP     ENDP            ;NO-END PASS
 6980  012242                             4$:
 6981
 6982  012242  005077  167130                 CLR     @ASR            ;YES-CLEAR CSR.
 6983  012246  012777  177775  167124         MOV     #-3,@ABR        ;SET TO COUNT THREE TIMES.
 6984  012254  012777  000061  167114         MOV     #61,@ASR        ;SET RATE: ST1,MODE 1 GO.
 6985  012262  104401  012270                 TYPE    ,65$            ;;TYPE ASCIZ STRING
  (1)  012266  000432                         BR      64$             ;;GET OVER THE ASCIZ
  (1)                               ;;65$:  .ASCIZ  <200><7><7>#SET ST1 THRESHOLD POT FULLY COUNTERCLOCKWIZE..#<7>
  (1)  012354                       64$:
```

```
 6986  012354  004737  013740                  JSR     PC,ANYKEY       ;TYPE LAST MESSAGE,AND WAIT FOR OPERATER.
 6987  012360  012737  177775  001124          MOV     #-3,$GDDAT      ;DON'T EXPECT COUNT TO CHANGE.
 6988  012366  017746  167004                  MOV     @ASR,-(6)       ;/SAVE CSR
  (1)  012372  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
  (1)  012376  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (1)  012404  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1)  012412  013777  001424  166756          MOV     $TMP3,@ASR              ;/LOAD CSR.
  (1)                                                                  ;/THIS MUST BE DONE IN
  (1)                                                                  ;/ORDER TO XFERR COUNTER
  (1)                                                                  ;/TO BUFFER ON ST2.
  (1)  012420  052777  001000  166750          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (1)  012426  017737  166746  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (1)                                                                  ;/PREVIOUS COUNTER
  (1)  012434  012677  166736                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (1)  012440  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
 6989  012444  001002                          BNE     2$      ;IF NON-ZERO,WE'RE OK.
 6990

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6991  012446  104002                          ERROR   ?               ;COUNTERCLOCKWIZE THRESHOLD SETTING
 6992                                                                 ;OF ST1 SHOULD HAVE INHIBITED
 6993                                                                 ;ST1 FROM CAUSEING CLOCK TO COUNT.
 6994

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 6995  012450  000463                          BR      TST63           ;;
 6996
 6997  012452                          2$:
 6998  012452  104401  012460                  TYPE    ,67$             ;;TYPE ASCIZ STRING
  (1)  012456  000425                          BR      66$             ;;GET OVER THE ASCIZ
  (1)                                  ;;67$:  .ASCIZ  <200><7><7>#SET ST1 THRESHOLD POT FULLY CLOCKWIZE#
  (1)  012532                          66$:
 6999  012532  004737  013740          3$:     JSR     PC,ANYKEY       ;TYPE LAST MESS. AND WAIT FOR OPERATOR.
 7000  012536  017746  166634                  MOV     @ASR,-(6)       ;/SAVE CSR
  (1)  012542  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
  (1)  012546  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (1)  012554  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1)  012562  013777  001424  166606          MOV     $TMP3,@ASR              ;/LOAD CSR.
  (1)                                                                  ;/THIS MUST BE DONE IN
  (1)                                                                  ;/ORDER TO XFERR COUNTER
  (1)                                                                  ;/TO BUFFER ON ST2.
  (1)  012570  052777  001000  166600          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (1)  012576  017737  166576  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (1)                                                                  ;/PREVIOUS COUNTER
  (1)  012604  012677  166566                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (1)  012610  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
 7001  012614  001001                          BNE     TST63           ;;
 7002

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 7003  012616  104002                          ERROR   2               ;CLOCKWIZE THRESHOLD SETTING OF
 7004                                                                 ;ST1 SHOULD HAVE INHIBITED CLOCK
 7005                                                                 ;FROM COUNTING.
 7006

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
 7007
 7008                                   ;;*****************************************************************
  (3)                                   ;*TEST 63       *IF ENABLED, CHECK ST1,ST2 IN FROM TESTOR
  (3)                                   ;;*****************************************************************
  (2)  012620  000004                   TST63:  SCOPE
  (2)
  (1)  012622  012737  000002  001160           MOV     #2,$TIMES       ;;DO 2 ITERATIONS
 7009
 7010  012630  005737  001440                   TST     EXS             ;OPERATING IN TESTOR MODE?
 7011  012634  001510                           BEQ     ENDP            ;NO-EXIT THIS TEST.
 7012
 7013  012636  005077  166534                   CLR     @ASR            ;CLEAR THE CSR
 7014  012642  012777  177775  166530           MOV     #-3,@ABR        ;SET TO COUNT THREE TIMES
 7015  012650  012777  000061  166520           MOV     #61,@ASR        ;SET RATE:ST1, MODE1, GO
 7016  012656  104401  012664                   TYPE    ,65$            ;;TYPE ASCIZ STRING
  (1)  012662  000433                           BR      64$             ;;GET OVER THE ASCIZ
  (1)                                   ;;65$:  .ASCIZ  <200><7><7>#SET ST1 THRESHOLD POT IN THE MIDDLE OF ITS RANGE.#
  (1)  012752                           64$:
 7017  012752  004737  013740                   JSR     PC,ANYKEY       ;TYPE LAST MESS WAITE FOR OPERATOR.
 7018  012756  005037  001124                   CLR     $GDDAT          ;EXPECT COUNTER TO BE CLEAR
 7019  012762  017746  166410                   MOV     @ASR,-(6)       ;/SAVE CSR
  (1)  012766  011637  001424                   MOV     (6),$TMP3       ;/GET CSR.
  (1)  012772  042737  177707  001424           BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (1)  013000  052737  004005  001424           BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1)  013006  013777  001424  166362           MOV     $TMP3,@ASR              ;/LOAD CSR.
  (1)                                                                   ;/THIS MUST BE DONE IN
  (1)                                                                   ;/ORDER TO XFERR COUNTER
  (1)                                                                   ;/TO BUFFER ON ST2.
  (1)  013014  052777  001000  166354           BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (1)  013022  017737  166352  001126           MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (1)                                                                   ;/PREVIOUS COUNTER
  (1)  013030  012677  166342                   MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (1)  013034  005737  001126                   TST     $BDDAT          ;/RESTORE CSR
 7020                                                                   ;OF NON-ZERO - REPORT ERROR
 7021  013040  001402                           BEQ     2$
 7022
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
 7023  013042  104002                           ERROR   2               ;INCORRECT # OF ST1'S
 7024                                                                   ;RECEIVED BY CLOCK
 7025
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 7026  013044  000404                           BR      ENDP
 7027
 7028  013046                           2$:
 7029  013046  005777  166324                   TST     @ASR            ;DID ST2 FLG SET?
 7030  013052  100401                           BMI     ENDP
 7031
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 7032  013054  104006                           ERROR   6               ;ST2 FLAG DID SET EVEN THOUGH
 7033                                                                   ;SWITCH WAS TOGGLED.
 7034
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
7035
7036                                        ;
7037                                        ;NOW WE'LL DETERMINE IF WE ARE ALLOWED TO AUTO-SIZE.
7038                                        ;IF SO, WE'LL FIND OUT IF THERE ARE OTHER CLOCKS OUT THERE
7039                                        ;TO TEST.
7040                                        ;
7041
7042  013056  000004              ENDP:    SCOPE
7043
7044
7045  013060  105737  001215              TSTB    $ENVM           ;SEE IF APT WILL LET UP AUTO-SIZE.
7046  013064  100537                       BMI     2$              ;NO - EXIT.
7047
7048
7049  013066  023737  001434  001436       CMP     MDEVCT,TSTCNT   ;TESTED MAX. UNITS?
7050  013074  001507                       BEQ     4$              ;YES EXIT.
7051  013076  006337  001426              ASL     ROTATE          ;POINT NEXT UNIT.
7052  013102  005237  001434              INC     MDEVCT
7053
7054  013106  062737  000004  001376       ADD     #4,ASR          ;YES, ADD TO BASE ADDR.
7055  013114  013746  000004              MOV     ERRVEC,-(6)     ;SAVE CONTENTS OF LOC 4.
7056  013120  012737  013274  000004       MOV     #1$,ERRVEC      ;SET UP IN CASE NO MORE CLOCKS.
7057
7058  013126  005777  166244              TST     @ASR            ;TIME OUT HERE IF NO MORE CLOCKS.
7059
7060                                                              ;IF HERE, ANOTHER CLOCK FOUND.
7061  013132  005737  001202              TST     $PASS           ;IS THIS 1ST PASS?
7062  013136  001003                       BNE     3$              ;NO-GET OUT.
7063  013140  053737  001426  001430       BIS     ROTATE,UTEST    ;YES-RECORD THIS UNIT.
7064  013146                      3$:
7065  013146  104401  013154              TYPE    ,65$            ;;TYPE ASCIZ STRING
 (1)  013152  000405                       BR      64$             ;;GET OVER THE ASCIZ
 (1)                              ;;65$:   .ASCIZ  <15><12>'UNIT #'
 (1)  013166                      64$:
7066  013166  013746  001204              MOV     $DEVCT,-(SP)    ;;SAVE $DEVCT FOR TYPEOUT
 (1)  013172  104402                       TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7067  013174  104401  013202              TYPE    ,67$            ;;TYPE ASCIZ STRING
 (1)  013200  000406                       BR      66$             ;;GET OVER THE ASCIZ
 (1)                              ;;67$:   .ASCIZ  " COMPLETED "
 (1)  013216                      66$:
7068  013216  005237  001204              INC     $DEVCT
7069  013222  104401  013230              TYPE    ,69$            ;;TYPE ASCIZ STRING
 (1)  013226  000410                       BR      68$             ;;GET OVER THE ASCIZ
 (1)                              ;;69$:   .ASCIZ  " TESTING UNIT #"
 (1)  013250                      68$:
7070  013250  013746  001204              MOV     $DEVCT,-(SP)    ;;SAVE $DEVCT FOR TYPEOUT
 (1)  013254  104402                       TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7071  013256  012637  000004              MOV     (6)+,ERRVEC     ;RESETORE LOC 4.
7072  013262  062737  000010  001402       ADD     #10,VECT1       ;UPDATE VECTOR ADDR.
7073  013270  000137  002334              JMP     LOOP            ;TEST NEW UNIT.
7074
7075  013274                      1$:
 (1)  013274  062706  000004              ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
7076  013300  012637  000004              MOV     (6)+,ERRVEC     ;RESTORE LOC 4
7077  013304  022737  000000  001204       CMP     #0,$DEVCT       ;TESTED ONLY ONE UNIT?
7078  013312  001424                       BEQ     2$              ;YES - NO NEED FOR TYPEOUT.
```

