DT07

DT07 UNIBUS SW DIAG
CRDTAC0

AH-F170C-MC
1 OF 1 OCT 1985

digital
MADE IN USA

```
                        .REM_


                        PRODUCT CODE: AC-F169C-MC

                        PRODUCT NAME:   CRDTACO DTO7 UNIBUS SW DIAG

                        PRODUCT DATE:     1 MAY 1985

                        MAINTAINER:     CSS/NSG NETWORKS SYSTEMS GROUP

                        THIS SOFTWARE IS FURNISHED UNDER A LINSCE AND MAY BE
                        USED AND COPIED ONLY IN ACCORDANCE WITH THE TERMS OF
                        SUCH LICENSE AND WITH THE INCLUSION OF THE ABOVE
                        COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES
                        THEROF, MAY NOT BE PROVIDED OR OTHERWISE MADE
                        AVAILABLE TO ANY OTHER PERSON. NO TITLE TO AND
                        OWNERSHIP OF THE SOFTWARE IS HEREBY TRANSFERED.

                        THE INFORMATION IN THE SOFTWARE IS SUBJECT TO CHANGE
                        WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT
                        BY DIGITAL EQUIPMENT CORPORATION.

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

                        THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT
                        COPORATION:
                        DEC     DECUS   DECTAPE
                        MASBUS  PDP     UNIBUS
                        VAX

                        COPYRIGHT (C)    1979,1985 BY DIGITAL EQUIPMENT CORPORATION
                                         MAYNARD MASSACHUSETTS ALL RIGHTS RESERVED
```

TABLE OF CONTENTS

SEQ 0002

SEQ 0003

```
                        1 .0        ABSTRACT:

                THIS DIAGNOSTIC IS DESIGNED TO TEST THE FUNCTIONS
AND CAPABILITIES OF THE DT07 UNIBUS SWITCH.  THE DIAGNOSTIC
IS DIVIDED INTO THREE MAJOR SECTIONS AS FOLLOWS:

MONOPORT; THIS SECTION TESTS A SINGLE PORT WITHOUT INTER-
          ACTION WITH OTHER PORTS.

MULTIPORT; THIS SECTION TESTS THE DT07 PORT WITH THE OTHER
           PORTS IN THE SYSTEM.
MANUAL INTERVENTION; THIS SECTION TESTS CERTAIN FUNCTIONS
                     WHICH REQUIRE OPERATOR ASSISTANCE.

NPR TEST; THIS TEST CHECKS OUT THE NPR LINE ON THE SWITCH BUS.

NOTE::
THIS DIAGNOSTIC WAS ASSEMBLED USING THE FOLLOWING ASSEMBLE
AND LINK COMMANDS ON A VAX/VMS SYSTEM

MCR MAC CRDTAC,CRDTAC/-SP=SYSMAC.MLB/ML,CRDTAC.P11
MCR TKB
CRDTAC.TSK/-HD/-MM,CRDTAC/CR/-SP=CRDTAC.OBJ
PAR=GEN:0:376000
STACK=0
CORSIZ=16

RAY BALDWIN - MODIFIED MULTIPORT SECTION (4/20/85) TO BE COMPATABLE
              WITH VAX DIAG. AND FACILITATE FOR THE 400+ MS
              LOCK OUT OF EXTERNAL INTERRUPTS AFTER BUS CONNECTS
```

```
                                2.0 REQUIREMENTS:


                                2.1 EQUIPMENT

                                    PDP 11-COMPUTER
                                    ASR-33 TELETYPE OR EQUIVALENT
                                    DT07 PORT MODULE
                                    MANUAL CONTROL PANEL (OPTIONAL BUT NEEDED FOR MANUAL INTERVENTION TEST!)

                                    UBE <UNIBUS EXERCISER> (OPTIONAL BUT NEEDED FOR NPR TEST)


                                2.2 PROGRAM STORAGE

                                    PROGRAM REQUIRES 8 K WORDS OF MEMORY


                                2.3 SOFTWARE

                                   ABSOLUTE LOADER OR OTHER INPUT MEDIUM
```

```
                        3.0 SWITCH REGISTER ASSIGNMENTS:

SW08(1)=LOOP ON TEST IN SWR<7:0>
SW08(0)=DO ALL TESTS IN SEQUENCE

SW09(1)=LOOP ON ERROR
SW09(0)=CONTINUE ON ERROR

SW10(1)=BELL ON ERROR
SW10(0)=NO BELL ON ERROR

SW11(1)=INHIBIT ITERATIONS
SW11(0)=ALLOW ITERATIONS

SW13(1)=INHIBIT ERROR TYPEOUTS
SW13(0)=ALLOW ERROR TYPEOUTS

SW14(1)=LOOP ON TEST
SW14(0)=DON'T LOOP ON TEST

SW15(1)=HALT ON ERROR
SW15(0)=CONTINUE ON ERROR

NOTE:
  COMPUTERS WITHOUT A HARDWARE SWITCH REGISTER HAVE A
  SOFTWARE SWITCH REGISTER LOCATED IN MEMORY AT LOCATION
  176 CALLED SWREG.  THIS LOCATION CAN BE CHANGED EITHER
  MANUALLY OR BY TYPING THE CNTL+G KEYES TOGETHER
  THEN RESPONDING TO THE TERMINAL DIALOGUE.
```

```
                        4.0         LOADING PROCEDURE:

STANDARD PROCEDURE FOR LOADING ABSOLUTE BINARY TAPES
SHOULD BE USED.
IF THE PROGRAM RESIDES ON A MASS STORAGE DEVICE SUCH AS
MAGTAPE OR DISK REFER TO THE XXDP USER'S GUIDE.

                        5.0     STARTING PROCEDURE:


                        5.1     STARTING ADDRESSES

                        200.INPUT PARAMETERS AND ALLOWS TEST SELECTION

                        5.2         STARTING SEQUENCES


                        5.2.1       STARTING SEQUENCE UNDER XXDP(CHAIN),APT OR ACT

                                    START AT ADDRESS 200 ONLY. (DEFAULT PARAMETERS)

                        5.2.2       NORMAL STARTING SEQUENCE

                               LOAD PROGRAM ACCORDING TO LOADING PROCEDURE

                               LOAD AND START ADDRESS 200

                               INPUT ACTUAL PARAMETERS AND SELECT TEST

                    5.3         RESTART ADDRESSES

                    RESTART ADDRESSES ARE THE SAME AS STARTING ADDRESSES
```

```
                        6.0     PRELIMINARY OPERATIONS:

                        6 .1     DEVICE ADDRESSES AND VECTOR ADDRESSES

                                 THE STANDARD DEVICE ADDRESS IS 177420.
                                 THE STANDARD DEVICE VECTOR ADDRESS IS 350.
                                 THE STANDARD DEVICE PRIORITY LEVEL IS 7.

                        6 .2     DEVICE/OPTION SETUP

                                 INSURE THAT THE DT07(S) ARE IN PROGRAMMABLE MODE.

                        6 .3     PRELIMINARY PROGRAMS NEEDED
                                 NONE

                        7 .0     OPERATIONAL PROCEDURES:

                                       NORMAL OPERATION WOULD BE TO LOAD THIS PROGRAM,
                                 INSURE THAT ALL DT07 PORTS ARE IN THE PROGRAMMABLE MODE,
                                 RUN THE MONOPORT TEST ON EACH PORT IN THE SYSTEM,THEN
                                 RUN THE MULTIPORT TEST ON ALL PORTS IN THE SYSTEM.
                                 IF YOU HAVE A MANUAL CONTROL PANEL, RUN THE MANUAL INTER-
                                 VENTION TEST.IF YOU HAVE UBE ON THE SWITCH BUS,RUN THE
                                 NPR TEST.
                                       THE PARAMETER INPUT ROUTINE IS TO BE USED WHEN-
                                 EVER YOU WISH TO INPUT OR CHANGE DIAGNOSTIC PARAMETERS
                                 INCLUDING DEVICE ADDRESS,VECTOR ADDRESS,INTERRUPT LEVEL,
                                 NUMBER AND I.D.#S OF OTHER PORTS TO BE TESTED.
                                 THE PARAMETER INPUT ROUTINE WILL BE ENTERED AUTOMATICALY
                                 FROM THE MULTIPORT TEST IF NO PARAMETERS HAVE BEEN ENTERED
                                 PREVIOUSLY.
                                       WHEN USING THE MULTIPORT SECTION, AFTER PARAMETERS
                                 HAVE BEEN ENTERED OR WHEN RESTARTING THE TEST, THE CPU
                                 WITH THE LOWEST NUMBERED DT07 PORT MUST BE STARTED LAST.
```

```
                                8.0     ERRORS:

                                8.1     MONOPORT SECTION ERRORS

                                        ALL ERRORS IN THE MONOPORT SECTION HAVE THE GENERAL FORMAT;
                                        ERROR PC,TEST#,GOOD DATA,BAD DATA,BAD DATA ADDRESS.  CONTROL
                                        OF ERROR PRINTOUT AND LOOPING IS BY THE SWITCH REGISTER OPTIONS.

                                8.2     MULTIPORT SECTION ERRORS

                                        ERRORS IN THE MULTIPORT SECTION PRINT AN ERROR ID.#, A MESSAGE
                                        BRIEFLY DESCRIBING THE ERROR AND THE CSR CONTENTS.  ALL ERRORS
                                        IN THIS SECTION ARE FATAL AND, THEREFORE, THE TEST MUST BE RESTARTED
                                        ON ERROR.  THERE IS NO ERROR LOOPING AND ERROR PRINTOUT IS AUTO-
                                        MATIC.  ERROR ID.#S 1A THRU 5A IDENTIFY ERRORS IN THE DT07 MODE
                                        MULTIPORT TEST AND ERROR ID.#S 6 THRU 16 IDENTIFY ERRORS IN THE
                                        DT03 MODE MULTIPORT TEST.

                                8.3     MANUAL INTERVENTION SECTION ERRORS

                                        ERRORS IN THIS SECTION PRINT AN ERROR ID.#, A BRIEF MESSAGE DES-
                                        CRIBING THE ERROR, AND THE CSR CONTENTS AT ERROR TIME.  THERE IS
                                        NO ERROR LOOPING AND ERROR PRINTOUT IS AUTOMATIC; HOWEVER,
                                        PRESSING CONTINUE, AFTER ERROR, WILL CAUSE THE FAILING SUBTEST TO
                                        REPEAT EXECUTION.

                                8.4     NPR TEST

                                        ERRORS IN THIS TEST PRINT A BRIEF MESSAGE DESCRIBING THE ERROR AND
                                        ERROR PC.

                                8.5     MISCELLANEOUS ERRORS

                                        THERE ARE TWO SUBROUTINES USED TO DETERMINE DEVICE MODE AND DEVICE
                                        TIMING.  ERRORS IN THESE ROUTINES ARE SELF-EXPLANATORY AND ARE
                                        FATAL.  THESE ERRORS USUALLY OCCUR BECAUSE ONE OF THE PORTS
                                        WAS LEFT IN THE MANUAL MODE.
```

```
                        9.0     TEST DESCRIPTIONS:

                        9.1     MONOPORT TESTS

                                TEST1.  TEST POST-RESET BIT PATTERN IN CSR.
                                        THIS TEST CHECKS THE CONTROL STATUS REGISTER CONTENTS
                                        AFTER RESET TO INSURE THAT ANY RESET-CLEARABLE BITS ARE
                                        CLEARED.
                                TEST2.  TEST READ/WRITE BITS IN CSR.
                                        THIS TEST CHECKS THAT BITS 0 AND 6 CAN BE SET AND CLEARED
                                        IN THE CSR.
                                TEST3.  TEST BYTE ACCESS IN CSR.
                                        THIS TEST CHECKS THE BYTE ACCESS CAPABILITY OF THE CSR.
                                TEST4.  CONNECT TEST WITHOUT INTERUPT ENABLE.
                                        THIS TEST VERIFIES THE PROPER OPERATION OF THE CONNECT
                                        PROCESS WITH INTERUPTS DISABLED.
                                TEST5.  RELEASE TEST WITHOUT INTERUPT ENABLE.
                                        THIS TEST VERIFIES THE PROPER OPERATION OF THE RELEASE
                                        PROCESS WITH INTERUPTS DISABLED.
                                TEST6.  CONNECT FAILURE TEST WITHOUT INTERUPT ENABLE.
                                        THIS TEST CHECKS THE PROPER OPERATION OF THE DEVICE
                                        WHEN THE CONNECTION PROCESS CANNOT CONNECT TO THE SHARED
                                        BUS WITHOUT INTERUPTS ENABLED.
                                TEST7.  RELEASE TEST WITHOUT BUS MASTERSHIP OR INTERUPT ENABLE.
                                        THIS TEST CHECKS THAT THE DEVICE WILL RELEASE THE SHARED
                                        BUS WHEN UNIBUS MASTERSHIP CANNOT BE OBTAINED AND
                                        INTERUPTS ARE DISABLED.
                                TEST10. CONNECT TEST
                                        THIS TEST VERIFIES THE CONNECT PROCESS UNDER INTERUPT CONTROL.
                                TEST11. RELEASE T ST
                                        THIS TEST VERIFIES THE RELEASE PROCESS UNDER INTERUPT CONTROL.
                                TEST12. CONNECT FAILURE TEST
                                        THIS TEST VERIFIES THE PROPER OPERATION OF THE CONNECT
                                        PROCESS,UNDER INTERUPT CONTROL,WHEN THE DEVICE CANNOT
                                        OBTAIN BUS MASTERSHIP.
                                TEST13. RELEASE TEST WITHOUT BUS MASTERSHIP
                                        THIS TEST VERIFIES THE THE PROPER OPERATION OF THE RELEASE
                                        PROCESS, UNDER INTERUPT CONTROL, WHEN THE DEVICE CANNOT
                                        OBTAIN BUS MASTERSHIP.
                                TEST14. RESET (RST) TEST
                                        THIS TEST CHECKS THE OPERATION OF THE DEVICE RESET BIT
                                        (RST) IF AN EXTERNAL DEVICE REGISTER HAS BEEN INSERTED
                                        IN THE PARAMETER BLOCK.
                                TEST15. RESET HOLD (HLD) TEST
                                        THIS TEST CHECKS THAT THE HLD BIT WILL HOLD THE  SHARED
                                        BUS DURING AND AFTER A BUS RESET.
                                TEST16. HI-SPEED SWITCHING TEST
                                        THIS TEST CAUSES THE DEVICE TO CONNECT AND DISCONNECT
                                        AT HIGH SPEED 1000 TIMES WHILE CHECKING FOR ERRORS.
                                TEST17. RESET-IN-NEUTRAL TEST
                                        THIS TEST CHECKS FOR RESET-IN-NEUTRAL IF AN EXTERNAL
                                        DEVICE REGISTER IS PRESENT AND PRINTS OUT THE RESULT ON
                                        THE FIRST PASS.
```

9.2 MULTIPORT TEST

THE MULTIPORT TEST IS ACTUALLY TWO SEPARATE TESTS: ONE FOR THE DT07 IN DT03 MODE AND ONE FOR THE DT07 IN DT07 MODE. IN DT07 MODE THE PORT I.D.#S OF ALL THE PORTS TO BE TESTED ARE ARRANGED IN ASCENDING NUMERICAL ORDER THEN A MENU OF FUNCTIONS TO BE PERFORMED IS CREATED. EACH DT07 PORT WILL HAVE A TURN AS MASTER WHILE THE OTHER PORTS ARE SLAVES. THERE ARE TWO MASTER FUNCTIONS AND THREE SLAVE FUNCTIONS; THESE BEING: FUNCTION1 (MASTER) DETECT SELECTED SLAVE REQUEST AND REJECT IT, FUNCTION2 (MASTER) DETECT SELECTED SLAVE REQUEST AND IGNORE IT, FUNCTION3 (SLAVE) SEND REQUEST AND EXPECT REJECTION, FUNCTION4 (SLAVE) SEND REQUEST AND EXPECT CONNECTION,AND FUNCTION5 (SLAVE) OBSERVE REQUEST INDICATORS.
IN DT03 MODE BOTH DT07S HAVE A TURN AS MASTER THEN AS SLAVE.
THERE ARE TWO MASTER FUNCTIONS AND TWO SLAVE FUNCTIONS;
THESE BEING: 1. RECEIVE REQUEST AND RELEASE THE SHARED BUS,
2. SEND REQUEST AND EXPECT REJECTION, 3. SEND REQUEST AND EXPECT CONNECTION, AND 4. DETECT REQUEST AND REJECT IT.

9.3 MANUAL INTERVENTION TEST

THE MANUAL INTERVENTION TEST VERIFIES THE PROPER OPERATION OF CERTAIN DEVICE FEATURES THAT REQUIRE MANUAL ASSISTANCE TO TEST. THIS TEST ALSO REQUIRES THE OPTIONAL MANUAL CONTROL PANEL FOR PROPER OPERATION. THE FEATURES TESTED ARE: MANUAL MODE, PORT POWER OK AND SWITCHED BUS POWER FAIL.

9.4 NPR TEST

THE NPR TEST CHECKS OUT THE NPR LINE ON THE SWITCH BUS.

```
                                .NLIST  MC,MD,CND,ME
                                .LIST   SEQ,LOC,BIN
                                ;.ENABL ABS
000000                          .ASECT

                                        .MCALL  .HEADER,.SETUP, .$TRAP,.$RDOCT,.KT11,.$RAND
                                        .MCALL  TYPNAM,TYPOCS,.$POWER,.$40CAT,.$TYPOCT,.EQUAT,.$CMTAG
                                        .MCALL  .SWRHI,.$EOP,.$ERROR,.$ERRTYP,.$TYPDEC,.$SCOPE
                                        .MCALL  .$READ,.$TYPE,.$ACT11,.$APTHDR,.$APTBLS,.$APTYPE

        167400                  $SWR=167400
        000001                  $TN=1

        177420                  ABASE=177420            ;BASE DT07 BUS ADDRESS EQUATE
        160350                  AVECT1=160350           ;BASE DT07 PRIORITY & VECTOR ADDRESS EQUATE
        001124                  GOOD=$GDDAT
        001126                  BAD=$BDDAT
        000001                  REQ=BIT0
        001122                  BADA=$BDADR
```

SEQ 0012

```
    349         001000                          .=1000
```

```
351                                     .EVEN
     001206                             $MAIL:                          ;;APT MAILBOX
     001206  000000                     $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
     001210  000000                     $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
     001212  000000                     $TESTN: .WORD   ATESTN          ;;TEST NUMBER
     001214  000000                     $PASS:  .WORD   APASS           ;;PASS COUNT
     001216  000000                     $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
     001220  000000                     $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
     001222  000000                     $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
     001224  000000                     $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
     001226                             $ETABLE:                        ;;APT ENVIRONMENT TABLE
     001226     000                     $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
     001227     000                     $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
     001230  000000                     $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
     001232  000000                     $USWR:  .WORD   AUSWR           ;;USER SWITCHES
     001234  000000                     $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
                                        ;*                              BITS 15-11=CPU TYPE
                                        ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                        ;*                                      11/70=06,PDQ=07,Q=10
                                        ;*                              BIT 10=REAL TIME CLOCK
                                        ;*                              BIT  9=FLOATING POINT PROCESSOR
                                        ;*                              BIT  8=MEMORY MANAGEMENT
     001236     000                     $MAMS1: .BYTE   AMAMS1          ;;HIGH ADDRESS,M.S. BYTE
     001237     000                     $MTYP1: .BYTE   AMTYP1          ;;MEM. TYPE,BLK#1
                                        ;*                              MEM.TYPE BYTE   --  (HIGH BYTE)
                                        ;*                                      900 NSEC CORE=001
                                        ;*                                      300 NSEC BIPOLAR=002
                                        ;*                                      500 NSEC MOS=003
     001240  000000                     $MADR1: .WORD   AMADR1          ;;HIGH ADDRESS,BLK#1
                                        ;*                                MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
     001242     000                     $MAMS2: .BYTE   AMAMS2          ;;HIGH ADDRESS,M.S. BYTE
     001243     000                     $MTYP2: .BYTE   AMTYP2          ;;MEM.TYPE,BLK#2
     001244  000000                     $MADR2: .WORD   AMADR2          ;;MEM.LAST ADDRESS,BLK#2
     001246     000                     $MAMS3: .BYTE   AMAMS3          ;;HIGH ADDRESS,M.S.BYTE
     001247     000                     $MTYP3: .BYTE   AMTYP3          ;;MEM.TYPE,BLK#3
     001250  000000                     $MADR3: .WORD   AMADR3          ;;MEM.LAST ADDRESS,BLK#3
     001252     000                     $MAMS4: .BYTE   AMAMS4          ;;HIGH ADDRESS,M.S.BYTE
     001253     000                     $MTYP4: .BYTE   AMTYP4          ;;MEM.TYPE,BLK#4
     001254  000000                     $MADR4: .WORD   AMADR4          ;;MEM.LAST ADDRESS,BLK#4
     001256  160350                     $VECT1: .WORD   AVECT1          ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
     001260  000000                     $VECT2: .WORD   AVECT2          ;;INTERRUPT VECTOR#2BUS PRIORITY#2
     001262  177420                     $BASE:  .WORD   ABASE           ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
     001264  000000                     $DEVM:  .WORD   ADEVM           ;;DEVICE MAP
     001266  000000                     $CDW1:  .WORD   ACDW1           ;;CONTROLLER DESCRIPTION WORD#1
     001270  000000                     $CDW2:  .WORD   ACDW2           ;;CONTROLLER DESCRIPTION WORD#2
     001272  000000                     $DDW0:  .WORD   ADDW0           ;;DEVICE DESCRIPTOR WORD#0
     001274  000000                     $DDW1:  .WORD   ADDW1           ;;DEVICE DESCRIPTOR WORD#1
     001276  000000                     $DDW2:  .WORD   ADDW2           ;;DEVICE DESCRIPTOR WORD#2
     001300  000000                     $DDW3:  .WORD   ADDW3           ;;DEVICE DESCRIPTOR WORD#3
     001302  000000                     $DDW4:  .WORD   ADDW4           ;;DEVICE DESCRIPTOR WORD#4
     001304  000000                     $DDW5:  .WORD   ADDW5           ;;DEVICE DESCRIPTOR WORD#5
     001306  000000                     $DDW6:  .WORD   ADDW6           ;;DEVICE DESCRIPTOR WORD#6
     001310  000000                     $DDW7:  .WORD   ADDW7           ;;DEVICE DESCRIPTOR WORD#7
     001312  000000                     $DDW8:  .WORD   ADDW8           ;;DEVICE DESCRIPTOR WORD#8
     001314  000000                     $DDW9:  .WORD   ADDW9           ;;DEVICE DESCRIPTOR WORD#9
     001316  000000                     $DDW10: .WORD   ADDW10          ;;DEVICE DESCRIPTOR WORD#10
     001320  000000                     $DDW11: .WORD   ADDW11          ;;DEVICE DESCRIPTOR WORD#11
```