```
 7079
 7080  013314                                  4$:
 7081  013314  104401  013322                         TYPE    ,71$              ;;TYPE ASCIZ STRING
  (1)  013320  000405                               BR      70$               ;;GET OVER THE ASCIZ
  (1)                                       ;;71$:  .ASCIZ  <15><12>'UNIT #'
  (1)  013334                               70$:
 7082  013334  013746  001204                         MOV     $DEVCT,-(SP)      ;;SAVE $DEVCT FOR TYPEOUT
  (1)  013340  104402                               TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 7083  013342  104401  013350                         TYPE    ,73$              ;;TYPE ASCIZ STRING
  (1)  013346  000406                               BR      72$               ;;GET OVER THE ASCIZ
  (1)                                       ;;73$:  .ASCIZ  " COMPLETED "
  (1)  013364                               72$:
 7084
 7085  013364  013737  001250  001376       2$:     MOV     $BASE,ASR
 7086  013372  013737  001244  001402               MOV     $VECT1,VECT1
 7087  013400  013737  001204  001442               MOV     $DEVCT,LCNT
 7088  013406  005237  001442                       INC     LCNT
 7089  013412  012737  000000  001204               MOV     #0,$DEVCT
 7090
 7091  013420  005037  001434                       CLR     MDEVCT            ;BEGIN TESTING 1ST UNIT.
 7092  013424  012737  000001  001426               MOV     #1,ROTATE         ;POINT TO IT.
 7104
 7105
 7106                                      .SBTTL  END OF PASS ROUTINE
  (1)
  (2)                                      ;;*****************************************************************
  (1)                                      ;*INCREMENT THE PASS NUMBER ($PASS)
  (1)                                      ;*IF THERES A MONITOR GO TO IT
  (1)                                      ;*IF THERE ISN'T JUMP TO LOOP
  (1)
  (1)  013432                              $EOP:
  (2)  013432  000240                             NOP
  (1)  013434  005037  001102                     CLR     $TSTNM            ;;ZERO THE TEST NUMBER
  (1)  013440  005037  001160                     CLR     $TIMES            ;;ZERO THE NUMBER OF ITERATIONS
  (1)  013444  005237  001202                     INC     $PASS             ;;INCREMENT THE PASS NUMBER
  (1)  013450  042737  100000  001202             BIC     #100000,$PASS     ;;DON'T ALLOW A NEG. NUMBER
  (1)  013456  005327                             DEC     (PC)+             ;;LOOP?
  (1)  013460  000001                      $EOPCT: .WORD   1
  (1)  013462  003122                             BGT     $DOAGN            ;;YES
  (1)  013464  012737                             MOV     (PC)+,@(PC)+      ;;RESTORE COUNTER
  (1)  013466  000001                      $ENDCT: .WORD   1
  (1)  013470  013460                             $EOPCT
  (3)  013472  104401  013500                     TYPE    ,65$              ;;TYPE ASCIZ STRING
  (3)  013476  000406                             BR      64$               ;;GET OVER THE ASCIZ
  (3)                                      ;;65$:  .ASCIZ  <15><12>#ENDPASS  #
  (3)  013514                              64$:
  (3)  013514  013746  001202                     MOV     $PASS,-(SP)       ;;SAVE $PASS FOR TYPEOUT
  (3)  013520  104402                             TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
  (3)  013522  104401  013530                     TYPE    ,67$              ;;TYPE ASCIZ STRING
  (3)  013526  000411                             BR      66$               ;;GET OVER THE ASCIZ
  (3)                                      ;;67$:  .ASCIZ  #  TOTAL ERRORS  #
  (3)  013552                              66$:
  (3)  013552  013746  001432                     MOV     ERCNT,-(SP)       ;;SAVE ERCNT FOR TYPEOUT
  (3)  013556  104402                             TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
  (3)  013560  104401  013566                     TYPE    ,69$              ;;TYPE ASCIZ STRING
  (3)  013564  000407                             BR      68$               ;;GET OVER THE ASCIZ
```

```
   (3)                                  ;;69$:  .ASCIZ  #;  THERE ARE #
   (3)  013604                          68$:
   (3)  013604  013746  001442                  MOV     LCNT,-(SP)      ;;SAVE LCNT FOR TYPEOUT
   (3)  013610  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
   (3)  013612  104401  013620                  TYPE    ,71$            ;;TYPE ASCIZ STRING
   (3)  013616  000411                          BR      70$             ;;GET OVER THE ASCIZ
   (3)                                  ;;71$:  .ASCIZ  #  (OCTAL) UNITS.#
   (3)  013642                          70$:
   (3)  013642  104401  013650                  TYPE    ,73$            ;;TYPE ASCIZ STRING
   (3)  013646  000415                          BR      72$             ;;GET OVER THE ASCIZ
   (3)                                  ;;73$:  .ASCIZ  <200>#THE GOOD UNITS (L TO R) #
   (3)  013702                          72$:
   (3)  013702  013746  001430                  MOV     UTEST,-(SP)     ;;SAVE UTEST FOR TYPEOUT
   (3)  013706  104405                          TYPBN                   ;;GO TYPE--BINARY ASCII
   (1)  013710  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
   (1)  013714  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
   (1)  013716  000005                          RESET                   ;;CLEAR THE WORLD
   (1)  013720  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
   (1)  013722  000240                          NOP                     ;;SAVE ROOM
   (1)  013724  000240                          NOP                     ;;FOR
   (1)  013726  000240                          NOP                     ;;ACT11
   (1)  013730                          $DOAGN:
   (1)  013730  000137                          JMP     @(PC)+          ;;RETURN
   (1)  013732  002334                  $RTNAD: .WORD   LOOP
   (1)
   (1)
   (1)  013734     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
   (1)          013740                          .EVEN
  7107
  7108                                          ;
  7109                                          ;THIS ROUTINE TYPES LAST MESSAGE AND WAITS FOR AN  OPERATOR
  7110                                          ;RESPONCE.
  7111                                          ;
  7112
  7113  013740  105777  165202          ANYKEY: TSTB    @$TKB           ;CLEAR TTY READY FLAG.
  7114  013744  104401  013752                  TYPE    ,65$            ;;TYPE ASCIZ STRING
   (1)  013750  000430                          BR      64$             ;;GET OVER THE ASCIZ
   (1)                                  ;;65$:  .ASCIZ  <200><7>#SWITCH ST1 3 TIMES,TYPE ANY KEY WHEN DONE..#<7>
   (1)  014032                          64$:
  7115
  7116  014032  105777  165106          1$:     TSTB    @$TKS           ;WAIT FOR OPERATOR.
  7117  014036  100375                          BPL     1$
  7118  014040  105777  165102                  TSTB    @$TKB           ;CLEAR TTY READY FLAG.
  7119  014044  000207                          RTS PC
  7139
```

```
                                                  ;
                                   .SBTTL         ;I/O SIGNAL TEST #1 ST1 IN AND ST2 OUT IN AND OUT
                                                  ;

                                          ;
                                          ;SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:
                                          ;
                                          ;       SWITCH  1 - OFF
                                          ;               2 - ON
                                          ;               3 - OFF
                                          ;               4 - OFF
                                          ;               5 - ON
                                          ;               6 - ON
                                          ;               7 - NOT USED
                                          ;
                                          ; THIS SELECTS TTL THRESHOLDS AND POSITIVE SLOPE FOR
                                          ; SCHMITT TRIGGER 1.
                                          ;
                                          ;PLEASE REMOVE ANY PREVIOUS JUMPER.
                                          ;
                                          ;JUMPER THE FOLLOWING PINS TOGETHER:
                                          ;
                                          ;       J1 - SS (ST2 OUT)       TO J1 - VV (ST1-IN)
                                          ;
                                          ;LOAD AND START AT LOCATION 210
                                          ;END PASSES OCCUR IMMEDIATELY AND ARE NOT REPORTED
                                          ;ERRORS ARE REPORTED AS IN THE REGULAR LOGIC TEST AND
                                          ;THEIR PRINTOUT MAY BE INHIBITED

014046  012737  014060  001420  OITST1: MOV     #IOTST1,$TMPO       ;LOAD RETURN ADDRESS
014054  000137  001526                  JMP     INIT                ;PRIME THE PROGRAM VECTOR SPACE
014060  104407                  IOTST1: CKSWR                       ;CHECK THE SWR
014062  005077  165310          1$:     CLR     @ASR                ;CLEAR THE CSR
014066  005077  165306                  CLR     @ABR                ;CLEAR THE BUFFER REG.
014072  012777  000061  165276          MOV     #61,@ASR            ;RATE ST1, MODE 0, GO.
014100  052777  001000  165270          BIS     #BIT9,@ASR          ;GENERATE A MAINTENANCE ST2.
014106  012777  000005  165262          MOV     #5,@ASR             ;NOW SET TO READ COUNT REG
014114  052777  001000  165254          BIS     #BIT9,@ASR          ;FORCE COUNT -> BUFFER REG.
014122  027727  165252  000001          CMP     @ABR,#1             ;DID COUNT REG ADVANCE ONCE?
014130  001753                          BEQ     IOTST1              ;YES - LOOP.
014132  104000                          ERROR                       ;ST2 OUT TO ST1 IN FAILED.
014134  000751                          BR      IOTST1
```

```
                                        ;
                                .SBTTL  ;I/O SIGNAL TEST #2 CLOCK OVFLOW OUT TEST.
                                        ;
                                ;
                                ;SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:
                                ;
                                ;       SWITCH  1 - OFF
                                ;               2 - OFF
                                ;               3 - OFF
                                ;               4 - ON
                                ;               5 - OFF
                                ;               6 - ON
                                ;               7 - NOT USED
                                ;
                                ; THIS SELECTS TTL THRESHOLDS AND POSITIVE SLOPE FOR
                                ; SCHMITT TRIGGER 2.
                                ;
                                ;PLEASE REMOVE ANY PREVIOUS JUMPER.
                                ;
                                ;JUMPER THE FOLLOWING PINS TOGETHER:
                                ;
                                ;       J1 - RR (CLK OV) TO J1 - TT (ST2-IN)
                                ;
                                ;LOAD AND START AT LOCATION 214.
                                ;END PASSES OCCUR IMMEDIATELY AND ARE NOT REPORTED.
                                ;ERRORS ARE REPORTED AS IN TH REGULAR LOGIC TEST AND
                                ;THEIR PRINTOUT MAY BE INHIBITED.
                                ;
014136  012737  014150  001420  OITST2: MOV     #IOTST2,$TMPO   ;LOAD RETURN ADDRESS
014144  000137  001526                  JMP     INIT            ;PRIME THE PROGRAM VECTOR SPACE
014150  104407                  IOTST2: CKSWR                   ;CHECK THE SWR.
014152  005077  165220                  CLR     @ASR            ;CLEAR THE CSR.
014156  012777  177777  165214          MOV     #-1,@ABR        ;PRELOAD PRESET BUFFER.
014164  012777  000063  165204          MOV     #63,@ASR        ;RATE ST1, MODE 1, GO.
014172  052777  000400  165176          BIS     #BIT8,@ASR      ;GENERATE A MAIN. ST1.
014200  000240                          NOP
014202  000240                          NOP
014204  005777  165166                  TST     @ASR            ;DID OVERFLOW SET ST2 FLAG?
014210  100757                          BMI     IOTST2          ;YES - LOOP
014212  104000                          ERROR                   ;CLK OV OUT TO ST2 IN FAILED.
014214  000755                          BR      IOTST2          ;LOOP
```

```
7199
7200
7201                                    .SBTTL          ;
7202                                                    ;I/O SIGNAL TEST #3 ST1 OUT AND ST2 IN
7203                                                    ;
7204                                                    ;
7205                                                    ;SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:
7206                                                    ;       SWITCH  1 - OFF
7207                                                    ;               2 - OFF
7208                                                    ;               3 - OFF
7209                                                    ;               4 - ON
7210                                                    ;               5 - ON
7211                                                    ;               6 - ON
7212                                                    ;               7 - NOT USED
7213                                                    ; THIS SELECTS TTL THRESHOLD AND POSITIVE SLOPE FOR
7214                                                    ; SCHMITT TRIGGER 2.
7215                                                    ;
7216                                                    ;PLEASE REMOVE ANY PREVIOUS JUMPERS.
7217                                                    ;
7218                                                    ;JUMPER THE FOLLOWING PINS TOGETHER:
7219                                                    ;       J1 - UU (ST1 OUT)       TO J1 - TT (ST2-IN)
7220                                                    ;
7221                                                    ;LOAD AND START AT LOCATION 220
7222                                                    ;END PASSES OCCUR IMMEDIATELY AND ARE NOT REPORTED
7223                                                    ;ERRORS ARE REPORTED AS IN THE REGULAR LOGIC TEST AND
7224                                                    ;THEIR PRINTOUT MAY BE INHIBITED
7225
7226  014216  012737  014230  001420  OITST3: MOV     #IOTST3,$TMP0   ;LOAD RETURN ADDRESS
7227  014224  000137  001526                  JMP     INIT            ;PRIME THE PROGRAM VECTOR SPACE
7228  014230  104407                  IOTST3: CKSWR                   ;CHECK THE SWR
7229  014232  012777  000001  165136          MOV     #1,@ASR         ;SET GO BIT.
7230  014240  052777  000400  165130          BIS     #BIT8,@ASR      ;GENERATE A MAIN. ST1.
7231  014246  005777  165124                  TST     @ASR            ;DID ST2 FLAG SET?
7232  014252  100401                          BMI     1$
7233  014254  104000                          ERROR                   ;ST1 OUT TO ST2 IN FAILED
7234
7235  014256  032777  010000  165112  1$:     BIT     #BIT12,@ASR     ;DID "FOR" BIT SET?
7236  014264  001761                          BEQ     IOTST3          ;NO - GOOD!
7237  014266  104000                          ERROR                   ;"FOR" BIT SET ON ONLY 1 ST2.
7238  014270  000757                          BR      IOTST3          ;LOOP
```