SEQ 0014

```
001322  000000                  $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
001324  000000                  $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
001326  000000                  $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
001330  000000                  $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
001332                          $ETEND:
```

```
                                .SBTTL  ERROR POINTER TABLE
                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                ;*      DH              ;;POINTS TO THE DATA HEADER
                                ;*      DT              ;;POINTS TO THE DATA
                                ;*      DF              ;;POINTS TO THE DATA FORMAT
    001332                      $ERRTB:
352
353 001332  033332              ;ERROR#1
354 001334  034563                      EM1             ;POST-RESET CONDITIONS NOT MET IN CSR
355 001336  035134                      DH1
356 001340  000000                      DT1
357                                     0
358 001342  033400              ;ERROR#2
359 001344  034563                      EM2             ;READ/WRITE BIT FAILURE IN CSR
360 001346  035134                      DH1
361 001350  000000                      DT1
362                                     0
363 001352  033437              ;ERROR#3
364 001354  034563                      EM3             ;BYTE ACCESS FAILURE IN CSR
365 001356  035134                      DH1
366 001360  000000                      DT1
367                                     0
368 001362  033473              ;ERROR#4
369 001364  034563                      EM4             ;ERROR IN CONNECT PROCESS
370 001366  035134                      DH1
371 001370  000000                      DT1
372                                     0
373 001372  033525              ;ERROR#5
374 001374  034563                      EM5             ;INTERUPT DETECTED WITH INTERUPT ENABLE CLEARED
375 001376  035134                      DH1
376 001400  000000                      DT1
377                                     0
378 001402  033605              ;ERROR#6
379 001404  034563                      EM6             ;COULD NOT CONNECT
380 001406  035134                      DH1
381 001410  000000                      DT1
382                                     0
383 001412  033630              ;ERROR#7
384 001414  034563                      EM7             ;ERROR IN RELEASE PROCESS
385 001416  035134                      DH1
386 001420  000000                      DT1
387                                     0
388 001422  033662              ;ERROR#10
389 001424  034563                      EM10            ;NO INTERRUPT DETECTED
390 001426  035134                      DH1
391 001430  000000                      DT1
392                                     0
393 001432  033710              ;ERROR#11
394 001434  034563                      EM11            ;INTERRUPT AT WRONG PRIORITY
395 001436  035134                      DH1
396 001440  000000                      DT1
397                                     0
                                ;ERROR#12
```

```
 398 001442  033744                    EM12            ;CONNECT CONDITION FAILURE
 399 001444  034563                    DH1
 400 001446  035134                    DT1
 401 001450  000000                    0
 402                            ;ERROR#13
 403 001452  033777                    EM13            ;ERROR IN CONNECT FAILURE PROCESS
 404 001454  034563                    DH1
 405 001456  035134                    DT1
 406 001460  000000                    0
 407                            ;ERROR#14
 408 001462  034041                    EM14            ;ERROR IN RELEASE TEST W/O BUSS MASTERSHIP
 409 001464  034563                    DH1
 410 001466  035134                    DT1
 411 001470  000000                    0
 412                            ;ERROR#15
 413 001472  034120                    EM15            ;RELEASE CONDITIONS NOT MET
 414 001474  034563                    DH1
 415 001476  035134                    DT1
 416 001500  000000                    0
 417                            ;ERROR#16
 418 001502  034154                    EM16            ;TMO BIT NOT SET
 419 001504  034563                    DH1
 420 001506  035134                    DT1
 421 001510  000000                    0
 422                            ;ERROR#17
 423 001512  034175                    EM17            ;COULDN'T READ/WRITE EXTERNAL DEVICE
 424 001514  034626                    DH2
 425 001516  035150                    DT2
 426 001520  000000                    0
 427                            ;ERROR#20
 428 001522  034242                    EM20            ;RESET HOLD (HLD) DIDN'T DISABLE RESET
 429 001524  034563                    DH1
 430 001526  035134                    DT1
 431 001530  000000                    0
 432                            ;ERROR#21
 433 001532  034311                    EM21            ;RESET (RST) DIDN'T WORK
 434 001534  034677                    DH3
 435 001536  035164                    DT3
 436 001540  000000                    0
 437                            ;ERROR#22
 438 001542  034342                    EM22            ;RESET-IN-NEUTRAL OPERATION HAS CHANGED
 439 001544  034760                    DH4
 440 001546  035150                    DT2
 441 001550  000000                    0
 442                            ;ERROR+23
 443 001552  027252                    EM23            ;ERROR:NPR DATO NOT DONE
 444 001554  035052                    DH5
 445 001556  035176                    DT4
 446 001560  000000                    0
 447                            ;ERROR+24
 448 001562  027302                    EM24            ;ERROR:NPR DID NOT SET RDY
 449 001564  035052                    DH5
 450 001566  035176                    DT4
 451 001570  000000                    0
 452                            ;ERROR+25
 453 001572  027334                    EM25            ;ERROR:UBE DID NOT INTERRUPT WHEN NPR FINISHED
 454 001574  035065                    DH6
```

SEQ 0017

```
    455 001576  035202                    DT5
    456 001600  000000                    0
    457                           ;ERROR+26
    458 001602  027412                    EM26        ;TWO LOC WRITTEN WHEN ONE NPR AND INT ON DONE ENABLE
    459 001604  035052                    DH5
    460 001606  035176                    DT4
    461 001610  000000                    0
    462                           ;ERROR+27
    463 001612  027504                    EM27        ;ERROR:DEVICE DOESN'T EXIST ON SWITCH BUS
    464 001614  035052                    DH5
    465 001616  035176                    DT4
    466 001620  000000                    0
```

```
                                ;
                                ;       DT07 BUS REGISTER ADDRESS POINTERS
                                ;
001622  177420                  CSR:    177420                  ;COMMAND/STATUS REGISTER
                                ;
                                ;       DT07 VECTOR ADDRESS POINTERS
                                ;
001624  000350                  VEC0:   350                     ;NORMAL INTERRUPT VECTOR
001626  000352                  VEC2:   352                     ;
                                ;
                                ;       DT07 DEVICE LEVEL
                                ;
001630  000340                  PRI:    340
                                ;
                                ;       COMMAND STATUS REGISTER BIT ASSIGNMENTS
                                ;
                                ;       COMMON PROGRAM LOCATION(S)
                                ;
001632  000000                  TSTNUM: 0                       ;CONTAINS TEST NUMBER ON ERROR
                                ;;*****************************************************************
                                ;PROGRAM PARAMETER BLOCK
001634  000000                  DT03:   0000            ;DT03/DT07 MODE INDIACATER,0=DT07
001636  000000                  DEVAD:  0000            ;EXTERNAL DEVICE REGISTER ADDRESS
001640  000000                  DEVRW:  0000            ;READ/WRITE BITS IN ABOVE
001642  000000                  DEVRC:  0000            ;MASK OF RESET CLEARABLE BITS IN ABOVE
001644  177777                  PORTNO: -1              ;THIS PORT'S ID#
001646  177777                  EXPRT1: -1              ;EXTERNAL PORT ID (-1=NO PORT)
001650  177777                  EXPRT2: -1              ;       "       "   "
001652  177777                  EXPRT3: -1              ;       "       "   "
001654  177777                  EXPEND: -1              ;TABLE TERMINATOR
001656  000000                  EXPTCT: 0               ;# OF EXTERNAL PORTS
001660  177777                  SEQ:    -1              ;SEQUENCER
001662  177777                          -1
001664  177777                          -1
001666  177777                          -1
001670  177777                          -1
001672  001660                  SEQPTR: SEQ             ;SEQUENCER POINTER
001674  001662                  NEXENT: SEQ+2           ;NEXT ENTRY POINTER
001676                          MENU:   .BLKW   9.      ;MENU STORAGE
001720  177777                          -1              ;MENU TERMINATOR
001722  000000                  CURFUN: 0               ;CURRANT FUNCTION
001724  000000                  CURID:  0               ;CURRANT SLAVE ID
001726  000000                  CYCLE:  0               ;SUBPASS COUNT
001730  000000                  COVID:  0               ;CONVERTED SLAVE ID
001732                          TEMP1:  .BLKW 5         ;TEMPORARY STORAGE
001744  000000                  ONOFF:  0               ;MASTER/SLAVE INDIACATOR
001746  000000                  TIMA:   0
001750  000000                  TIMB:   0
001752  000000                  KONST:  0
001754  000001                  DEVCON: 1               ;RESET-IN-NEUTRAL CONDITION FLAG
001756  177777                  TRIN:   -1              ;TEST RESET IN NEUTRAL FLAG
001760  000000                  USEPAR: 0               ;USING PARIN FLAG 0=NOT USING
```

```
523
524 001762                                  .SBTTL PROGRAM START
                                    START:
                                    .SBTTL  INITIALIZE THE COMMON TAGS
                                    ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
    001762  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
    001766  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
    001770  022706  001140                  CMP     #SWR,R6 ;;DONE?
    001774  001374                          BNE     .-6             ;;LOOP BACK IF NO
    001776  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                    ;;INITIALIZE A FEW VECTORS
    002002  012737  032446  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
    002010  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
    002016  012737  031762  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
    002024  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
    002032  012737  033146  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
    002040  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
    002046  012737  032740  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
    002054  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
    002062  016767  017276  017266          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
    002070  005067  177076                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
    002074  005067  177074                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
    002100  112767  000001  177007          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
    002106  012767  002106  176772          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
    002114  012767  002114  176766          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                    ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                    ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
    002122  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
    002126  012737  002162  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
    002134  012767  177570  176776          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
    002142  012767  177570  176772          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
    002150  022777  177777  176762          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
    002156  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                    ;;AND  THE HARDWARE SWR IS NOT = -1
    002160  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
    002162  012716  002170          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
    002166  000002                          RTI
    002170  012767  000176  176742  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
    002176  012767  000174  176736          MOV     #DISPREG,DISPLAY
    002204  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
    002210  005067  177000                  CLR     $PASS           ;;CLEAR PASS COUNT
    002214  132767  000200  177005          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
    002222  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
    002224  012767  001230  176706          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
    002232                          67$:
525 002232  012737  026222  000004  SETUP1: MOV     #TRAP4,@#ERRVEC         ;POINT TO TRAP ROUTINE
526 002240  005000                          CLR     R0
527 002242  016700  177010                  MOV     $VECT1,R0       ;GET PRIORITY LEVEL
528 002246  042700  000777                  BIC     #777,R0         ;CLEAR UNWANTED BITS
529 002252  000300                          SWAB    R0
530 002254  010067  177350                  MOV     R0,PRI  ;SAVE DEVICE LEVEL
531                                 ;
532                                 ;       NOW SETUP BUS ADDRESS VALUES
533                                 ;
534 002260  016767  176776  177334  SETUP2: MOV     $BASE,CSR       ;STORE  BUS ADDRESS
535                                 ;
536                                 ;       NOW SETUP VECTOR ADDRESSES
537                                 ;
```