```
 7240
 7241
 7242                                    .SBTTL
 7243                                    .SBTTL  *SYSMAC ROUTINES
 7244                                    .SBTTL
 7245
 7246                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
  (1)
  (2)                               ;;**************************************************************
  (1)                               ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
  (1)                               ;*OCTAL (ASCII) NUMBER AND TYPE IT.
  (1)                               ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
  (1)                               ;*CALL:
  (1)                               ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                               ;*      TYPOS                   ;;CALL FOR TYPEOUT
  (1)                               ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
  (1)                               ;*      .BYTE   M               ;;M=1 OR 0
  (1)                               ;*                                      ;;1=TYPE LEADING ZEROS
  (1)                               ;*                                      ;;0=SUPPRESS LEADING ZEROS
  (1)                               ;*
  (1)                               ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
  (1)                               ;*$TYPOS OR $TYPOC
  (1)                               ;*CALL:
  (1)                               ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                               ;*      TYPON                   ;;CALL FOR TYPEOUT
  (1)                               ;*
  (1)                               ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
  (1)                               ;*CALL:
  (1)                               ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                               ;*      TYPOC                   ;;CALL FOR TYPEOUT
  (1)
  (1)  014272  017646  000000       $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
  (1)  014276  116637  000001  014515       MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
  (1)  014304  112637  014517               MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
  (1)  014310  062716  000002               ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
  (1)  014314  000406                       BR      $TYPON
  (1)  014316  112737  000001  014515 $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
  (1)  014324  112737  000006  014517       MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
  (1)  014332  112737  000005  014514 $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
  (1)  014340  010346                       MOV     R3,-(SP)        ;;SAVE R3
  (1)  014342  010446                       MOV     R4,-(SP)        ;;SAVE R4
  (1)  014344  010546                       MOV     R5,-(SP)        ;;SAVE R5
  (1)  014346  113704  014517               MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
  (1)  014352  005404                       NEG     R4
  (1)  014354  062704  000006               ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
  (1)  014360  110437  014516               MOVB    R4,$OMODE       ;;SAVE IT FOR USE
  (1)  014364  113704  014515               MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
  (1)  014370  016605  000012               MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
  (1)  014374  005003                       CLR     R3              ;;CLEAR THE OUTPUT WORD
  (1)  014376  006105                 1$:   ROL     R5              ;;ROTATE MSB INTO "C"
  (1)  014400  000404                       BR      3$              ;;GO DO MSB
  (1)  014402  006105                 2$:   ROL     R5              ;;FORM THIS DIGIT
  (1)  014404  006105                       ROL     R5
  (1)  014406  006105                       ROL     R5
  (1)  014410  010503                       MOV     R5,R3
  (1)  014412  006103                 3$:   ROL     R3              ;;GET LSB OF THIS DIGIT
```

```
    (1)  014414  105337  014516                 DECB    $OMODE          ;;TYPE THIS DIGIT?
    (1)  014420  100016                         BPL     7$              ;;BR IF NO
    (1)  014422  042703  177770                 BIC     #177770,R3      ;;GET RID OF JUNK
    (1)  014426  001002                         BNE     4$              ;;TEST FOR 0
    (1)  014430  005704                         TST     R4              ;;SUPPRESS THIS 0?
    (1)  014432  001403                         BEQ     5$              ;;BR IF YES
    (1)  014434  005204                 4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
    (1)  014436  052703  000060                 BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
    (1)  014442  052703  000040         5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
    (1)  014446  110337  014512                 MOVB    R3,8$           ;;SAVE FOR TYPING
    (1)  014452  104401  014512                 TYPE    ,8$             ;;GO TYPE THIS DIGIT
    (1)  014456  105337  014514         7$:     DECB    $OCNT           ;;COUNT BY 1
    (1)  014462  003347                         BGT     2$              ;;BR IF MORE TO DO
    (1)  014464  002402                         BLT     6$              ;;BR IF DONE
    (1)  014466  005204                         INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
    (1)  014470  000744                         BR      2$              ;;GO DO THE LAST DIGIT
    (1)  014472  012605                 6$:     MOV     (SP)+,R5        ;;RESTORE R5
    (1)  014474  012604                         MOV     (SP)+,R4        ;;RESTORE R4
    (1)  014476  012603                         MOV     (SP)+,R3        ;;RESTORE R3
    (1)  014500  016666  000002  000004         MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
    (1)  014506  012616                         MOV     (SP)+,(SP)
    (1)  014510  000002                         RTI                     ;;RETURN
    (1)  014512     000                 8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
    (1)  014513     000                         .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
    (1)  014514     000                 $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
    (1)  014515     000                 $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
    (1)  014516  000000                 $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
 7247                                   .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
    (1)
    (2)                                 ;;**************************************************************
    (1)                                 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
    (1)                                 ;*BINARY-ASCII NUMBER AND TYPE IT.
    (1)                                 ;*CALL:
    (1)                                 ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
    (1)                                 ;*      TYPBN                   ;;TYPE IT
    (1)
    (1)  014520  010146                 $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
    (1)  014522  016601  000006                 MOV     6(SP),R1        ;;GET THE INPUT NUMBER
    (1)  014526  000261                         SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
    (1)  014530  112737  000060  014572 1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
    (1)  014536  006101                         ROL     R1              ;;GET THIS BIT
    (1)  014540  001406                         BEQ     2$              ;;DONE?
    (1)  014542  105537  014572                 ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
    (1)  014546  104401  014572                 TYPE    ,$BIN           ;;GO TYPE THIS BIT
    (1)  014552  000241                         CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
    (1)  014554  000765                         BR      1$              ;;GO DO THE NEXT BIT
    (1)  014556  012601                 2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
    (1)  014560  016666  000002  000004         MOV     2(SP),4(SP)     ;;ADJUST THE STACK
    (1)  014566  012616                         MOV     (SP)+,(SP)
    (1)  014570  000002                         RTI                     ;;RETURN TO USER
    (1)  014572     000     000         $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
 7248
 7259
 7260                                   .SBTTL  ERROR HANDLER ROUTINE
    (1)
    (2)                                 ;;**************************************************************
```

```
(1)                                     ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
(1)                                     ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(1)                                     ;*AND GO TO $ERRTYP ON ERROR
(1)                                     ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                                     ;*SW15=1        HALT ON ERROR
(1)                                     ;*SW13=1        INHIBIT ERROR TYPEOUTS
(1)                                     ;*SW10=1        BELL ON ERROR
(1)                                     ;*SW09=1        LOOP ON ERROR
(1)                                     ;*CALL
(1)                                     ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
(1)
(1)  014574                             $ERROR:
(1)  014574  104407                             CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
(1)  014576  105237  001103        7$:          INCB    $ERFLG                  ;;SET THE ERROR FLAG
(1)  014602  001775                             BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
(1)  014604  013777  001102  164330             MOV     $TSTNM,@DISPLAY         ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  014612  032777  002000  164320             BIT     #BIT10,@SWR             ;;BELL ON ERROR?
(1)  014620  001402                             BEQ     1$                      ;;NO - SKIP
(1)  014622  104401  001164                     TYPE    ,$BELL                  ;;RING BELL
(1)  014626  005237  001112        1$:          INC     $ERTTL                  ;;COUNT THE NUMBER OF ERRORS
(1)  014632  011637  001116                     MOV     (SP),$ERRPC             ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  014636  162737  000002  001116             SUB     #2,$ERRPC
(1)  014644  117737  164246  001114             MOVB    @$ERRPC,$ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  014652  032777  020000  164260             BIT     #BIT13,@SWR             ;;SKIP TYPEOUT IF SET
(1)  014660  001004                             BNE     20$                     ;;SKIP TYPEOUTS
(1)  014662  004737  015002                     JSR     PC,$ERRTYP              ;;GO TO USER ERROR ROUTINE
(1)  014666  104401  001171                     TYPE    ,$CRLF
(1)  014672                             20$:
(1)  014672  122737  000001  001214             CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
(1)  014700  001007                             BNE     2$                      ;;NO,SKIP APT ERROR REPORT
(1)  014702  113737  001114  014714             MOVB    $ITEMB,21$              ;;SET ITEM NUMBER AS ERROR NUMBER
(1)  014710  004737  016474                     JSR     PC,$ATY4                ;;REPORT FATAL ERROR TO APT
(1)  014714     000                     21$:    .BYTE   0
(1)  014715     000                             .BYTE   0
(1)  014716  000777                     22$:    BR      22$                     ;;APT ERROR LOOP
(1)  014720  005777  164214             2$:     TST     @SWR                    ;;HALT ON ERROR
(1)  014724  100002                             BPL     3$                      ;;SKIP IF CONTINUE
(1)  014726  000000                             HALT                            ;;HALT ON ERROR!
(1)  014730  104407                             CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
(1)  014732  032777  001000  164200     3$:     BIT     #BIT09,@SWR             ;;LOOP ON ERROR SWITCH SET?
(1)  014740  001402                             BEQ     4$                      ;;BR IF NO
(1)  014742  013716  001110                     MOV     $LPERR,(SP)             ;;FUDGE RETURN FOR LOOPING
(1)  014746  005737  001162             4$:     TST     $ESCAPE                 ;;CHECK FOR AN ESCAPE ADDRESS
(1)  014752  001402                             BEQ     5$                      ;;BR IF NONE
(1)  014754  013716  001162                     MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  014760                             5$:
(3)
(3)  014760  005237  001432                     INC     ERCNT                   ;/UPDATE ERROR COUNT.
(3)  014764  001002                             BNE     10$                     ;/BUT DON'T LET IT OVERFLOW.
(3)  014766  005337  001432                     DEC     ERCNT                   ;/KEEP AT 177777 IF OVERFLOW.
(3)  014772                             10$:
(3)  014772  043737  001426  001430             BIC     ROTATE,UTEST            ;/REMOVE UNIT FROM LIST OF GOOD ONES.
(3)  015000  000002                             RTI                             ;/EXIT.
(3)
7261                                    .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
(1)
```

```
(2)                                     ;;**********************************************************************
(1)                                     ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
(1)                                     ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
(1)                                     ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
(1)
(1)  015002                             $ERRTYP:
(1)  015002  104401  001171                     TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  015006  010046                             MOV     R0,-(SP)        ;;SAVE R0
(1)  015010  005000                             CLR     R0              ;;PICKUP THE ITEM INDEX
(1)  015012  153700  001114                     BISB    @#$ITEMB,R0
(1)  015016  001004                             BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                     ;;TYPE THE PC OF THE ERROR
(2)  015020  013746  001116                     MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                     ;;ERROR ADDRESS
(2)  015024  104402                             TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  015026  000426                             BR      6$              ;;GET OUT
(1)  015030  005300                     1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(1)  015032  006300                             ASL     R0              ;;      WORK FOR THE ERROR TABLE
(1)  015034  006300                             ASL     R0
(1)  015036  006300                             ASL     R0
(1)  015040  062700  001256                     ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
(1)  015044  012037  015054                     MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
(1)  015050  001404                             BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(1)  015052  104401                             TYPE                    ;;TYPE THE "ERROR MESSAGE"
(1)  015054  000000                     2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
(1)  015056  104401  001171                     TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  015062  012037  015072             3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
(1)  015066  001404                             BEQ     5$              ;;SKIP TYPEOUT IF 0
(1)  015070  104401                             TYPE                    ;;TYPE THE "DATA HEADER"
(1)  015072  000000                     4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
(1)  015074  104401  001171                     TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  015100  011000                     5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
(1)  015102  001004                             BNE     7$              ;;GO TYPE THE DATA
(1)  015104  012600                     6$:     MOV     (SP)+,R0        ;;RESTORE R0
(1)
(1)  015106  104401  001171                     TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  015112  000207                             RTS     PC              ;;RETURN
(1)  015114                             7$:
(2)  015114  013046                             MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
(2)  015116  104402                             TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  015120  005710                             TST     (R0)            ;;IS THERE ANOTHER NUMBER?
(1)  015122  001770                             BEQ     6$              ;;BR IF NO
(1)  015124  104401  015132                     TYPE    ,8$             ;;TYPE TWO(2) SPACES
(1)  015130  000771                             BR      7$              ;;LOOP
(1)  015132  020040      000           8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
(1)          015136                             .EVEN
7262                                    .SBTTL  SCOPE HANDLER ROUTINESTARS
 (1)                                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
 (1)                                    ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
 (1)                                    ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
 (1)                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 (1)                                    ;*SW14=1        LOOP ON TEST
 (1)                                    ;*SW11=1        INHIBIT ITERATIONS
 (1)                                    ;*SW09=1        LOOP ON ERROR
 (1)                                    ;*SW08=1        LOOP ON TEST IN SWR<7:0>
 (1)                                    ;*CALL
```