INITIALIZE THE COMMON TAGS


```
                                                MOV     #VEC0,R0        ;SETUP VECTOR POINTER
                                                MOV     $VECT1,R1       ;GET BASE VECTOR ADDR
                                                BIC     #177000,R1      ;CLEAR UNWANTED BITS
                                        SETUP3: MOV     R1,(R0)+        ;PUT ADDRS
                                                ADD     #2,R1           ;INCR.TO NEXT LOC.
                                                CMP     #VEC2+2,R0      ;DONE ALL LOCATIONS?
                                                BNE     SETUP3          ;BR,IF NOT DONE
                                                TST     @#42            ;TEST CHAIN MODE UNDER XXDP
                                                BNE     1$              ;BR,IF CHAIN MODE
                                        .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
                                                INC     #-1             ;;FIRST TIME?
                                                BNE     64$             ;;BRANCH IF NO
                                                CMP     #$ENDAD,@#42    ;;ACT-11?
                                                BEQ     64$             ;;BRANCH IF YES
                                                TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
                                                TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
                                                BNE     66$             ;;BRANCH IF YES
                                                CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
                                                BEQ     66$             ;;BRANCH IF YES
                                                CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
                                                BNE     67$             ;;BRANCH IF NO
                                                GTSWR                   ;;GET SOFT-SWR SETTINGS
                                                BR      67$
                                        66$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
                                        67$:
                                                BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>/MAINDEC-11-CRDTAC  DT07 TEST/<CRLF>
                                        64$:
                                        1$:     TST     USEPAR          ;PARIN IN USE?
                                                BNE     RESTRT          ;BR=YES; SKIP OVER DEVICE ADDRESS DEPENDENT CODE
                                                MOV     #OOPS,@VEC0     ;LOAD UNEXPECTED INTERUPT RETURN
                                                MOV     #340,@VEC2      ;SET PRIORITY 7 ON INTERUPT
                                                CLR     @CSR            ;DETERMINE STATUS OF PWR OK BIT
                                                RESET                   ;RESET THE BUSS
                                                NOP
                                                NOP
                                                NOP
                                                CLR     KONST
                                                MOV     @CSR,KONST      ;SAVE CSR
                                                BIC     #173777,KONST   ;MASK EVERYTHING EXCEPT PWR OK
                                        RESTRT: MOV     #STACK,SP       ;RESET STACK POINTER
                                                MOV     #2,@#512        ;PUT RTI IN UBE VECTORS
                                                MOV     #2,@#516        ;DITTO
                                                MOV     #2,@#522        ;DITTO
                                                MOV     #2,@#526        ;DITTO
                                                JMP     @ROUTE          ;GO DO IT
                                        ROUT1:  MOV     #MONPRT,ROUTE   ;SET UP FOR MONOPORT
                                                JMP     START
                                        ROUT2:  MOV     #MULPRT,ROUTE   ;SET UP FOR MULTIPORT
                                                JMP     START
                                        ROUT3:  MOV     #1,USEPAR       ;SET FLAG
                                                MOV     #PARIN,ROUTE    ;SET RETURN
                                                CLR     PARFLG  ;CLEAR PARAMETER FLAG
                                                JMP     START
                                        ROUT4:  MOV     #1$,ROUTE       ;SET UP RETURN
```

```
    575 002644  000167  177112                      JMP     START
    576 002650  012767  015056  000014  1$:         MOV     #MANIN,ROUTE    ;SET UP FOR MANUAL INTERVENTION TEST
    577 002656  004767  020434                      JSR     PC,DTIME        ;GO GET DEVICE TIMING
    578 002662  004767  020066                      JSR     PC,DMODE        ;GO GET DEVICE MODE
    579 002666  000167  012164                      JMP     MANIN
    580 002672  002676                      ROUTE:  MONPRT                  ;MAIN TEST DISPATCHER
    581 002674  000000                      TEMP2:  0
    582
    583 002676                               MONPRT:
    584 002676  004767  023474                      JSR     PC,CNTLC        ;IS IT CONTROL C ?
    585 002702  005767  023276                      TST     PARFLG
    586 002706  100402                              BMI     1$
    587 002710  004767  021074                      JSR     PC,GETADR
    588 002714  004767  020034              1$:     JSR     PC,DMODE
    589 002720  004767  020372                      JSR     PC,DTIME
    590 002724  005737  000042                      TST     @#42            ;XXDP CHAINMODE OR APT/ACT?
    591 002730  001053                              BNE     TST1            ;SKIP PRINTOUT IF ANYONE OF THE ABOVE
    592 002732  005767  176256                      TST     $PASS           ;0 PASSES?
    593 002736  001046                              BNE     3$              ;BR=NO
    594 002740  005767  176670                      TST     DT03            ;WHAT MODE IS DEVICE?
    595 002744  001422                              BEQ     2$              ;BR =DT07 MODE
    596 002746  104401  002754                      TYPE    ,65$            ;;TYPE ASCIZ STRING
        002752  000416                              BR      64$             ;;GET OVER THE ASCIZ
                                             ;;65$:  .ASCIZ  <200>/THIS DT07 IS IN DT03 MODE./
        003010                               64$:
    597 003010  000421                              BR      3$              ;GO START TESTS
    598 003012                               2$:
        003012  104401  003020                      TYPE    ,67$            ;;TYPE ASCIZ STRING
        003016  000413                              BR      66$             ;;GET OVER THE ASCIZ
                                             ;;67$:  .ASCIZ  <200>/THIS DT07 IS PORT# /
        003046                               66$:
    599 003046  016746  176572                      MOV     PORTNO,-(SP)    ;;SAVE PORTNO FOR TYPEOUT
        003052  104402                              TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    600 003054  004767  016460              3$:     JSR     PC,WAIT10       ;WAIT ONE SECOND
    601
```

```
603                                     .SBTTL
604
605                                     ;;*********************************************************************
                                        ;*TEST 1        TEST POST-RESET BIT PATTERN IN CSR.IDX1
                                        ;;*********************************************************************
    003060  000004                      TST1:   SCOPE
    003062  012767  000062  176102              MOV     #50.,$TIMES     ;;DO 50. ITERATIONS
606 003070  004767  023302                      JSR     PC,CNTLC        ;IS IT CONTROL C?
607 003074  012767  000340  174674              MOV     #340,PSW        ;RISE PRIORITY
608 003102  012777  000101  176512              MOV     #101,@CSR       ;SET SOME BITS
609 003110  016767  176636  176006              MOV     KONST,GOOD      ;STORE GOOD DATA
610 003116  000005                              RESET                   ;INIT. BUS
611 003120  005200                      1$:     INC     R0              ;DELAY AWHILE
612 003122  001376                              BNE     1$
613 003124  017767  176472  175774              MOV     @CSR,BAD        ;STORE CONTENTS OF CSR AND
614 003132  016767  176464  175762              MOV     CSR,BADA        ;ADDRESS
615 003140  026767  175760  175760              CMP     GOOD,BAD        ;COMPARE
616 003146  001401                              BEQ     2$              ;BR IF OK
617 003150  104001                              ERROR+1                 ;POST-RESET CONDITIONS NOT MET IN CSR
618 003152                              2$:                             ;GO TO NEXT TEST
```

```
620                                     ;;*************************************************************************
                                        ;*TEST 2         TEST READ/WRITE BITS IN CSR.IDX2
                                        ;;*************************************************************************
     003152  000004                     TST2:   SCOPE
     003154  012767  000062  176010             MOV     #50.,$TIMES     ;;DO 50. ITERATIONS
621  003162  004767  023210                    JSR     PC,CNTLC        ;IS IT CONTROL C?
622
623  003166  012767  000340  174602             MOV     #340,PSW        ;LOCKOUT INTERUPTS
624  003174  005077  176422                    CLR     @CSR            ;ZERO CSR
625  003200  016767  176546  175716  1$:       MOV     KONST,GOOD      ;STORE GOOD DATA
626  003206  062767  000001  175710             ADD     #1,GOOD
627  003214  012767  003200  175666             MOV     #1$,$LPERR      ;SET LOOP-ON-ERROR ADDRESS
628  003222  016777  175676  176372             MOV     GOOD,@CSR       ;SET BIT PATTERN
629  003230  017767  176366  175670             MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
630  003236  042767  120600  175662             BIC     #120600,BAD     ;MASK THE CONNECT PROCESS INDIACATORS
631  003244  042767  000074  175654             BIC     #74,BAD         ;MASK RQO-3
632  003252  026767  175646  175646             CMP     GOOD,BAD        ;COMPARE
633  003260  001401                             BEQ     3$              ;BR IF OK
634  003262  104002                             ERROR+2                 ;READ/WRITE BIT FAILURE IN CSR
635  003264  016767  176462  175632  3$:       MOV     KONST,GOOD      ;STORE GOOD DATA
636  003272  062767  000100  175624             ADD     #100,GOOD
637  003300  012767  003264  175602             MOV     #3$,$LPERR      ;SET LOOP-ON-ERROR ADDRESS
638  003306  016777  175612  176306             MOV     GOOD,@CSR       ;SET BIT PATTERN
639  003314  017767  176302  175604             MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
640  003322  042767  120600  175576             BIC     #120600,BAD     ;MASK THE CONNECT PROCESS INDIACATORS
641  003330  026767  175570  175570             CMP     GOOD,BAD        ;COMPARE
642  003336  001401                             BEQ     4$              ;BR IF OK
643  003340  104002                             ERROR+2                 ;READ/WRITE BIT FAILURE IN CSR
644  003342  016767  176404  175554  4$:       MOV     KONST,GOOD      ;STORE GOOD DATA
645  003350  012767  003342  175532             MOV     #4$,$LPERR      ;SET LOOP-ON-ERROR ADDRESS
646  003356  016777  175542  176236             MOV     GOOD,@CSR       ;SET BIT PATTERN
647  003364  017767  176232  175534             MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
648  003372  026767  175526  175526             CMP     GOOD,BAD        ;COMPARE
649  003400  001401                             BEQ     5$              ;BR IF OK
650  003402  104002                             ERROR+2                 ;READ/WRITE BIT FAILURE IN CSR
651  003404                         5$:                                 ;GO TO NEXT TEST
```

```
                                        ;;*****************************************************************
                                        ;*TEST 3         TEST OF BYTE ACCESS IN CSR.IDX3
                                        ;;*****************************************************************
003404  000004                  TST3:   SCOPE
003406  012767  000012  175556          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
003414  004767  022756                  JSR     PC,CNTLC        ;IS IT CONTRL C?

003420  005077  176176                  CLR     @CSR            ;CLEAR CONTENTS OF CSR
003424  016700  176172                  MOV     CSR,R0          ;STORE ADDRESS OF CSR
003430  112710  000376                  MOVB    #376,@R0        ;TRY TO LOAD LOW ORDER BYTE
003434  016767  176312  175462          MOV     KONST,GOOD      ;STORE GOOD DATA
003442  062767  000100  175454          ADD     #100,GOOD
003450  011067  175452                  MOV     @R0,BAD         ;STORE BAD DATA
003454  026767  175444  175444          CMP     GOOD,BAD        ;COMPARE
003462  001401                          BEQ     2$              ;BR IF OK
003464  104003                          ERROR+3                 ;LOW BYTE ACCESS FAILURE IN CSR
003466  005077  176130          2$:     CLR     @CSR            ;CLEAR CSR
003472  016767  176254  175424          MOV     KONST,GOOD      ;STORE GOOD DATA (PWR OK)
003500  112760  000377  000001          MOVB    #377,1(R0)      ;TRY LOADING HI ORDER BYTE OF CSR
003506  011067  175414                  MOV     @R0,BAD         ;STORE CONTENTS OF CSR
003512  026767  175406  175406          CMP     GOOD,BAD        ;COMPARE
003520  001401                          BEQ     3$              ;BR IF OK
003522  104003                          ERROR+3                 ;HI-BYTE ACCESS FAILURE IN CSR
003524                          3$:                             ;GO TO NEXT TEST
```

SEQ 0025

```
                                        ;;*********************************************************************
                                        ;*TEST 4         CONNECT TEST WITHOUT INTERUPT ENABLE.IDX4
                                        ;;*********************************************************************
003524  000004                  TST4:   SCOPE
003526  012767  000012  175436          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
003534  004767  022636                  JSR     PC,CNTLC        ;IS IT CONTROL  C?

003540  005000                  1$:     CLR     R0              ;SET DELAY TIMER
003542  012767  000000  174226          MOV     #0,PSW          ;LOWER PRIORITY
003550  005077  176046                  CLR     @CSR            ;CLEAR CSR
003554  012777  003736  176042          MOV     #8$,@VEC0       ;SET RETURN
003562  012767  000340  174206          MOV     #340,PSW        ;RISE PRIORITY
003570  016767  176156  175326          MOV     KONST,GOOD      ;STORE GOOD DATA (SWB ACT,PWR OK,REQ)
003576  062767  020001  175320          ADD     #20001,GOOD
003604  005767  176024                  TST     DT03            ;DT03 MODE?
003610  001003                          BNE     11$             ;BR IF DT03
003612  062767  000400  175304          ADD     #400,GOOD       ;SET BRQ IF DT07
003620  012777  000001  175774  11$:    MOV     #REQ,@CSR       ;SET REQ (BIT0)
003626  017767  175770  175272  2$:     MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
003634  042767  000074  175264          BIC     #74,BAD         ;MASK RQ0-3
003642  026767  175256  175256          CMP     GOOD,BAD        ;COMPARE
003650  001403                          BEQ     5$              ;BR IF OK
003652  005200                          INC     R0              ;DELAY
003654  001364                          BNE     2$              ;BR IF NOT DONE
003656  104004                          ERROR+4         ;SWB ACT (BIT13),BRQ OR REQ NOT SET
003660  012767  000000  174110  5$:     MOV     #0,PSW          ;LOWER PRIORITY
003666  016767  176060  175230          MOV     KONST,GOOD      ;STORE GOOD DATA (SWB ACT,PWR OK,CON,REQ)
003674  062767  020201  175222          ADD     #20201,GOOD
003702  005000                          CLR     R0              ;RESET DELAY
003704  017767  175712  175214  6$:     MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
003712  026767  175206  175206          CMP     GOOD,BAD        ;COMPARE
003720  001403                          BEQ     7$              ;BR IF OK
003722  005200                          INC     R0              ;DELAY
003724  001367                          BNE     6$              ;BR IF NOT DONE
003726  104004                          ERROR+4                 ;CON(BIT7) NOT SET
003730  005077  175666          7$:     CLR     @CSR            ;CLEAR CSR AND DISCONNECT
003734  000406                          BR      9$              ;EXIT
003736                          8$:
003736  012600                  MOV     (SP)+,R0                ;;POP STACK INTO R0
003740  012600                  MOV     (SP)+,R0                ;;POP STACK INTO R0
003742  017767  175654  175156  MOV     @CSR,BAD                ;STORE CONTENTS OF CSR
003750  104005                  ERROR+5                         ;INTERUPT DETECTED WITH INTERUPT ENABLE CLEARED
003752  012777  022600  175644  9$:     MOV     #OOPS,@VEC0     ;RESET RETURN
                                                                ;GO TO NEXT TEST
```

```
                                        ;;*************************************************************************
                                        ;*TEST 5         RELEASE TEST WITHOUT INTERUPT ENABLE.IDX5
                                        ;;*************************************************************************
003760  000004                    TST5:   SCOPE
003762  012767  000012  175202            MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
003770  004767  022402                    JSR     PC,CNTLC        ;IS IT CONTROL C?

003774  005000                    1$:     CLR     R0              ;SET DELAY TIMER
003776  012777  000001  175616            MOV     #1,@CSR         ;CONNECT
004004  016767  175742  175112            MOV     KONST,GOOD      ;STORE GOOD DATA (SWB ACT,PWR OK,CON,REQ)
004012  062767  020201  175104            ADD     #20201,GOOD
004020  012767  000000  173750            MOV     #0,PSW          ;DROP PRIORITY
004026  017767  175570  175072    2$:     MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
004034  026767  175064  175064            CMP     GOOD,BAD        ;COMPARE
004042  001404                            BEQ     3$              ;BR IF CONNECTED
004044  005200                            INC     R0              ;DELAY
004046  001367                            BNE     2$              ;BR IF NOT DONE
004050  104006                            ERROR+6                 ;COULD NOT CONNECT
004052  000461                            BR      7$              ;EXIT TEST
004054  012777  004220  175542    3$:     MOV     #8$,@VEC0       ;SET RETURN
004062  012767  000340  173706            MOV     #340,PSW        ;RISE PRIORITY
004070  005000                            CLR     R0              ;SET DELAY TIMER
004072  042777  000001  175522            BIC     #REQ,@CSR       ;CLEAR REQUEST
004100  016767  175646  175016            MOV     KONST,GOOD      ;STORE GOOD DATA (SWB ACT,PWR OK,CON)
004106  062767  020200  175010            ADD     #20200,GOOD
004114  005767  175514                    TST     DT03            ;DT03 MODE?
004120  001003                            BNE     4$              ;BR IF YES
004122  062767  000400  174774            ADD     #400,GOOD       ;ADD BRQ IF DT07
004130  017767  175466  174770    4$:     MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
004136  026767  174762  174762            CMP     GOOD,BAD        ;COMPARE
004144  001403                            BEQ     5$              ;BR IF OK
004146  005200                            INC     R0              ;DELAY
004150  001367                            BNE     4$              ;BR IF NOT DONE
004152  104007                            ERROR+7                 ;BRQ NOT SET OR REQ NOT CLEARED
004154  012767  000000  173614    5$:     MOV     #0,PSW          ;DROP PRIORITY
004162  016767  175564  174734            MOV     KONST,GOOD      ;STORE GOOD DATA (PWR OK)
004170  005000                            CLR     R0              ;RESET DELAY
004172  017767  175424  174726    6$:     MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
004200  026767  174720  174720            CMP     GOOD,BAD        ;COMPARE
004206  001403                            BEQ     7$              ;BR IF OK
004210  005200                            INC     R0              ;DELAY
004212  001367                            BNE     6$              ;BR IF NOT DONE
004214  104007                            ERROR+7                 ;CON OR SWB ACT NOT CLEARED
004216  000406                    7$:     BR      9$              ;EXIT
004220                            8$:
004220  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
004222  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
004224  017767  175372  174674            MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
004232  104005                            ERROR+5                 ;INTERUPPT DETECTED WITH INTERUPT ENABLE CLRD
004234  012777  022600  175362    9$:     MOV     #OOPS,@VEC0     ;RESET RETURN
                                                                  ;GO TO NEXT TEST
```

```
                                        ;;*****************************************************************
                                        ;*TEST 6        CONNECT FAILURE TEST WITHOUT INTERUPT ENABLE.IDX6
                                        ;;*****************************************************************
004242  000004                  TST6:   SCOPE
004244  012767  000012  174720          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
763 004252  004767  022120              JSR     PC,CNTLC        ;IS IT CONTROL C?
764
765 004256  005000              1$:     CLR     R0              ;SET DELAY TIMER
766 004260  012767  000000  173510      MOV     #0,PSW          ;LOWER PRIORITY
767 004266  005077  175330              CLR     @CSR            ;CLEAR CSR
768 004272  012777  004524  175324      MOV     #8$,@VEC0       ;SET RETURN
769 004300  012767  000340  173470      MOV     #340,PSW        ;RISE PRIORITY
770 004306  016767  175440  174610      MOV     KONST,GOOD      ;STORE GOOD DATA (SWB ACT,PWR OK,REQ)
771 004314  062767  020001  174602      ADD     #20001,GOOD
772 004322  005767  175306              TST     DT03            ;DT03 MODE?
773 004326  001003                      BNE     11$             ;BR IF DT03
774 004330  062767  000400  174566      ADD     #400,GOOD       ;ADD BRQ IF DT07
775 004336  012777  000001  175256  11$:MOV     #REQ,@CSR       ;SET REQ (BIT0)
776 004344  017767  175252  174554  2$: MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
777 004352  042767  000074  174546      BIC     #74,BAD         ;MASK RQ0-3
778 004360  026767  174540  174540      CMP     GOOD,BAD        ;COMPARE
779 004366  001403                      BEQ     3$              ;BR IF OK
780 004370  005200                      INC     R0              ;DELAY
781 004372  001364                      BNE     2$              ;BR IF NOT DONE
782 004374  104013                      ERROR+13                        ;SWB ACT (BIT13) OR REQ NOT SET
783 004376  005767  175232          3$: TST     DT03            ;DT03 MODE?
784 004402  001024                      BNE     5$              ;BR IF YES
785 004404  016767  175342  174512      MOV     KONST,GOOD      ;STORE GOOD DATA (SWB ACT,PWR OK,BRQ,REQ)
786 004412  062767  020401  174504      ADD     #20401,GOOD
787 004420  005000                      CLR     R0              ;RESET DELAY
788 004422  017767  175174  174476  4$: MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
789 004430  042767  000074  174470      BIC     #74,BAD         ;MASK RQ0-3
790 004436  026767  174462  174462      CMP     GOOD,BAD        ;COMPARE
791 004444  001403                      BEQ     5$              ;BR IF OK
792 004446  005200                      INC     R0              ;DELAY
793 004450  001364                      BNE     4$              ;BR IF NOT DONE
794 004452  104013                      ERROR+13                        ;BRQ(BIT8) DID NOT SET
795 004454  016767  175272  174442  5$: MOV     KONST,GOOD      ;STORE GOOD DATA (TMO,PWR OK)
796 004462  062767  100000  174434      ADD     #100000,GOOD
797 004470  005000                      CLR     R0              ;RESET DELAY
798 004472  017767  175124  174426  6$: MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
799 004500  026767  174420  174420      CMP     GOOD,BAD        ;COMPARE
800 004506  001403                      BEQ     7$              ;BR IF OK
801 004510  005200                      INC     R0              ;DELAY
802 004512  001367                      BNE     6$              ;BR IF NOT DONE
803 004514  104013                      ERROR+13                        ;TMO(BIT15) NOT SET
804 004516  005077  175100          7$: CLR     @CSR            ;CLEAR CSR AND DISCONNECT
805 004522  000407                      BR      9$              ;EXIT
806 004524                          8$:
004524  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
004526  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
807 004530  017767  175066  174370      MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
808 004536  104005                      ERROR+5                 ;INTERUPT DETECTED WITH INTERUPT ENABLE CLEARED
809 004540  000766                      BR      7$              ;GO CLEAR CSR
810 004542  012777  022600  175054  9$: MOV     #OOPS,@VEC0     ;RESET RETURN
811                                                             ;GO TO NEXT TEST
```

```
                                        ;;*************************************************************************
                                        ;*TEST 7        RELEASE TEST WITHOUT BUS MASTERSHIP OR INTERUPT ENABLE.IDX7
                                        ;;*************************************************************************
004550  000004                        TST7:   SCOPE
004552  012767  000012  174412                MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
004560  004767  021612                        JSR     PC,CNTLC        ;IS IT CONTROL C?

004564  005000                        1$:     CLR     R0              ;SET DELAY TIMER
004566  012777  000001  175026                MOV     #1,@CSR         ;CONNECT
004574  016767  175152  174322                MOV     KONST,GOOD      ;STORE GOOD DATA (SWB ACT,PWR OK,CON,REQ)
004602  062767  020201  174314                ADD     #20201,GOOD
004610  012767  000000  173160                MOV     #0,PSW          ;DROP PRIORITY
004616  017767  175000  174302  2$:     MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
004624  026767  174274  174274                CMP     GOOD,BAD        ;COMPARE
004632  001404                                BEQ     3$              ;BR IF CONNECTED
004634  005200                                INC     R0              ;DELAY
004636  001367                                BNE     2$              ;BR IF NOT DONE
004640  104006                                ERROR+6                 ;COULD NOT CONNECT
004642  000461                                BR      7$              ;EXIT TEST
004644  012777  005010  174752  3$:     MOV     #8$,@VEC0       ;SET RETURN
004652  012767  000340  173116                MOV     #340,PSW        ;RISE PRIORITY
004660  005000                                CLR     R0              ;SET DELAY TIMER
004662  042777  000001  174732                BIC     #REQ,@CSR       ;CLEAR REQUEST
004670  016767  175056  174226                MOV     KONST,GOOD      ;STORE GOOD DATA (SWB ACT,PWR OK,CON)
004676  062767  020200  174220                ADD     #20200,GOOD
004704  005767  174724                        TST     DT03            ;DT03 MODE?
004710  001003                                BNE     4$              ;BR IF YES
004712  062767  000400  174204                ADD     #400,GOOD       ;ADD BRQ IF DT07
004720  017767  174676  174200  4$:     MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
004726  026767  174172  174172                CMP     GOOD,BAD        ;COMPARE
004734  001403                                BEQ     5$              ;BR IF OK
004736  005200                                INC     R0              ;DELAY
004740  001367                                BNE     4$              ;BR IF NOT DONE
004742  104014                                ERROR+14
                                                        ;BRQ NOT SET OR REQ NOT CLEARED
004744  016767  175002  174152  5$:     MOV     KONST,GOOD      ;STORE GOOD DATA (TMO,PWR OK)
004752  062767  100000  174144                ADD     #100000,GOOD
004760  005000                                CLR     R0              ;RESET DELAY
004762  017767  174634  174136  6$:     MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
004770  026767  174130  174130                CMP     GOOD,BAD        ;COMPARE
004776  001403                                BEQ     7$              ;BR IF OK
005000  005200                                INC     R0              ;DELAY
005002  001367                                BNE     6$              ;BR IF NOT DONE
005004  104014                                ERROR+14
                                                        ;TMO NOT SET
005006  000406                        7$:     BR      9$              ;EXIT