```
(1)                                     ;*      SCOPE           ;;SCOPE=IOT
(1)
(1)  015136                             $SCOPE:
(1)  015136  104407                             CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
(2)  015140  104407                             CKSWR
(1)  015142  032777  040000  163770     1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(1)  015150  001114                             BNE     $OVER           ;;YES IF SW14=1
(1)                                     ;#####START OF CODE FOR THE XOR TESTER#####
(1)  015152  000416                     $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                                                                     ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(1)  015154  013746  000004                     MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(1)  015160  012737  015200  000004             MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(1)  015166  005737  177060                     TST     @#177060        ;;TIME OUT ON XOR?
(1)  015172  012637  000004                     MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  015176  000463                             BR      $SVLAD          ;;GO TO THE NEXT TEST
(1)  015200  022626                     5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(1)  015202  012637  000004                     MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  015206  000423                             BR      7$              ;;LOOP ON THE PRESENT TEST
(1)  015210                             6$:;#####END OF CODE FOR THE XOR TESTER#####
(1)  015210  032777  000400  163722             BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
(1)  015216  001404                             BEQ     2$              ;;BR IF NO
(1)  015220  127737  163714  001102             CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
(1)  015226  001465                             BEQ     $OVER           ;;BR IF YES
(1)  015230  105737  001103             2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(1)  015234  001421                             BEQ     3$              ;;BR IF NO
(1)  015236  123737  001115  001103             CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
(1)  015244  101015                             BHI     3$              ;;BR IF NO
(1)  015246  032777  001000  163664             BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
(1)  015254  001404                             BEQ     4$              ;;BR IF NO
(1)  015256  013737  001110  001106     7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
(1)  015264  000446                             BR      $OVER
(1)  015266  105037  001103             4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  015272  005037  001160                     CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  015276  000415                             BR      1$              ;;ESCAPE TO THE NEXT TEST
(1)  015300  032777  004000  163632     3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
(1)  015306  001011                             BNE     1$              ;;BR IF YES
(1)  015310  005737  001202                     TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(1)  015314  001406                             BEQ     1$              ;;      INHIBIT ITERATIONS
(1)  015316  005237  001104                     INC     $ICNT           ;;INCREMENT ITERATION COUNT
(1)  015322  023737  001160  001104             CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(1)  015330  002024                             BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
(1)  015332  012737  000001  001104     1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(1)  015340  013737  015416  001160             MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(1)  015346  105237  001102             $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(1)  015352  113737  001102  001200             MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
(1)  015360  011637  001106                     MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  015364  011637  001110                     MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
(1)  015370  005037  001162                     CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)  015374  112737  000001  001115             MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1)  015402  013777  001102  163532     $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(1)  015410  013716  001106                     MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  015414  000002                             RTI                     ;;FIXES PS
(1)  015416  003720                     $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
7263                                    .SBTTL  TTY INPUT ROUTINE
 (1)
 (2)                                    ;;**************************************************************
```

```
(1)
(1)
(1)                                     .ENABL  LSB
(1)
(2)                                     ;;**********************************************************************
(1)                                     ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
(1)                                     ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
(1)                                     ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
(1)                                     ;*WHEN OPERATING IN TTY FLAG MODE.
(1)  015420  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
(1)  015426  001074                         BNE     15$             ;;BRANCH IF NO
(1)  015430  105777  163510                 TSTB    @$TKS           ;;CHAR THERE?
(1)  015434  100071                         BPL     15$             ;;IF NO, DON'T WAIT AROUND
(1)  015436  117746  163504                 MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
(1)  015442  042716  177600                 BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
(1)  015446  022726  000007                 CMP     #7,(SP)+        ;;IS IT A CONTROL G?
(1)  015452  001062                         BNE     15$             ;;NO, RETURN TO USER
(1)  015454  123727  001134  000001         CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
(1)  015462  001456                         BEQ     15$             ;;BRANCH IF YES
(1)
(1)  015464  104401  016145                 TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
(1)  015470  104401  016152          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
(2)  015474  013746  000176                 MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
(2)  015500  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  015502  104401  016163                 TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
(1)  015506  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
(1)  015510  005046                          CLR     -(SP)
(1)                                          ;;THE NEW SWR
(1)  015512  105777  163426          7$:     TSTB    @$TKS           ;;CHAR THERE?
(1)  015516  100375                         BPL     7$              ;;IF NOT TRY AGAIN
(1)
(1)  015520  117746  163422                 MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
(1)  015524  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)
(1)
(1)
(1)  015530  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
(1)  015534  001005                         BNE     10$             ;;BRANCH IF NOT
(1)  015536  104401  016140                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
(1)  015542  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
(1)  015546  000757                         BR      19$             ;;LET'S TRY IT AGAIN
(1)
(1)
(1)  015550  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
(1)  015554  001022                         BNE     16$             ;;BRANCH IF NO
(1)  015556  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
(1)  015562  001403                         BEQ     11$             ;;BRANCH IF YES
(1)  015564  016677  000002  163346         MOV     2(SP),@SWR      ;;SAVE NEW SWR
(1)  015572  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
(1)  015576  104401  001171          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
(1)  015602  123727  001135  000001         CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
(1)  015610  001003                         BNE     15$             ;;BRANCH IF NOT
(1)  015612  012777  000100  163324         MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1)  015620  000002                  15$:    RTI                     ;;RETURN
(1)  015622  004737  016406          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
(1)  015626  021627  000060                 CMP     (SP),#60        ;;CHAR < 0?
```

```
(1)  015632  002420                          BLT     18$                ;;BRANCH IF YES
(1)  015634  021627  000067                  CMP     (SP),#67           ;;CHAR > 7?
(1)  015640  003015                          BGT     18$                ;;BRANCH IF YES
(1)  015642  042726  000060                  BIC     #60,(SP)+          ;;STRIP-OFF ASCII
(1)  015646  005766  000002                  TST     2(SP)              ;;IS THIS THE FIRST CHAR
(1)  015652  001403                          BEQ     17$                ;;BRANCH IF YES
(1)  015654  006316                          ASL     (SP)               ;;NO, SHIFT PRESENT
(1)  015656  006316                          ASL     (SP)               ;;   CHAR OVER TO MAKE
(1)  015660  006316                          ASL     (SP)               ;;   ROOM FOR NEW ONE.
(1)  015662  005266  000002          17$:    INC     2(SP)              ;;KEEP COUNT OF CHAR
(1)  015666  056616  177776                  BIS     -2(SP),(SP)        ;;SET IN NEW CHAR
(1)  015672  000707                          BR      7$                 ;;GET THE NEXT ONE
(1)  015674  104401  001170          18$:    TYPE    ,$QUES             ;;TYPE ?<CR><LF>
(1)  015700  000720                          BR      20$                ;;SIMULATE CONTROL-U
(1)                                  .DSABL  LSB
(1)
(1)
(2)                                  ;;*****************************************************************
(1)                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                  ;*CALL:
(1)                                  ;*      RDCHR                      ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                  ;*      RETURN HERE                ;;CHARACTER IS ON THE STACK
(1)                                  ;*                                 ;;WITH PARITY BIT STRIPPED OFF
(1)                                  ;
(1)
(1)  015702  011646                  $RDCHR: MOV     (SP),-(SP)         ;;PUSH DOWN THE PC
(1)  015704  016666  000004  000002          MOV     4(SP),2(SP)        ;;SAVE THE PS
(1)  015712  105777  163226          1$:     TSTB    @$TKS              ;;WAIT FOR
(1)  015716  100375                          BPL     1$                 ;;A CHARACTER
(1)  015720  117766  163222  000004          MOVB    @$TKB,4(SP)        ;;READ THE TTY
(1)  015726  042766  177600  000004          BIC     #^C<177>,4(SP)     ;;GET RID OF JUNK IF ANY
(1)  015734  026627  000004  000023          CMP     4(SP),#23          ;;IS IT A CONTROL-S?
(1)  015742  001013                          BNE     3$                 ;;BRANCH IF NO
(1)  015744  105777  163174          2$:     TSTB    @$TKS              ;;WAIT FOR A CHARACTER
(1)  015750  100375                          BPL     2$                 ;;LOOP UNTIL ITS THERE
(1)  015752  117746  163170                  MOVB    @$TKB,-(SP)        ;;GET CHARACTER
(1)  015756  042716  177600                  BIC     #^C177,(SP)        ;;MAKE IT 7-BIT ASCII
(1)  015762  022627  000021                  CMP     (SP)+,#21          ;;IS IT A CONTROL-Q?
(1)  015766  001366                          BNE     2$                 ;;IF NOT DISCARD IT
(1)  015770  000750                          BR      1$                 ;;YES, RESUME
(1)  015772  026627  000004  000140  3$:     CMP     4(SP),#140         ;;IS IT UPPER CASE?
(1)  016000  002407                          BLT     4$                 ;;BRANCH IF YES
(1)  016002  026627  000004  000175          CMP     4(SP),#175         ;;IS IT A SPECIAL CHAR?
(1)  016010  003003                          BGT     4$                 ;;BRANCH IF YES
(1)  016012  042766  000040  000004          BIC     #40,4(SP)          ;;MAKE IT UPPER CASE
(1)  016020  000002                  4$:     RTI                        ;;GO BACK TO USER
(2)                                  ;;*****************************************************************
(1)                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1)                                  ;*CALL:
(1)                                  ;*      RDLIN                      ;;INPUT A STRING FROM THE TTY
(1)                                  ;*      RETURN HERE                ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1)                                  ;*                                 ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
(1)
(1)  016022  010346                  $RDLIN: MOV     R3,-(SP)           ;;SAVE R3
(1)  016024  012703  016130          1$:     MOV     #$TTYIN,R3         ;;GET ADDRESS
(1)  016030  022703  016140          2$:     CMP     #$TTYIN+8.,R3      ;;BUFFER FULL?
```

```
(1)  016034  101405                          BLOS    4$              ;;BR IF YES
(1)  016036  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
(1)  016040  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
(1)  016042  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
(1)  016046  001003                          BNE     3$              ;;SKIP IF NOT
(1)  016050  104401  001170          4$:     TYPE    ,$QUES          ;;TYPE A '?'
(1)  016054  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1)  016056  111337  016126          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
(1)  016062  104401  016126                  TYPE    ,9$
(1)  016066  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
(1)  016072  001356                          BNE     2$              ;;LOOP IF NOT RETURN
(1)  016074  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
(1)  016100  104401  001172                  TYPE    ,$LF            ;;TYPE A LINE FEED
(1)  016104  012603                          MOV     (SP)+,R3        ;;RESTORE R3
(1)  016106  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  016110  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
(1)  016116  012766  016130  000004          MOV     #$TTYIN,4(SP)
(1)  016124  000002                          RTI                     ;;RETURN
(1)  016126     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  016127     000                  .BYTE   0                       ;;TERMINATOR
(1)  016130  000010          $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
(1)  016140  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
(1)  016145     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
(1)  016152  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)  016160  020075     000
(1)  016163     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
(1)  016170  036440  000040
7264                                 .SBTTL  TYPE ROUTINE
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(1)                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(1)                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(1)                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(1)                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(1)                                  ;*
(1)                                  ;*CALL:
(1)                                  ;*1) USING A TRAP INSTRUCTION
(1)                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(1)                                  ;*OR
(1)                                  ;*      TYPE
(1)                                  ;*      MESADR
(1)                                  ;*
(1)
(1)  016174  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(1)  016200  100002                          BPL     1$              ;;BR IF YES
(1)  016202  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
(1)  016204  000430                          BR      3$              ;;LEAVE
(1)  016206  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
(1)  016210  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(1)  016214  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  016222  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
(1)  016224  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
(1)  016232  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(1)  016234  010037  016244                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(1)  016240  004737  016464                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
```