005010                                  8$:
005010  012600                                MOV     (SP)+,R0        ;;POP STACK INTO R0
005012  012600                                MOV     (SP)+,R0        ;;POP STACK INTO R0
005014  017767  174602  174104                MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
005022  104005                                ERROR+5                 ;INTERUPPT DETECTED WITH INTERUPT ENABLE CLRD
005024  012777  022600  174572  9$:     MOV     #OOPS,@VEC0     ;RESET RETURN
                                                                        ;GO TO NEXT TEST
```

```
 360
 861                                    ;;*************************************************************************
                                        ;*TEST 10       CONNECT TEST.IDX8
                                        ;;*************************************************************************
     005032  000004                     TST10:  SCOPE
     005034  012767  000012  174130             MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
 862 005042  004767  021330                     JSR     PC,CNTLC        ;IS IT CONTROL C?
 863
 864 005046  005077  174550                     CLR     @CSR            ;CLEAR CSR
 865 005052  012767  000340  172716             MOV     #340,PSW        ;RISE PRIORITY
 866 005060  016767  174544  174036             MOV     PRI,GOOD        ;STORE PRIORITY LEVEL
 867 005066  012767  000340  174032             MOV     #340,BAD        ;STORE ACTUAL PRIORITY INDIACATOR
 868 005074  012777  005150  174522             MOV     #2$,@VEC0       ;SET RETURN
 869 005102  012777  000101  174512             MOV     #101,@CSR       ;SET I.E. AND REQ
 870 005110  032777  020000  174504             BIT     #20000,@CSR     ;WAIT FOR SWB ACT TO APPEAR
 871 005116  001774                             BEQ     .-6             ;BR IF NOT THERE
 872 005120  162767  000040  172650  1$:        SUB     #40,PSW         ;LOWER PRIORITY UNTIL INTERUPTED
 873 005126  000240                             NOP
 874 005130  000240                             NOP
 875 005132  000240                             NOP
 876 005134  162767  000040  173764             SUB     #40,BAD         ;  "      "     INDIACATOR
 877 005142  001366                             BNE     1$              ;BR UNTIL DONE
 878 005144  104010                             ERROR+10                ;NO INTERUPT DETECTED
 879 005146  000425                             BR      4$              ;EXIT
 880 005150                         2$:
     005150  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
     005152  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
 881 005154  026767  173744  173744             CMP     GOOD,BAD        ;CORRECT BREAK LEVEL?
 882 005162  001401                             BEQ     3$              ;BR IF OK
 883 005164  104011                             ERROR+11                ;INTERUPT AT WRONG PRIORITY
 884 005166  017767  174430  173732  3$:        MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
 885 005174  016767  174552  173722             MOV     KONST,GOOD      ;STORE GOOD DATA (SWB ACT,PWR OK,CON,IE,REQ)
 886 005202  062767  020301  173714             ADD     #20301,GOOD
 887 005210  026767  173710  173710             CMP     GOOD,BAD        ;COMPARE
 888 005216  001401                             BEQ     4$              ;BR IF OK
 889 005220  104012                             ERROR+12                ;CONNECT CONDITION FAILURE
 890 005222  005077  174374          4$:        CLR     @CSR            ;CLEAR CSR
 891 005226  005067  172544                     CLR     PSW             ;LOWER PRIORITY
 892 005232  012777  022600  174364  5$:        MOV     #OOPS,@VEC0     ;RESET RETURN
 893                                            ;GO TO NEXT TEST
```

```
 895
 896                                    ;;*****************************************************************
                                        ;*TEST 11        RELEASE TEST .IDX9
                                        ;;*****************************************************************
     005240  000004                     TST11:  SCOPE
     005242  012767  000012  173722             MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
 897 005250  004767  021122                     JSR     PC,CNTLC        ;IS IT CONTROL C?
 898
 899 005254  005000                     1$:     CLR     R0              ;SET DELAY TIMER
 900 005256  012777  000001  174336             MOV     #1,@CSR         ;CONNECT
 901 005264  016767  174462  173632             MOV     KONST,GOOD
 902 005272  062767  020201  173624             ADD     #20201,GOOD     ;STORE GOOD DATA (SWB ACT,PWR OK,CON,REQ)
 903 005300  012767  000000  172470             MOV     #0,PSW          ;DROP PRIORITY
 904 005306  017767  174310  173612  2$:        MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
 905 005314  026767  173604  173604             CMP     GOOD,BAD        ;COMPARE
 906 005322  001404                           BEQ     3$              ;BR IF CONNECTED
 907 005324  005200                           INC     R0              ;DELAY
 908 005326  001367                           BNE     2$              ;BR IF NOT DONE
 909 005330  104006                           ERROR+6                 ;COULD NOT CONNECT
 910 005332  000422                           BR      7$              ;EXIT TEST
 911 005334  005767  174274             3$:     TST     DT03            ;DT03 MODE?
 912 005340  001420                           BEQ     8$              ;BR=NO
 913 005342  005000                           CLR     R0              ;SET DELAY TIMER
 914 005344  042777  000001  174250           BIC     #REQ,@CSR       ;CLEAR REQUEST
 915 005352  016767  174374  173544  5$:        MOV     KONST,GOOD      ;STORE GOOD DATA (PWR OK)
 916 005360  017767  174236  173540  6$:        MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
 917 005366  026767  173532  173532             CMP     GOOD,BAD        ;COMPARE
 918 005374  001401                           BEQ     7$              ;BR IF OK
 919 005376  104015                           ERROR+15                ;RELEASE CONDITIONS NOT MET
 920 005400  000446                     7$:     BR      10$             ;GO TO NEXT TEST
 921 005402  012777  005456  174214  8$:        MOV     #9$,@VEC0       ;SET RETURN
 922 005410  052777  000100  174204             BIS     #100,@CSR       ;SET I.E.
 923 005416  042777  000001  174176             BIC     #1,@CSR         ;CLR REQ.
 924 005424  004767  014136                     JSR     PC,WAIT05       ;GO WAIT
 925 005430  016767  174316  173466             MOV     KONST,GOOD      ;LOAD GOOD
 926 005436  062767  000100  173460             ADD     #100,GOOD
 927 005444  017767  174152  173454             MOV     @CSR,BAD
 928 005452  104010                           ERROR+10                ;NO INTERUPT
 929 005454  000420                           BR      10$             ;EXIT
 930 005456  012706  001100             9$:     MOV     #STACK,SP       ;RESET STACK
 931 005462  016767  174264  173434             MOV     KONST,GOOD      ;LOAD GOOD
 932 005470  062767  000100  173426             ADD     #100,GOOD
 933 005476  017767  174120  173422             MOV     @CSR,BAD
 934 005504  026767  173414  173414             CMP     GOOD,BAD
 935 005512  001401                           BEQ     10$
 936 005514  104015                           ERROR+15
 937 005516  005077  174100             10$:    CLR     @CSR            ;CLEAR AND DISCONNECT
 938 005522  012777  022600  174074             MOV     #OOPS,@VEC0     ;RESET RETURN
```

```
                                        ;;*************************************************************************
                                        ;*TEST 12       CONNECT FAILURE TEST.IDX10
                                        ;;*************************************************************************
    005530  000004                      TST12:  SCOPE
    005532  012767  000012  173432              MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
942 005540  004767  020632                      JSR     PC,CNTLC        ;IS IT CONTROL C?
943
944 005544  005077  174052                      CLR     @CSR            ;CLEAR CSR
945 005550  012767  000340  172220              MOV     #340,PSW        ;RISE PRIORITY
946 005556  005000                              CLR     R0              ;SET DELAY TIMER
947 005560  012777  005672  174036              MOV     #2$,@VEC0       ;SET RETURN
948 005566  016767  174160  173330              MOV     KONST,GOOD
949 005574  062767  100100  173322              ADD     #100100,GOOD    ;STORE GOOD DATA (TM0,PWR OK,IE)
950 005602  005767  174026                      TST     DT03            ;DT03 MODE?
951 005606  001003                              BNE     6$              ;BR IF DT03
952 005610  062767  000400  173306              ADD     #400,GOOD       ;ADD BRQ IF DT07
953 005616  012777  000101  173776  6$:         MOV     #101,@CSR       ;SET I.E. AND REQ
954 005624  017767  173772  173274  1$:         MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
955 005632  026767  173266  173266              CMP     GOOD,BAD        ;COMPARE
956 005640  001404                              BEQ     11$             ;BR IF OK
957 005642  005200                              INC     R0              ;DELAY
958 005644  001367                              BNE     1$              ;BR IF NOT DONE
959 005646  104016                              ERROR+16                ;TM0 NOT SET
960 005650  000414                              BR      5$              ;EXIT
961 005652  005000                      11$:    CLR     R0              ;RESET DELAY
962 005654  012767  000000  172114              MOV     #0,PSW          ;DROP PRIORITY
963 005662  005200                      22$:    INC     R0              ;DELAY
964 005664  001376                              BNE     22$             ;BR IF NOT DONE
965 005666  104010                              ERROR+10                ;NO INTERUPT DETECTED
966 005670  000404                              BR      5$              ;EXIT
967 005672                          2$:
    005672  012600                              MOV     (SP)+,R0        ;;POP STACK INTO R0
    005674  012600                              MOV     (SP)+,R0        ;;POP STACK INTO R0
968 005676  005077  173720          4$:         CLR     @CSR            ;CLEAR CSR
969 005702  012777  022600  173714  5$:         MOV     #OOPS,@VEC0     ;RESET RETURN
970                                             ;GO TO NEXT TEST
```

```
 972
 973                                        ;;*****************************************************************
                                            ;*TEST 13       RELEASE TEST WITHOUT BUSS MASTERSHIP.IDX11
                                            ;;*****************************************************************
     005710  000004                         TST13:  SCOPE
     005712  012767  000012  173252                 MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
 974 005720  004767  020452                         JSR     PC,CNTLC        ;IS IT CONTROL C?
 975
 976 005724  005000                         1$:     CLR     R0              ;SET DELAY TIMER
 977 005726  012777  000001  173666                 MOV     #1,@CSR         ;CONNECT
 978 005734  016767  174012  173162                 MOV     KONST,GOOD
 979 005742  062767  020201  173154                 ADD     #20201,GOOD     ;STORE GOOD DATA (SWB ACT,PWR OK,CON,REQ)
 980 005750  012767  000000  172020                 MOV     #0,PSW          ;DROP PRIORITY
 981 005756  017767  173640  173142  2$:     MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
 982 005764  026767  173134  173134                 CMP     GOOD,BAD        ;COMPARE
 983 005772  001404                                 BEQ     3$              ;BR IF CONNECTED
 984 005774  005200                                 INC     R0              ;DELAY
 985 005776  001367                                 BNE     2$              ;BR IF NOT DONE
 986 006000  104006                                 ERROR+6                 ;COULD NOT CONNECT
 987 006002  000461                                 BR      7$              ;EXIT TEST
 988 006004  012777  006142  173612  3$:     MOV     #6$,@VEC0       ;SET RETURN
 989 006012  012767  000340  171756                 MOV     #340,PSW        ;RISE PRIORITY
 990 006020  005000                                 CLR     R0              ;SET DELAY TIMER
 991 006022  012777  000100  173572                 MOV     #100,@CSR       ;CLEAR REQUEST AND SET IE
 992 006030  017767  173566  173070  4$:     MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
 993 006036  016767  173710  173060                 MOV     KONST,GOOD
 994 006044  062767  100100  173052                 ADD     #100100,GOOD    ;STORE GOOD DATA(TMO,PWR OK,IE)
 995 006052  005767  173556                         TST     DT03            ;DT03 MODE?
 996 006056  001003                                 BNE     41$             ;BR IF DT03
 997 006060  062767  000400  173036                 ADD     #400,GOOD       ;ADD BRQ IF DT07
 998 006066  026767  173032  173032  41$:    CMP     GOOD,BAD        ;COMPARE
 999 006074  001404                                 BEQ     5$              ;BR IF OK
1000 006076  005200                                 INC     R0              ;DELAY
1001 006100  001353                                 BNE     4$              ;BR IF NOT DONE
1002 006102  104016                                 ERROR+16                ;TMO NOT SET
1003 006104  000420                                 BR      7$              ;EXIT
1004 006106  016767  173640  173010  5$:     MOV     KONST,GOOD
1005 006114  062767  100100  173002                 ADD     #100100,GOOD    ;STORE GOOD DATA (TMO,PWR OK,IE)
1006 006122  005000                                 CLR     R0              ;RESET DELAY
1007 006124  012767  000000  171644                 MOV     #0,PSW          ;DROP PRIORITY
1008 006132  005200                         51$:    INC     R0              ;DELAY
1009 006134  001376                                 BNE     51$             ;BR IF NOT DONE
1010 006136  104010                                 ERROR+10                ;NO INTERUPT
1011 006140  000402                                 BR      7$              ;EXIT
1012 006142                                 6$:
     006142  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
     006144  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
1013 006146  012777  022600  173450  7$:     MOV     #OOPS,@VEC0     ;RESET RETURN
1014                                                                 ;GO TO NEXT TEST
```

```
1016
1017
1018                                    ;;*****************************************************************
                                        ;*TEST 14        RESET (RST) TEST.IDX12
                                        ;;*****************************************************************
     006154  000004                     TST14:  SCOPE
     006156  012767  000012  173006             MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
1019 006164  004767  020206                     JSR     PC,CNTLC        ;IS IT CONTROL C?
1020
1021 006170  005767  173020                     TST     $PASS           ;FIRST PASS?
1022 006174  001562                             BEQ     5$              ;SKIP TEST ON FIRST PASS
1023 006176  005767  173552                     TST     DEVCON          ;RESET IN NEUTRAL?
1024 006202  100557                             BMI     5$              ;BR=NO RESET IN NEUTRAL
1025 006204  005767  173426                     TST     DEVAD           ;IS A DEVICE REGISTER PRESENT?
1026 006210  001554                             BEQ     5$              ;EXIT IF NOT
1027 006212  005767  173424                     TST     DEVRC           ;RESETABLE BITS PRESENT?
1028 006216  001551                             BEQ     5$              ;BR=NO SO EXIT
1029 006220  012767  000000  171550             MOV     #0,PSW          ;DROP PRIORITY
1030 006226  012777  000001  173366             MOV     #REQ,@CSR       ;CONNECT
1031 006234  105777  173362          1$:        TSTB    @CSR            ;CONNECTED?
1032 006240  100417                             BMI     2$              ;BR IF YES
1033 006242  005777  173354                     TST     @CSR            ;TIMEOUT?
1034 006246  100372                             BPL     1$              ;BR IF NOT
1035 006250  017767  173346  172650             MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
1036 006256  016767  173470  172640             MOV     KONST,GOOD
1037 006264  062767  020201  172632             ADD     #20201,GOOD     ;STORE GOOD DATA(SWB ACT,PWR OK,CON,REQ)
1038 006272  104006                             ERROR+6                 ;COULDN'T CONNECT
1039 006274  000167  000242                     JMP     5$
1040
1041 006300  004767  020034          2$:        JSR     PC,CKADR        ;GO CHECK DEVICE ADDRESS EXIST?
1042 006304  005767  020026                     TST     FLAG            ;IF FLAG=1,TRAPPED
1043 006310  001403                             BEQ     20$
1044 006312  104027                             ERROR+27
1045 006314  000167  000222                     JMP     5$              ;EXIT
1046 006320                          20$:
1047 006320  016777  173314  173310             MOV     DEVRW,@DEVAD    ;SET READ/WRITE BITS IN EXTERNAL DEVICE REGISTER
1048 006326  017767  173304  172572             MOV     @DEVAD,BAD      ;STORE CONTENTS OF DEVAD
1049 006334  005167  173300                     COM     DEVRW           ;COMPLEMENT R/W BITS
1050 006340  046767  173274  172560             BIC     DEVRW,BAD       ;CLEAR OUT BITS NOT UNDER TEST
1051 006346  005167  173266                     COM     DEVRW           ;RESTORE ORIGINAL PATTERN
1052 006352  026767  173262  172546             CMP     DEVRW,BAD       ;DID THEY SET?
1053 006360  001402                             BEQ     3$              ;BR IF YES
1054 006362  104017                             ERROR+17                ;COULDN'T SET READ/WRITE BITS IN DEVAD
1055 006364  000466                             BR      5$              ;EXIT
1056 006366  052777  001000  173226  3$:        BIS     #1000,@CSR      ;SET RST
1057 006374  012767  000002  013236             MOV     #2,TOCK         ;SET UP FOR 22MSEC COUNT
1058 006402  004767  013172                     JSR     PC,WAIT
1059 006406  032777  001000  173206  31$:       BIT     #1000,@CSR      ;STILL SET?
1060 006414  001430                             BEQ     4$              ;BR=NO
1061 006416  104401  006424                     TYPE    ,65$            ;;TYPE ASCIZ STRING
     006422  000424                     BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <200>/ERROR! RST(BIT9) DIDN'T CLEAR IN TIME./
     006474                             64$:
1062 006474  000000                             HALT
1063 006476  017767  173134  172422  4$:        MOV     @DEVAD,BAD      ;STORE CONTENTS OF DEVAD
1064 006504  036767  173132  172414             BIT     DEVRC,BAD       ;RESET CONDITIONS MET?
1065 006512  001413                             BEQ     5$              ;BR IF YES
```

```
 1066 006514  005200                          INC     R0              ;WAIT AWHILE
 1067 006516  001367                          BNE     4$
 1068 006520  016767  173116  172376          MOV     DEVRC,GOOD      ;SAVE DEVRC
 1069 006526  005167  172372                  COM     GOOD
 1070 006532  046767  172366  172366          BIC     GOOD,BAD        ;HI LITE BAD BITS
 1071 006540  104021                          ERROR+21                ;RESET (RST) DIDN'T WORK
 1072 006542  005077  173054          5$:     CLR     @CSR            ;SHUT DOWN AND GO TO NEXT TEST
```

```
 1074
 1075                                    ;;*********************************************************************
                                         ;*TEST 15       RESET HOLD (HLD) TEST.IDX13
                                         ;;*********************************************************************
      006546  000004                     TST15:  SCOPE
      006550  012767  000012  172414             MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
 1076 006556  004767  017614                     JSR     PC,CNTLC        ;IS IT CONTROL C?
 1077
 1078 006562  005767  173046                     TST     DT03            ;DT03 MODE
 1079 006566  001043                             BNE     3$              ;BR IF YES
 1080 006570  012767  000000  171200             MOV     #0,PSW          ;DROP PRIORITY
 1081 006576  012777  000C01  173016             MOV     #REQ,@CSR       ;CONNECT
 1082 006604  105777  173012          1$:        TSTB    @CSR            ;CONNECTED?
 1083 006610  100416                             BMI     2$              ;BR IF YES
 1084 006612  005777  173004                     TST     @CSR            ;TIMEOUT?
 1085 006616  100372                             BPL     1$              ;BR IF NOT
 1086 006620  017767  172776  172300             MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
 1087 006626  016767  173120  172270             MOV     KONST,GOOD
 1088 006634  062767  020201  172262             ADD     #20201,GOOD     ;STORE GOOD DATA(SWB ACT,PWR OK,CON,REQ)
 1089 006642  104006                             ERROR+6                 ;COULDN'T CONNECT
 1090 006644  000414                             BR      3$              ;EXIT
 1091 006646  052777  000002  172746  2$:        BIS     #2,@CSR         ;SET HLD(BIT1)
 1092 006654  000005                             RESET                   ;DO RESET
 1093 006656  017767  172740  172242             MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
 1094 0C6664  032767  000200  172234             BIT     #200,BAD        ;STILL CONNECTED?
 1095 006672  001001                             BNE     3$              ;BR IF YES
 1096 006674  104020                             ERROR+20                ;HLD DIDN'T DISABLE RESET
 1097 006676  005077  172720          3$:        CLR     @CSR            ;SHUTDOWN AND GO TO NEXT TEST
```

```
1099
1100
1101                                    ;;*****************************************************************
                                        ;*TEST 16       HI-SPEED SWITCHING TEST.IDX14
                                        ;;*****************************************************************
     006702  000004                     TST16:  SCOPE
     006704  012767  000012  172260             MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
1102 006712  004767  017460                     JSR     PC,CNTLC        ;IS IT CONTROL C?
1103
1104 006716  012701  001750                     MOV     #1000.,R1       ;SET COUNTER
1105 006722  012767  000000  171046  11$:      MOV     #0,PSW          ;DROP PRIORITY
1106 006730  012777  000001  172664             MOV     #REQ,@CSR       ;CONNECT
1107 006736  105777  172660           1$:      TSTB    @CSR            ;CONNECTED?
1108 006742  100416                     BMI     2$              ;BR IF YES
1109 006744  005777  172652                     TST     @CSR            ;TIMEOUT?
1110 006750  100372                     BPL     1$              ;BR IF NOT
1111 006752  017767  172644  172146             MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
1112 006760  016767  172766  172136             MOV     KONST,GOOD
1113 006766  062767  020201  172130             ADD     #20201,GOOD     ;STORE GOOD DATA(SWB ACT,PWR OK,CON,REQ)
1114 006774  104006                     ERROR+6                         ;COULDN'T CONNECT
1115 006776  000422                     BR      5$                      ;EXIT
1116 007000  042777  000001  172614  2$:      BIC     #1,@CSR         ;RELEASE CONNECTION
1117 007006  016767  172740  172110             MOV     KONST,GOOD      ;STORE GOOD DATA(PWR OK)
1118 007014  005000                     CLR     R0              ;SET DELAY TIMER
1119 007016  017767  172600  172102  3$:      MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
1120 007024  026767  172074  172074             CMP     GOOD,BAD        ;COMPARE
1121 007032  001402                     BEQ     4$              ;BR IF OK
1122 007034  104015                     ERROR+15                        ;RELEASE CONDITIONS NOT MET
1123 007036  000402                     BR      5$                      ;EXIT
1124 007040  005301           4$:      DEC     R1              ;COUNT A SUBPASS
1125 007042  001327                     BNE     11$             ;BR IF CONTINUE
1126 007044  005077  172552           5$:      CLR     @CSR            ;SHUTDOWN AND GO TO NEXT TEST
1127
1128
1129
1130
1131                                    ;;*****************************************************************
                                        ;*TEST 17       RESET-IN-NEUTRAL TEST.IDX15
                                        ;;*****************************************************************
     007050  000004                     TST17:  SCOPE
     007052  012767  000001  172112             MOV     #1,$TIMES       ;;DO 1 ITERATION
1132
1133 007060  005767  172552                     TST     DEVAD           ;IS A DEVICE REGISTER PRESENT?
1134 007064  001012                     BNE     15$             ;CONTINUE TEST IF PRESENT
1135 007066  000167  000446                     JMP     12$             ;GO EXIT
1136 007072                           13$:
1137 007072  004767  017242                     JSR     PC,CKADR        ;CHECK IF DEVICE ADDRESS EXIST?
1138 007076  005767  017234                     TST     FLAG            ;IF FLAG=1;TRAP OCCURED
1139 007102  001437                     BEQ     2$              ;BR:NO TRAP
1140 007104  104027                     ERROR+27
1141 007106  000167  000426                     JMP     12$             ;GO EXIT
1142 007112  005767  172524           15$:      TST     DEVRC           ;ANY RESETABLE BITS?
1143 007116  001002                     BNE     14$             ;BR=YES
1144 007120  000167  000414                     JMP     12$             ;GO EXIT
1145 007124  012767  000000  170644  14$:      MOV     #0,PSW          ;DROP PRIORITY
1146 007132  012777  000001  172462             MOV     #REQ,@CSR       ;CONNECT
1147 007140  105777  172456           1$:      TSTB    @CSR            ;CONNECTED?
```

```
1148 007144  100752                                    BMI     13$             ;BR IF YES
1149 007146  005777  172450                            TST     @CSR            ;TIMEOUT?
1150 007152  100372                                    BPL     1$              ;BR IF NOT
1151 007154  017767  172442  171744                    MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
1152 007162  016767  172564  171734                    MOV     KONST,GOOD
1153 007170  062767  020201  171726                    ADD     #20201,GOOD     ;STORE GOOD DATA(SWB ACT,PWR OK,CON,REQ)
1154 007176  104006                                    ERROR+6                 ;COULDN'T CONNECT
1155 007200  000557                                    BR      12$             ;EXIT
1156 007202                                    2$:
1157 007202  016777  172432  172426                    MOV     DEVRW,@DEVAD    ;SET READ/WRITE BITS IN EXTERNAL DEVICE REGISTER
1158 007210  017767  172422  171710                    MOV     @DEVAD,BAD      ;STORE CONTENTS OF DEVAD
1159 007216  005167  172416                            COM     DEVRW           ;COMPLEMENT R/W BITS
1160 007222  046767  172412  171676                    BIC     DEVRW,BAD       ;CLEAR OUT BITS NOT UNDER TEST
1161 007230  005167  172404                            COM     DEVRW           ;RESTORE ORIGINAL PATTERN
1162 007234  026767  172400  171664                    CMP     DEVRW,BAD       ;DID THEY SET?
1163 007242  001402                                    BEQ     3$              ;BR IF YES
1164 007244  104017                                    ERROR+17                ;COULDN'T SET READ/WRITE BITS IN DEVAD
1165 007246  000534                                    BR      12$             ;EXIT
1166 007250  005077  172346            3$:     CLR     @CSR            ;DISCONNECT
1167 007254  105777  172342            4$:     TSTB    @CSR            ;DONE?
1168 007260  100775                                    BMI     4$              ;BR=NO
1169 007262  012767  000000  170506                    MOV     #0,PSW          ;DROP PRIORITY
1170 007270  012777  000001  172324                    MOV     #REQ,@CSR       ;CONNECT
1171 007276  105777  172320            5$:     TSTB    @CSR            ;CONNECTED?
1172 007302  100416                                    BMI     6$              ;BR IF YES
1173 007304  005777  172312                            TST     @CSR            ;TIMEOUT?
1174 007310  100372                                    BPL     5$              ;BR IF NOT
1175 007312  017767  172304  171606                    MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
1176 007320  016767  172426  171576                    MOV     KONST,GOOD
1177 007326  062767  020201  171570                    ADD     #20201,GOOD     ;STORE GOOD DATA(SWB ACT,PWR OK,CON,REQ)
1178 007334  104006                                    ERROR+6                 ;COULDN'T CONNECT
1179 007336  000500                                    BR      12$             ;EXIT
1180 007340  017767  172272  171560  6$:     MOV     @DEVAD,BAD      ;STORE CONTENTS OF DEVAD
1181 007346  036767  172270  171552                    BIT     DEVRC,BAD       ;RESET CONDITIONS MET?
1182 007354  001403                                    BEQ     7$              ;BR=YES
1183 007356  012702  177777                            MOV     #-1,R2          ;SET R2 FOR NO RESET
1184 007362  000401                                    BR      8$
1185 007364  005002                            7$:     CLR     R2              ;SET R2 FOR RESET
1186 007366  005767  171622            8$:     TST     $PASS           ;FIRST PASS?
1187 007372  001056                                    BNE     11$             ;BR=NO
1188 007374  005702                                    TST     R2              ;CHECK RESET CONDITION
1189 007376  001427                                    BEQ     9$              ;BR=RESET
1190 007400  104401  007406                            TYPE    ,65$            ;;TYPE ASCIZ STRING
     007404  000423                                    BR      64$             ;;GET OVER THE ASCIZ
                                       ;;65$:   .ASCIZ  <200>/THIS DT07 INHIBITS RESET-IN-NEUTRAL./
     007454                            64$:
1191 007454  000422                                    BR      10$
1192 007456                            9$:
     007456  104401  007464                            TYPE    ,67$            ;;TYPE ASCIZ STRING
     007462  000417                                    BR      66$             ;;GET OVER THE ASCIZ
                                       ;;67$:   .ASCIZ  <200>/THIS DT07 RESETS IN NEUTRAL./
     007522                            66$:
1193 007522  010267  172226            10$:    MOV     R2,DEVCON       ;STORE CONDITION
1194 007526  000404                                    BR      12$
1195 007530  020267  172220            11$:    CMP     R2,DEVCON       ;HAS IT CHANGED?
1196 007534  001401                                    BEQ     12$             ;BR=NO
1197 007536  104022                                    ERROR+22                ;RESET-IN-NEUTRAL OPERATION HAS CHANGED
```