```
(1)  016244  000000                    61$:    .WORD   0               ;;MESSAGE ADDRESS
(1)  016246  132737  000040  001215    62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(1)  016254  001003                            BNE     60$             ;;YES,SKIP TYPE OUT
(1)  016256  112046                    2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  016260  001005                            BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  016262  005726                            TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  016264  012600                    60$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  016266  062716  000002            3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  016272  000002                            RTI                     ;;RETURN
(1)  016274  122716  000011            4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  016300  001430                            BEQ     8$
(1)  016302  122716  000200                    CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)  016306  001006                            BNE     5$
(1)  016310  005726                            TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  016312  104401                            TYPE                    ;;TYPE A CR AND LF
(1)  016314  001171                            $CRLF
(1)  016316  105037  016452                    CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)  016322  000755                            BR      2$              ;;GET NEXT CHARACTER
(1)  016324  004737  016406            5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  016330  123726  001156            6$:     CMPB    $FILIC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  016334  001350                            BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  016336  013746  001154                    MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                                                    ;;AND THE NULL CHAR.
(1)  016342  105366  000001            7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  016346  002770                            BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  016350  004737  016406                    JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  016354  105337  016452                    DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  016360  000770                            BR      7$              ;;LOOP
(1)
(1)                                    ;HORIZONTAL TAB PROCESSOR
(1)
(1)  016362  112716  000040            8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  016366  004737  016406            9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  016372  132737  000007  016452            BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  016400  001372                            BNE     9$              ;;TAB STOP
(1)  016402  005726                            TST     (SP)+           ;;POP SPACE OFF STACK
(1)  016404  000724                            BR      2$              ;;GET NEXT CHARACTER
(1)  016406  105777  162536            $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  016412  100375                            BPL     $TYPEC
(1)  016414  116677  000002  162530            MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)
(1)  016422  122766  000015  000002    CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  016430  001003                            BNE     1$              ;.BRANCH IF NO
(1)  016432  105037  016452                    CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  016436  000406                            BR      $TYPEX          ;;EXIT
(1)  016440  122766  000012  000002    1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  016446  001402                            BEQ     $TYPEX          ;;BRANCH IF YES
(1)  016450  105227                            INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  016452  000000                    $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(1)  016454  000207                    $TYPEX: RTS     PC
(1)
(1)
(1)
7265                                   .SBTTL  APT COMMUNICATIONS ROUTINE
(1)
(2)                                    ;;**************************************************************************
```

```
(1)  016456  112737  000001  016722  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
(1)  016464  112737  000001  016720  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
(1)  016472  000403                          BR      $ATYC
(1)  016474  112737  000001  016722  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
(1)  016502                          $ATYC:
(3)  016502  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  016504  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(1)  016506  105737  016720                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
(1)  016512  001450                          BEQ     5$              ;;IF NOT:  BR
(1)  016514  122737  000001  001214          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
(1)  016522  001031                          BNE     3$              ;;IF NOT:  BR
(1)  016524  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(1)  016532  001425                          BEQ     3$              ;;IF NOT:  BR
(1)  016534  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
(1)  016540  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(1)  016546  005737  001174          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
(1)  016552  001375                          BNE     1$              ;;IF NOT:  WAIT
(1)  016554  010037  001210                  MOV     R0,$MSGAD
(1)                                          ;;PUT ADDR IN MAILBOX
(1)  016560  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
(1)  016562  001376                          BNE     2$
(1)  016564  163700  001210                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
(1)  016570  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
(1)  016572  010037  001212                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
(1)  016576  012737  000004  001174          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
(1)  016604  000413                          BR      5$
(1)  016606  017637  000004  016632  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
(1)  016614  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
(3)  016622  013746  177776                  MOV     #177776,-(SP)   ;;PUSH 177776 ON STACK
(1)  016626  004737  016174                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
(1)  016632  000000                  4$:     .WORD   0
(1)  016634                          5$:
(1)  016634  105737  016722          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
(1)  016640  001416                          BEQ     12$             ;;IF NOT:  BR
(1)  016642  005737  001214                  TST     $ENV            ;;RUNNING UNDER APT?
(1)  016646  001413                          BEQ     12$             ;;IF NOT:  BR
(1)  016650  005737  001174          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
(1)  016654  001375                          BNE     11$             ;;IF NOT:  WAIT
(1)  016656  017637  000004  001176          MOV     @4(SP),$FATAL   ;;GET ERROR #
(1)  016664  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(1)  016672  005237  001174                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
(1)  016676  105037  016722          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
(1)  016702  105037  016721                  CLRB    $LFLG           ;;CLEAR LOG FLAG
(1)  016706  105037  016720                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
(3)  016712  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  016714  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  016716  000207                          RTS     PC              ;;RETURN
(1)  016720     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
(1)  016721     000          $LFLG:  .BYTE   0
(1)                                          ;;LOG FLAG
(1)  016722     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
(1)          016724                      .EVEN
(1)          000200          APTSIZE=200
(1)          000001          APTENV=001
(1)          000100          APTSPOOL=100
(1)          000040          APTCSUP=040
```

```
 7266                                   .SBTTL  POWER DOWN AND UP ROUTINES
  (1)
  (2)                                   ;;*****************************************************************
  (1)                                   ;POWER DOWN ROUTINE
  (1)  016724  012737  017064  000024   $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
  (1)  016732  012737  000300  000026          MOV     #PR6,@#PWRVEC+2 ;;PRIO:6
  (3)  016740  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)  016742  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (3)  016744  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (3)  016746  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (3)  016750  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (3)  016752  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
  (3)  016754  017746  162160                 MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
  (1)  016760  010637  017070                 MOV     SP,$SAVR6       ;;SAVE SP
  (1)  016764  012737  016776  000024         MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
  (1)  016772  000000                         HALT
  (1)  016774  000776                         BR      .-2             ;;HANG UP
  (1)
  (2)                                   ;;*****************************************************************
  (1)                                   ;POWER UP ROUTINE
  (1)  016776  012737  017064  000024   $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
  (1)  017004  013706  017070                 MOV     $SAVR6,SP       ;;GET SP
  (1)  017010  005037  017070                 CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
  (1)  017014  005237  017070          1$:    INC     $SAVR6          ;;WAIT FOR THE INC
  (1)  017020  001375                         BNE     1$              ;;OF  WORD
  (3)  017022  012677  162112                 MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
  (3)  017026  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
  (3)  017030  012604                         MOV     (SP)+,R4        ;;POP STACK INTO R4
  (3)  017032  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
  (3)  017034  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
  (3)  017036  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
  (3)  017040  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
  (1)  017042  012737  016724  000024         MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
  (1)  017050  012737  000300  000026         MOV     #PR6,@#PWRVEC+2 ;;PRIO:6
  (1)  017056  104401                         TYPE                    ;REPORT THE POWER FAILURE
  (1)  017060  017072                $PWRMG:  .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
  (1)  017062  000002                         RTI
  (1)  017064  000000                $!LLUP:  HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
  (1)  017066  000776                         BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
  (1)  017070  000000                $SAVR6:  0                       ;;PUT THE SP HERE
  (1)  017072  005015  047520  042527  $POWER: .ASCIZ  <15><12>'POWER"
  (1)  017100  000122
  (1)                                   .EVEN
 7267                                   ;*
 7268                                   ;*THIS ROUTINE WILL PROTECT THE PROGRAM
 7269                                   ;*FROM INTERUPTS (BAD ONES).
 7270                                   ;*
 7271                                   ;*THE TRAP CATCHER IS SET UP FOR
 7272                                   ;*      .WORD   .+2
 7273                                   ;*      JSR PC,R0
 7274                                   ;*
 7275                                   ;*ILLEGAL INTERRUPTS OR INTERRUPTS TO THE WRONG VECTOR
 7276                                   ;*GOTO THE VECTOR AND PCITK UP THE ".+2" AS AN ADDRESS
 7277
 7278                                   ;*AND "4700" AS NEW STATUS.
 7279                                   ;*THE .+2 AS A PC WILL CAUSE EXECUTION OF THE "JSR PC,R0" (AN ILLEGAL INSTR.).
```

```
                                        ;*AND TRAP TO LOCATION "4". IN LOCATION 4 WE HAVE A
                                        ;*POINTER HERE. IF THIS CONDITION CAUSES A TRAP TO LOC. 4.
                                        ;*WE WILL REPORT IT IN THE SAME MANNER THAT WER WOULD
                                        ;*REPORT ANY OTHER ERROR.
                                        ;*IF A BUSS ERROR TRAP DID OCCUT AND CAUSE A TRAP TO 4.
                                        ;*WE WILL HALT.

017102  011637  017146          IOTRD:  MOV     (6),TRTO        ;GET WHERE WE CAME TO.
017106  162737  000004  017146          SUB     #4,TRTO         ;FORM READ ADDR.

017114  023727  017146  001000          CMP     TRTO,#1000      ;DID TRAP FROM LESS THAN ADDR. 1000?
017122  003402                          BLE     2$              ;NO-CONTINUE.

017124  000000                  1$:     HALT                    ;A BUSS ERROR TIME OUT TRAP BROUGHT US HERE.
                                                                ;ADDRESS CONTAINED IN TRTO.

017126  000776                          BR      1$              ;DON'T ALLOW CONTINUE.

017130  016637  000004  017150  2$:     MOV     4(6),TRFRO      ;GET TRAPPED FROM ADDR.
017136  062706  000004                  ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

017142  104004                          ERROR   4               ;ERROR! ILLEGAL INTERRUPT OR
                                                                ;INTERRUPT TO WRONG VECTOR.
                                                                ;IF TEST NO. IS LESS THAN 10,ITS
                                                                ;LIKELY(BUT NO EXCLUSIVELY)TO BE A
                                                                ;DEVICE OTHER THAN THE DEVICE UNDER TEST.
                                                                ;IF THE INTERRUPT OCCURED
                                                                ;DURING AN INTERRUPT TEST, I'D
                                                                ;SUSPECT A PROBLEM WITH THE DEVICE UNDER TEST.
                                                                ;IF THE ADDRESS THE INTERRUPT
                                                                ;VECTORED TO IS WITHIN THE RANGE OF
                                                                ;VECTORS ASSIGNED TO THE DEVICE,
                                                                ;THEN I'D SUSPECT THE DEVICE
                                                                ;INTERRUPTD ILLEGALLY.
                                                                ;IF THE ADDRESS THE INTERRUPT
                                                                ;VECTORED TO IS OUTSIDE OF THE
                                                                ;RANGE ASSIGNED TO THE DEVICE
                                                                ;I'D SUSPECT THAT THE
                                                                ;DEVICE PUT THE WRONG INTERRUPT
                                                                ;VICTOR ON THE BUS DURING THE INTERRUPT
                                                                ;PROCESS.
                                                                ;       NOTE:
                                                                ;FOR THIS ERROR - DON'T USE
                                                                ; "LOOP ON ERROR" OPTION.
                                                                ;ALSO EXPECT THAT THE INTERRUPT TEST TO
                                                                ;WILL REPOT THAT THE DEVICE DIDN'T
                                                                ;INTERRUPT.
                                                                ;FOLLOW THE RECOMMENDED PROCEEDURE
                                                                ; IN THE DOCUMENT (ON THIS DIAGNOSTIC)
                                                                ;FOR LOOPING ON TEST.


                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
                                        RTI
                                TRTO:   .WORD   0               ;CONTAINS ADDR. WE TRAPPED OR INTERRUPTED TO.
                                TRFRO:  .WORD   0               ;CONTAINS ADDR. WE TRAPPED OR INTR. FROM.
                                .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.

017152  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
017154  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
017160  005740                          TST     -(R0)           ;;BACKUP BY 2
017162  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
017164  006300                          ASL     R0              ;;POSITION FOR INDEXING
017166  016000  017206                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
017172  000200                          RTS     R0              ;;GO TO ROUTINE


                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

017174  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
017176  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
017204  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE "TRAP" INSTRUCTION.
                                ;       ROUTINE
                                ;       -------
017206  017174                  $TRPAD: .WORD   $TRAP2
                                $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
                                $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
                                $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
                                $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
                                $TYPBN  ;;CALL=TYPBN    TRAP+5(104405)  TYPE BINARY (ASCII) NUMBER

                                $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

                                $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
                                $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
                                $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
017232  005015  046103  041517  EM1:    .ASCIZ  <15><12>/CLOCK SR FUNCTION ERROR/
017240  020113  051123  043040
017246  047125  052103  047511
017254  020116  051105  047522
017262  000122
017264  005015  046103  041517  EM2:    .ASCIZ  <15><12>/CLOCK SR DATA ERROR/
017272  020113  051123  042040
017300  052101  020101  051105
017306  047522  000122
017312  005015  046103  041517  EM3:    .ASCIZ  <15><12>/CLOCK BR DATA ERROR/
017320  020113  051102  042040
```

```
                017326  052101  020101  051105
                017334  047522  000122
  7339          017340  044600  052116  051105  EM4:    .ASCIZ  <200>/INTERRUPT ERROR/
                017346  052522  052120  042440
                017354  051122  051117     000
  7340          017361     015  041412  052517  EM5:    .ASCIZ  <15><12>/COUNT REG. ERROR/
                017366  052116  051040  043505
                017374  020056  051105  047522
                017402  000122
  7341          017404  005015  047503  047125  EM11:   .ASCIZ  <15><12>#COUNT ERROR #
                017412  020124  051105  047522
                017420  020122     000
  7342          017423     015  041412  052517  EM12:   .ASCIZ  <15><12>#COUNT FUNCTION ERROR#
                017430  052116  043040  047125
                017436  052103  047511  020116
                017444  051105  047522  000122
  7343          017452  005015  046103  041517  EM16:   .ASCIZ  <15><12>#CLOCK INTERRUPT ERROR #
                017460  020113  047111  042524
                017466  051122  050125  020124
                017474  051105  047522  020122
                017502     000
  7344          017503     015  051012  050105  EM20:   .ASCIZ  <15><12>#REPEATABILITY ERROR #
                017510  040505  040524  044502
                017516  044514  054524  042440
                017524  051122  051117  000040
  7345          017532  005015  042101  051104  EM26:   .ASCIZ  <15><12>#ADDRESSING ERROR#
                017540  051505  044523  043516
                017546  042440  051122  051117
                017554     000
  7346
  7347          017555     015  042412  051122  DH1:    .ASCIZ  <15><12>#ERRPC  ASR     WAS     S/B#
                017562  041520  040411  051123
                017570  053411  051501  051411
                017576  041057     000
  7348          017601     015  042412  051122  DH3:    .ASCIZ  <15><12>#ERRPC  ABR     WAS     S/B#
                017606  041520  040411  051102
                017614  053411  051501  051411
                017622  041057     000
  7349          017625     200  051105  050122  DH4A:   .ASCIZ  <200>#ERRPC   TO      FROM ADDR.#
                017632  020103  020040  047524
                017640  020040  020040  020040
                017646  051106  046517  040440
                017654  042104  027122     000
  7350          017661     015  042412  051122  DH12:   .ASCIZ  <15><12>#ERRPC  ASR     #
                017666  041520  040411  051123
                017674  000011
  7351          017676  005015  051105  050122  DH20:   .ASCIZ  <15><12>#ERRPC  ASR     2NDCNT  1STNCT  3RDCNT#
                017704  004503  051501  004522
                017712  047062  041504  052116
                017720  030411  052123  041516
                017726  004524  051063  041504
                017734  052116     000
  7352          017737     015  042412  051122  DH26:   .ASCIZ  <15><12>#ERRPC  CLOCK ADDR.#
                017744  041520  041411  047514
                017752  045503  040440  042104
                017760  027122     000
```

```
7353
7354                    017764                          .EVEN
7355
7356    017764  001116  001376  001126  DT1:    .WORD   $ERRPC,ASR,$BDDAT,$GDDAT,0
        017772  001124  000000
7357    017776  001116  001400  001126  DT3:    .WORD   $ERRPC,ABR,$BDDAT,$GDDAT,0
        020004  001124  000000
7358    020010  001116  017146  017150  DT4:    .WORD   $ERRPC,TRTO,TRFRO,0
        020016  000000
7359    020020  001116  001376  000000  DT12:   .WORD   $ERRPC,ASR,0
7360    020026  001116  001376  001126  DT20:   .WORD   $ERRPC,ASR,$BDDAT,$GDDAT,$TMP0,0
        020034  001124  001420  000000
7361    020042  001116  001376  001126  DT22:   .WORD   $ERRPC,ASR,$BDDAT,$TMP0,0
        020050  001420  000000
7362    020054  001116  001420  000000  DT26:   .WORD   $ERRPC,$TMP0,0
7363
7364    020062  000000  000000          DF0:    .WORD   0,0
7365
7366                                    ;;THE FOLLOWING MACRO CALL TO CNMAC2.SML LIBRARY WAS ADDED TO INIT 11/21
7367                                    ;;SPECIFIC VECTORS.
7368                    020066          POINT=.                 ;SAVE POINTER
 (1)                    000100          .=100
 (1)    000100  020066                  $CLKVEC                 ;LKVEC  HANDLER
 (1)    000102  000300                  300                     ;INTERRUPT HANDLER PRI
 (1)                    000140          .=140                   ;BRKVEC
 (1)    000140  170000                  170000                  ;ODT START ADDRESS
 (1)    000142  000300                  300                     ;PRIORITY
 (1)                    020066          .=POINT                 ;RESTORE POINTER
 (1)    020066  104401  020074          $CLKVEC:        TYPE,CLKMES
 (1)    020072  000000                  HALT
 (1)    020074  005015  045514  042526  CLKMES:  .ASCIZ <15><12>/LKVEC INTERRUPT - DISCONNECT LTC /
 (1)    020102  020103  047111  042524
 (1)    020110  051122  050125  020124
 (1)    020116  020055  044504  041523
 (1)    020124  047117  042516  052103
 (1)    020132  046040  041524  000040
7369                    000001                  .END
```

```
ABASE = 174420          5776#   5786    5863    5864
ABR     001400          5864#   5930*   5931*   5969    6060*   6063*   6142*   6150    6169*   6177    6241*   6245    6311*
                        6315    6336*   6354    6364*   6380*   6410*   6430*   6493*   6496*   6499*   6502*   6505*   6508*
                        6515*   6525    6541*   6574*   6715*   6757*   6765    6776*   6787    6867*   6868*   6869*   6871*
                        6872*   6897*   6907    6916    6937*   6947    6957    6983*   6988    7000    7014*   7019    7159*
                        7164    7188*   7357
ACDW1 = 000000          5786
ACDW2 = 000000          5786
ACPUOP= 000000          5786
ADDW0 = 000000          5786
ADDW1 = 000000          5786
ADDW10= 000000          5786
ADDW11= 000000          5786
ADDW12= 000000          5786
ADDW13= 000000          5786
ADDW14= 000000          5786
ADDW15= 000000          5786
ADDW2 = 000000          5786
ADDW3 = 000000          5786
ADDW4 = 000000          5786
ADDW5 = 000000          5786
ADDW6 = 000000          5786
ADDW7 = 000000          5786
ADDW8 = 000000          5786
ADDW9 = 000000          5786
ADEVCT= 000000          5786
ADEVM = 000000          5786
AENV  = 000000          5786
AENVM = 000000          5786
AFATAL= 000000          5786
AMADR1= 000000          5786
AMADR2= 000000          5786
AMADR3= 000000          5786
AMADR4= 000000          5786
AMAMS1= 000000          5786
AMAMS2= 000000          5786
AMAMS3= 000000          5786
AMAMS4= 000000          5786
AMSGAD= 000000          5786
AMSGLG= 000000          5786
AMSGTY= 000000          5786
AMTYP1= 000000          5786
AMTYP2= 000000          5786
AMTYP3= 000000          5786
AMTYP4= 000000          5786
ANYKEY  013740          6986    6999    7017    7113#
APASS = 000000          5786
APRIOR= 000200          5778#   5786    5869
APTCSU= 000040          7264    7265#
APTENV= 000001          7260    7264    7265#
APTSIZ= 000200          5899    7265#
APTSPO= 000100          7264    7265#
ASR     001376          5863#   5906*   5930    5968    6016*   6017*   6018*   6019*   6020*   6021*   6022*   6023*   6024*
                        6025*   6083*   6084*   6089    6110*   6112*   6116*   6122*   6124    6141*   6146*   6150*   6168*
                        6173*   6177*   6203*   6207    6263*   6264*   6266    6289*   6290*   6291    6300*   6310*   6312*
                        6313*   6315*   6335*   6340*   6342*   6354*   6363*   6379*   6382*   6384*   6386    6408*   6412*
```

```
                        6414*  6418   6429*  6431*  6433*  6435   6442*  6493*  6496*  6499*  6502*  6505*  6508*
                        6514*  6517*  6519*  6525*  6527   6535*  6540*  6542*  6547   6573*  6576*  6577*  6588*
                        6594*  6596*  6597*  6607*  6696*  6697*  6698*  6699*  6701   6704   6714*  6716*  6722
                        6725   6734*  6735*  6736*  6738*  6740*  6743   6752*  6756*  6760*  6761*  6765*  6771*
                        6775*  6777*  6778*  6781*  6787*  6794*  6967*  6868*  6869*  6871*  6872*  6896*  6898*
                        6901*  6905*  6916*  6936*  6938*  6941*  6945*  6957*  6966*  6967   6982*  6984*  6988*
                        7000*  7013*  7015*  7019*  7029   7054*  7058   7085*  7158*  7160*  7161*  7162*  7163*
                        7187*  7189*  7190*  7193   7229*  7230*  7231   7235   7356   7359   7360   7361
ASWREG= 000000          5786
ATESTN= 000000          5786
AUNIT = 000000          5786
AUSWR = 000000          5786
AVECT1= 000240          5777#  5786   5865   5866   5867   5868
AVECT2= 000000          5786
BIT0  = 000001          5774#  6025   6146   6150   6173   6177   6312   6315   6340   6354   6382   6412   6418
                        6435   6493   6496   6499   6502   6505   6508   6517   6525   6596   6705   6734   6738
                        6740   6761   6765   6778   6787   6867   6868   6869   6871   6872   6916   6957   6988
                        7000   7019
BIT00 = 000001          5774#  6291
BIT01 = 000002          5774#
BIT02 = 000004          5774#
BIT03 = 000010          5774#
BIT04 = 000020          5774#
BIT05 = 000040          5774#
BIT06 = 000100          5774#
BIT07 = 000200          5774#
BIT08 = 000400          5774#  7262
BIT09 = 001000          5774#  7260   7262
BIT1  = 000002          5774#  6024   6346
BIT10 = 002000          5774#  6221   6257   6781   6867   6868   6869   6871   6872   6901   6941   7260
BIT11 = 004000          5774#  6018   6150   6177   6221   6257   6315   6354   6493   6496   6499   6502   6505
                        6508   6525   6760   6765   6777   6787   6867   6868   6869   6871   6872   6916   6957
                        6988   7000   7019   7262
BIT12 = 010000          5774#  6221   6257   6508   6701   6705   6722   6746   7235
BIT13 = 020000          5774#  6017   6289   7260
BIT14 = 040000          5774#  6016   6596   7262
BIT15 = 100000          5774#  6705
BIT2  = 000004          5774#  6023   6150   6177   6277   6315   6354   6493   6496   6499   6502   6505   6508
                        6525   6765   6787   6867   6868   6869   6871   6872   6916   6957   6988   7000   7019
BIT3  = 000010          5774#  6022   6761   6778
BIT4  = 000020          5774#  6021   6312   6340   6382   6412
BIT5  = 000040          5774#  6020   6312   6340   6376   6382   6412
BIT6  = 000100          5774#  6019
BIT7  = 000200          5774#
BIT8  = 000400          5774#  6313   6342   6384   6414   6433   6519   6577   7190   7230
BIT9  = 001000          5774#  6150   6177   6264   6290   6315   6354   6493   6496   6499   6502   6505   6508
                        6525   6597   6698   6699   6735   6736   6765   6787   6867   6868   6869   6871   6872
                        6905   6916   6945   6957   6988   7000   7019   7161   7163
BPTVEC= 000014          5774#
BRKVEC= 000140          5774#
CKSWR = 104407          7157   7186   7228   7260   7262   7335#
CLKMES  020074          7368#
CR    = 000015          5774#  7264
CRLF  = 000200          5774#  5908   7264
DDISP = 177570          5774#  5786   5899
DFO     020062          5798   5805   5812   5819   5826   5833   5840   5847   5854   5861   7364#
```

```
DH1      017555            5796    5803    5824    5852    7347#
DH12     017661            5831    5838    7350#
DH20     017676            5845    7351#
DH26     017737            5859    7352#
DH3      017601            5810    7348#
DH4A     017625            5817    7349#
DISPLA   001142            5786#   5899*   7260*   7262*
DISPRE   000174            5752#   5899
DR       001414            5870#   6222    6277    6346
DR2      001416            5871#   6221*   6257*
DSWR   = 177570            5774#   5786    5899
DT1      017764            5797    5804    5825    7356#
DT12     020020            5832    5839    7359#
DT20     020026            5846    7360#
DT22     020042            5853    7361#
DT26     020054            5860    7362#
DT3      017776            5811    7357#
DT4      020010            5818    7358#
EMTVEC=  000030            5774#   5899*
EM1      017232            5795    7336#
EM11     017404            5851    7341#
EM12     017423            5830    7342#
EM16     017452            5837    7343#
EM2      017264            5802    7337#
EM20     017503            5844    7344#
EM26     017532            5858    7345#
EM3      017312            5809    7338#
EM4      017340            5816    7339#
EM5      017361            5823    7340#
ENDP     013056            6979    7011    7026    7030    7042#
ERCNT    001432            5877#   5915*   7106    7260*
ERRVEC=  000004            5774#   5899*   5904*   5968*   5969*   7055    7056*   7071*   7076*   7262*
EXS      001440            5880#   5888*   5893*   5897*   5910    6219    6255    6275    6344    6371    6393    6977    7010
GNS    = ****** U          5908    5912    5913    6985    6998    7016    7065    7067    7069    7081    7083    7106    7114
                           7335
GTSWR  = 104406            5908    7335#
HT     = 000011            5774#   7264
INIT     001526            5886    5899#   7156    7185    7227
IOTRD    017102            5750    5904    7287#
IOTST1   014060            7155    7157#   7165    7167
IOTST2   014150            7184    7186#   7194    7196
IOTST3   014230            7226    7228#   7236    7238
IOTVEC=  000020            5774#   5899*
LCNT     001442            5881#   7087*   7088*   7106
LF     = 000012            5774#   7264
LKVEC  = 000100            5774#
LOOP     002334            5919#   7073    7106
MDEVCT   001434            5878#   5914*   7049    7052*   7091*
ODTST  = 170000            5774#
OITST1   014046            5762    7155#
OITST2   014136            5763    7184#
OITST3   014216            5764    7226#
PIRQ   = 177772            5774#
PIRQVE=  000240            5774#
POINT  = 020066            7368#
PRIOR    001412            5869#
```

```
PR0    = 000000          5774#
PR1    = 000040          5774#
PR2    = 000100          5774#
PR3    = 000140          5774#
PR4    = 000200          5774#
PR5    = 000240          5774#
PR6    = 000300          5774#   5899    7266
PR7    = 000340          5774#
PS     = 177776          5774#
PSW    = 177776          5774#
PWRVEC= 000024           5774#   5899*   7266*
QSTART  001444           5761    5885#
RDCHR  = 104410          7263    7335#
RDLIN  = 104411          7335#
RESVEC= 000010           5774#
ROTATE  001426           5875#   5917*   5918    7051*   7063    7092*   7260
RSTART  002144           5885    5909#
STACK  = 001100          5774#   5899
START  = 001506          5760    5895#
STKLMT= 177774           5774#
SWR     001140           5786#   5899*   5908    6508    7260    7262    7263*   7266*
SWREG   000176           5753#   5899    5908    7263
SW0    = 000001          5774#
SW00   = 000001          5774#
SW01   = 000002          5774#
SW02   = 000004          5774#
SW03   = 000010          5774#
SW04   = 000020          5774#
SW05   = 000040          5774#
SW06   = 000100          5774#
SW07   = 000200          5774#
SW08   = 000400          5774#
SW09   = 001000          5774#
SW1    = 000002          5774#
SW10   = 002000          5774#
SW11   = 004000          5774#
SW12   = 010000          5774#
SW13   = 020000          5774#
SW14   = 040000          5774#
SW15   = 100000          5774#
SW2    = 000004          5774#
SW3    = 000010          5774#
SW4    = 000020          5774#
SW5    = 000040          5774#
SW6    = 000100          5774#
SW7    = 000200          5774#
SW8    = 000400          5774#
SW9    = 001000          5774#
TBITVE= 000014           5774#
TKVEC  = 000060          5774#
TPVEC  = 000064          5774#
TRAPVE= 000034           5774#   5899*
TRFRO   017150           7298*   7334#   7358
TRTO    017146           7287*   7288*   7290    7333#   7358
TRTVEC= 000014           5774#
TSTCNT  001436           5879#   5889*   5892*   5896*   7049
```

```
TSTSTR  001456            5769    5888#
TST1    002422            5968#
TST10   003244            6021#
TST11   003342            6022#
TST12   003440            6023#
TST13   003536            6024#
TST14   003634            6025#
TST15   003732            6060#
TST16   004022            6063#
TST17   004112            6081#
TST2    002514            5969#
TST20   004164            6108#
TST21   004246            6139#
TST22   004374            6166#
TST23   004522            6200#
TST24   004610            6217    6220    6223    6238#
TST25   004662            6256    6261#
TST26   004732            6274    6276    6278    6287#
TST27   004766            6308#
TST3    002556            6016#
TST30   005076            6333#
TST31   005306            6352    6369#
TST32   005422            6392    6394    6397    6406#
TST33   005464            6427#
TST34   005532            6493#
TST35   005662            6496#
TST36   006012            6499#
TST37   006142            6502#
TST4    002654            6017#
TST40   006272            6505#
TST41   006422            6508#
TST42   006562            6508    6512#
TST43   006724            6538#
TST44   006764            6569#
TST45   007104            6590#
TST46   007210            6694#
TST47   007264            6712#
TST5    002752            6018#
TST50   007350            6732#
TST51   007442            6754#
TST52   007576            6773#
TST53   007730            6867#
TST54   010222            6868#
TST55   010514            6869#
TST56   011006            6871#
TST57   011300            6872#
TST6    003050            6019#
TST60   011612            6894#
TST61   012004            6934#
TST62   012220            6954    6964    6968    6974#
TST63   012620            6995    7001    7008#
TST7    003146            6020#
TYPBN = 104405            7106    7335#
TYPE  = 104401            5908    5912    5913    6985    6998    7016    7065    7067    7069    7081    7083    7106    7114
                          7246    7247    7260    7261    7263    7264    7266    7335#   7368
TYPOC = 104402            7066    7070    7082    7106    7261    7263    7335#
```

```
TYPON = 104404           7335#
TYPOS = 104403           7335#
UTEST   001430           5876#   5918*   7063*   7106    7260*
VECTP   001404           5866#   5922*   5923*
VECT1   001402           5865#   5905*   5921*   5922    5924    6578*   7072*   7086*
VECT2   001406           5867#   5924*   5925*   5926    6595*
VECT2P  001410           5868#   5926*   5927*
WSTART= 001472           5767    5891#
ZSTART  001774           5898    5902#
$APTHD  001000           5785#
$ASTAT= ****** U         7265
$ATYC   016502           7265#
$ATY1   016456           7265#
$ATY3   016464           7264    7265#
$ATY4   016474           7260    7265#
$AUTOB  001134           5786#   5908*   7263
$BASE   001250           5786#   5906    7085
$BDADR  001122           5786#
$BDDAT  001126           5786#   6016*   6017*   6018*   6019*   6020*   6021*   6022*   6023*   6024*   6025*   6060*   6063*
                         6089*   6091    6094    6124*   6125    6127    6150*   6152    6177*   6179    6207*   6245*   6315*
                         6354*   6356    6493*   6496*   6499*   6502*   6505*   6508*   6525*   6704*   6725*   6743*   6746
                         6765*   6787*   6867*   6868*   6869*   6871*   6872*   6907*   6908    6916*   6917    6947*   6948
                         6957*   6967*   6988*   7000*   7019*   7356    7357    7360    7361
$BELL   001164           5786#   7260
$BIN    014572           7247#*
$CDW1   001254           5786#
$CHARC  016452           7264#*
$CKSWR  015420           7263#   7335
$CLKVE  020066           7368#
$CMTAG  001100           5786#   5899
$CM3  = 000000           5786#
$CNTLG  016145           7263#
$CNTLU  016140           7263#
$CPUOP  001222           5786#
$CRLF   001171           5786#   7260    7261    7263    7264
$DEVCT  001204           5786#   5903*   7066    7068*   7070    7077    7082    7087    7089*
$DEVM   001252           5786#
$DOAGN  013730           7106#
$ENDAD  013720           5783    7106#
$ENDCT  013466           7106#
$ENULL  013734           7106#
$ENV    001214           5786#   5908    7260    7264    7265
$ENVM   001215           5786#   5899    7045    7264    7265
$EOP    013432           7106#
$EOPCT  013460           7106#
$ERFLG  001103           5786#   7260*   7262*
$ERMAX  001115           5786#   5899*   7262*
$ERROR  014574           5899    7260#
$ERRPC  001116           5786#   7260*   7261    7356    7357    7358    7359    7360    7361    7362
$ERRTB  001256           5786#   7261
$ERRTY  015002           7260    7261#
$ERTTL  001112           5786#   7260*
$ESCAP  001162           5786#   5899*   7260    7262*
$ETABL  001214           5786#
$ETEND  001256           5785    5786#
$FATAL  001176           5786#   7265*
```

```
$FFLG   016722          7265#*
$FILLC  001156          5786#   7264
$FILLS  001155          5786#   7264
$GDADR  001120          5786#
$GDDAT  001124          5786#   6016*   6017*   6018*   6019*   6020*   6021*   6022*   6023*   6024*   6025*   6060*   6063*
                        6091*   6092*   6125*   6126*   6148*   6152    6175*   6179    6202*   6240*   6337*   6355*   6356
                        6364    6365    6530*   6705*   6726*   6745*   6758*   6867*   6868*   6869*   6871*   6872*   6906*
                        6908    6917    6946*   6948    6956*   6969*   6987*   7018*   7356    7357    7360
$GET42  013710          7106#
$GTSWR  015470          7263#   7335
$HD   = 000001          5728
$HIBTS  0C1000          5785#
$ICNT   001104          5786#   7262*
$ILLUP  017064          7266#
$INTAG  001135          5786#   7263
$ITEMB  001114          5786#   7260*   7261
$LF     001172          5786#   7260    7263    7264
$LFLG   016721          7265#*
$LPADR  001106          5786#   5899*   5968*   7262*
$LPERR  001110          5786#   5899*   5968*   6338*   7260    7262*
$MADR1  001226          5786#
$MADR2  001232          5786#
$MADR3  001236          5786#
$MADR4  001242          5786#
$MAIL   001174          5785    5786#   5899    5908    7260    7262    7264
$MAMS1  001224          5786#
$MAMS2  001230          5786#
$MAMS3  001234          5786#
$MAMS4  001240          5786#
$MBADR  001002          5785#
$MFLG   016720          7265#*
$MNEW   016163          7263#
$MSGAD  001210          5786#   7265*
$MSGLG  001212          5786#   7265*
$MSGTY  001174          5786#   7265*
$MSWR   016152          7263#
$MTYP1  001225          5786#
$MTYP2  001231          5786#
$MTYP3  001235          5786#
$MTYP4  001241          5786#
$MXCNT  015416          7262#
$NULL   001154          5786#   7264
$NWTST= 000001          5968#   5969#   6016#   6017#   6018#   6019#   6020#   6021#   6022#   6023#   6024#   6025#   6060#
                        6063#   6081#   6108#   6139#   6166#   6200#   6238#   6261#   6287#   6308#   6333#   6369#   6406#
                        6427#   6493#   6496#   6499#   6502#   6505#   6508#   6512#   6538#   6569#   6590#   6694#   6712#
                        6732#   6754#   6773#   6867#   6868#   6869#   6871#   6872#   6894#   6934#   6974#   7008#
$OCNT   014514          7246#*
$OMODE  014516          7246#*
$OVER   015402          7262#
$PASS   001202          5786#   5899*   5916*   7061    7106*   7262
$PASTM  001006          5785#
$POWER  017072          7266#
$PWRDN  016724          5899    7266#
$PWRMG  017060          7266#
$PWRUP  016776          7266#
$QUES   001170          5786#   7260    7263    7264
```

```
$RDCHR  015702            7263#   7335
$RDDEC= ****** U          7335
$RDLIN  016022            7263#   7335
$RDOCT= ****** U          7335
$RDSZ = 000010            7263#
$RTNAD  013732            7106#
$R2A  = ****** U          7335
$SAVRE= ****** U          7335
$SAVR6  017070            7266#*
$SCOPE  015136            5899    7262#
$SETUP= 000117            5884#   5899    5908    7106    7260    7262    7263
$STUP = 177777            5884#
$SVLAD  015346            7262#
$SVPC = 000244            5783#
$SWR  = 167400            5683#   5728    5730    5780#   5786    5899    5968    5969    6016    6017    6018    6019    6020
                          6021    6022    6023    6024    6025    6060    6063    6081    6108    6139    6166    6200    6238
                          6261    6287    6308    6333    6369    6406    6427    6493    6496    6499    6502    6505    6508
                          6512    6538    6569    6590    6694    6712    6732    6754    6773    6867    6868    6869    6871
                          6872    6894    6934    6974    7008    7106    7260    7262    7266
$SWREG  001216            5786#   5899
$SWRMK= 000000            5730    7262
$TESTN  001200            5786#   7262*
$TIMES  001160            5786#   5899*   5968*   6016*   6017*   6018*   6019*   6020*   6021*   6022*   6023*   6024*   6025*
                          6081*   6108*   6139*   6166*   6200*   6238*   6333*   6493*   6496*   6499*   6502*   6505*   6508*
                          6512*   6569*   6712*   6754*   6867*   6868*   6869*   6871*   6872*   6894*   6934*   6976*   7008*
                          7106*   7262*
$TKB    001146            5786#   7113    7118    7263
$TKS    001144            5786#   7116    7263*
$TMP0   001420            5872#   5885*   5898*   5900    5968*   5969*   6867*   6868*   6869*   6871*   6872*   7155*   7184*
                          7226*   7360    7361    7362
$TMP1   001422            5873#
$TMP3   001424            5874#   6150*   6177*   6315*   6354*   6493*   6496*   6499*   6502*   6505*   6508*   6525*   6765*
                          6787*   6867*   6868*   6869*   6871*   6872*   6916*   6957*   6988*   7000*   7019*
$TN   = 000064            5728#   5781#   5968#   5969#   6016#   6017#   6018#   6019#   6020#   6021#   6022#   6023#   6024#
                          6025#   6060#   6063#   6081#   6108#   6139#   6166#   6200#   6217    6220    6223    6238#   6256
                          6261#   6274    6276    6278    6287#   6308#   6333#   6352    6369#   6392    6394    6397    6406#
                          6427#   6493#   6496#   6499#   6502#   6505#   6508#   6512#   6538#   6569#   6590#   6694#   6712#
                          6732#   6754#   6773#   6867#   6868#   6869#   6871#   6872#   6894#   6934#   6954    6964    6968
                          6974#   6995    7001    7008#
$TPB    001152            5786#   7264*
$TPFLG  001157            5786#   7264
$TPS    001150            5786#   7264
$TRAP   017152            5899    7335#
$TRAP2  017174            7335#
$TRP  = 000012            7335#
$TRPAD  017206            7335#
$TSTM   001004            5785#
$TSTNM  001102            5786#   5968*   7106*   7260    7262*
$TTYIN  016130            7263#
$TYPBN  014520            7247#   7335
$TYPDS= ****** U          7335
$TYPE   016174            7264#   7265    7335
$TYPEC  016406            7263    7264#
$TYPEX  016454            7264#
$TYPOC  014316            7246#   7335
$TYPON  014332            7246#   7335
```

```
$TYPOS  014272            7246#   7335
$UNIT   001206            5786#
$UNITM  001010            5785#
$USWR   001220            5786#
$VECT1  001244            5786#   5905    7086
$VECT2  001246            5786#
$XTSTR  015152            7262#
$$GET4= 000000            7106#
$OFILL  014515            7246#*
$40CAT= ****** U          7260    7262
.       = 020140          5734#   5745    5746#   5749#   5751#   5754#   5759#   5766#   5768#   5783#   5784#   5785#   5786#
                          5891    5895    5899    5912#   6985#   6998#   7016#   7065#   7081#   7106#   7114#   7260    7261#
                          7262    7263#   7264    7265#   7266    7354#   7368#
.$ASTA= ****** U          7265
.$X     = 001000          5785#
```

```
ADTST    5932#  5968   5969
BUFLO    6027#  6060   6063
COMMEN   1566#  5774#
COUNTM   6444#  6493   6496   6499   6502   6505   6508
CSRDTA   5978#  6016   6017   6018   6019   6020   6021   6022   6023   6024   6025
DFC      5788#  5798   5805   5812   5819   5826   5833   5840   5847   5854   5861
DIVCH    6796#  6867   6868   6869   6871   6872
ECB      5686#  5968   5969   6016   6017   6018   6019   6020   6021   6022   6023   6024   6025   6060   6063
         6098   6103   6130   6134   6156   6161   6183   6188   6211   6216   6224   6227   6249   6253   6269
         6273   6279   6282   6294   6299   6317   6320   6348   6351   6358   6360   6389   6400   6403   6421
         6437   6440   6493   6496   6499   6502   6505   6508   6531   6534   6550   6553   6583   6585   6602
         6604   6706   6709   6727   6730   6748   6751   6767   6770   6790   6793   6867   6868   6869   6871
         6872   6910   6913   6919   6923   6950   6953   6959   6963   6970   6972   6990   6994   7002   7006
         7022   7025   7031   7034   7300   7331
ENDCOM   1578#  5774#
ENDPAS   7093#  7106
ERROR    5774#  5968   5969   6016   6017   6018   6019   6020   6021   6022   6023   6024   6025   6060   6063
         6099   6131   6157   6184   6213   6225   6250   6270   6280   6295   6318   6349   6359   6390   6401
         6422   6438   6493   6496   6499   6502   6505   6508   6532   6551   6584   6603   6707   6728   6749
         6768   6791   6867   6868   6869   6871   6872   6911   6920   6951   6960   6971   6991   7003   7023
         7032   7166   7195   7233   7237   7301
ESCAPE   1698#  5774#
GETPRI   1312#  5774#
GETSWR   1771#  5774#  5908#
INSTR2   7249#  7260
LOCKM    6561#  6571   6579   6582   6592   6598   6601
MULT     4484#  5774#
NEWTST   1626#  5774#  5968   5969   6016   6017   6018   6019   6020   6021   6022   6023   6024   6025   6060
         6063   6081   6108   6139   6166   6200   6238   6261   6287   6308   6333   6369   6406   6427   6493
         6496   6499   6502   6505   6508   6512   6538   6569   6590   6694   6712   6732   6754   6773   6867
         6868   6869   6871   6872   6894   6934   6974   7008
POP      2153#  5774#  7265   7266
POPSP2   5698#  5968   5969   6587   6606   7075   7299
PR       5702#  5902
PUSH     2145#  5774#  7265   7266
RDCLK    5711#  6150   6177   6315   6354   6493   6496   6499   6502   6505   6508   6525   6765   6787   6867
         6868   6869   6871   6872   6916   6957   6988   7000   7019
REPORT   5463#  5774#
SCOPE    5774#  5969   6016   6017   6018   6019   6020   6021   6022   6023   6024   6025   6060   6063   6081
         6108   6139   6166   6200   6238   6261   6287   6308   6333   6369   6406   6427   6493   6496   6499
         6502   6505   6508   6512   6538   6569   6590   6694   6712   6732   6754   6773   6867   6868   6869
         6871   6872   6894   6934   6974   7008   7042
SETPRI   1279#  5774#
SETTRA   7335#
SETUP    1337#  5774#  5899
SKIP     1733#  5774#  6217   6220   6223   6256   6274   6276   6278   6352   6392   6394   6397   6508   6954
         6964   6968   6995   7001
SLASH    1517#  5774#
SPACE    5774#
STARS    1485#  5774#  5783   5785   5786   5968   5969   6016   6017   6018   6019   6020   6021   6022   6023
         6024   6025   6060   6063   6081   6108   6139   6166   6200   6238   6261   6287   6308   6333   6369
         6406   6427   6493   6496   6499   6502   6505   6508   6512   6538   6569   6590   6694   6712   6732
         6754   6773   6867   6868   6869   6871   6872   6894   6934   6974   7008   7106   7246   7247   7260
         7261   7263   7264   7265   7266   7335
SWRSU    1453#  5774#  5899#
TRMTRP   7335#
```

```
TYPBIN   2088#   5774#   7106
TYPDEC   2058#   5774#
TYPNAM   1826#   5774#   5908
TYPNUM   2025#   5774#
TYPOCS   1978#   5680#   5774#
TYPOCT   1941#   5774#   7066   7070   7082   7106   7261   7263
TYPTXT   1894#   5774#   5912   5913   6985   6998   7016   7065   7067   7069   7081   7083   7106   7114
ZIOTM    7120#   7145    7174
ZP21     6070#   6081    6108
ZP25     6191#   6200
ZP25A    6228#   6238
ZZ1      5970#   6016    6017   6018   6019   6020   6021   6022   6023   6024   6025   6323#  6333   6493#  6496#
         6499#   6502#   6505#  6508#  6874#  6894   6926#  6934
$$CMRE   5786#
$$CMTM   5786#
$$ESCA   1711#   5774#
$$NEWT   1662#   5774#   5968   5969   6016   6017   6018   6019   6020   6021   6022   6023   6024   6025   6060
         6063    6081    6108   6139   6166   6200   6238   6261   6287   6308   6333   6369   6406   6427   6493
         6496    6499    6502   6505   6508   6512   6538   6569   6590   6694   6712   6732   6754   6773   6867
         6868    6869    6871   6872   6894   6934   6974   7008
$$SET    7335#
$$SETM   5899#
$$SKIP   1746#   5774#   6217   6220   6223   6256   6274   6276   6278   6352   6392   6394   6397   6508   6954
         6964    6968    6995   7001
.EQUAT    189#   5680#   5774
.HEADE     61#   5679#   5728
.INIT    5658#   5774#   7368
.SETTR   5679#
.SETUP   1213#   5679#   5884
.SWRHI    104#   5680#   5730
.SWRLO   5730#
.TRMTR   5679#
.$ACT1   5064#   5682#   5783
.$APTB   5109#   5786#
.$APTH   5370#   5682#   5785
.$APTY   5547#   5682#   7265
.$ASTA   5417#
.$CATC    932#   5680#
.$CMTA   1047#   5680#   5786
.$DB2D   4686#
.$DB2O   4812#
.$DIV    4587#
.$EOP    2214#   5681#   7106
.$ERRO   2700#   5681#   7260
.$ERRT   2896#   5681#   7261
.$MULT   4523#
.$POWE   4229#   5680#   7266
.$RAND   4307#
.$RDDE   3891#
.$RDOC   3797#   5679#
.$READ   3395#   5681#   7263
.$R2AZ   4958#
.$SAVE   3969#
.$SB2D   4771#
.$SB2O   4874#
.$SCOP   2454#   5681#   7262
```

```
.$SIZE   4361#
.$SUPR   4913#
.$TRAP   4073#    5679#    7335
.$TYPB   3287#    5679#    7247
.$TYPD   3209#    5681#
.$TYPE   2985#    5681#    7264
.$TYPO   3112#    5680#    7246
.$40CA    972#


. ABS.  020140     000


 ERRORS DETECTED:  0

 CNKWAA,CNKWAA/CRF/NL:TOC=CNMAC2.SML,CNKWAA.P11
 RUN-TIME: 19 18 1 SECONDS
 RUN-TIME RATIO: 79/38=2.0
 CORE USED:  35K  (70 PAGES)
```