SEQ 0038

```
1198 007540  005077  172056          12$:    CLR     @CSR            ;CLEAR AND DISCONNECT
1199
1200
```

```
 1202
 1203
 1204 007544  012767  002526  011656  MONEND: MOV     #RESTRT,$RTNAD  ;SET RETURN
 1205 007552  000167  011552                  JMP     $EOP            ;GO DO END PASS
 1206
```

```
                                        .SBTTL  MULTIPORT TEST

                                MULPRT:
007556  004767  016614                        JSR     PC,CNTLC        ;IS IT CONTROL C?
007562  005767  016416                        TST     PARFLG          ;CHECK FOR PARAMETERS
007566  100404                              1$: BMI     1$              ;BR=YES
007570  004767  016602                        JSR     PC,CNTLC        ;IS IT CONTROL C?
007574  004767  014734                        JSR     PC,PARIN        ;GO INPUT PARAMETERS
007600  012767  000100  172120          1$:   MOV     #64.,CYCLE      ;SET COUNTER
007606  005767  172142                        TST     DEVCON          ;RESET IN NEUTRAL OPTION USED?
007612  001003                              2$: BNE     2$              ;BR=NO
007614  005067  172136                        CLR     TRIN            ;SET FLAG FOR TEST
007620  000403                              3$: BR      2$+6            ;CONTINUE
007622  012767  177777  172126          2$:   MOV     #-1,TRIN        ;SET FLAG FOR NO TEST
007630  005767  172000                        TST     DT03            ;DT03 MODE?
007634  001404                              4$: BEQ     3$              ;BR=DT07 MODE
007636  004767  016534                        JSR     PC,CNTLC        ;IS IT CONTROL C?
007642  000167  002672                        JMP     MULDT3          ;GO DO DT03 MULTIPORT
007646  012777  022600  171750          3$:   MOV     #OOPS,@VEC0     ;SET RETURN
007654  012777  000340  171744                MOV     #340,@VEC2      ;SET LEVEL
007662  004767  012210                        JSR     PC,SEQMAK       ;GO CONSTRUCT SEQUENCER
007666  004767  012552                        JSR     PC,SEQINT       ;INIT. SEQUENCER
007672  012767  000000  170076                MOV     #0,PSW          ;DROP PRIORITY
007700  026705  171740                        CMP     PORTNO,R5       ;MASTER OR SLAVE?
007704  001063                              5$: BNE     SLAVE           ;BR=SLAVE
```

```
1235                                    ;*****************************************************************************
1236                                    ; ON FIRST PASS WAIT BEFORE CONNECTING TO THE SWITCHED BUS SO THAT SLAVES CAN
1237                                    ; CLEANUP AND READY THENSELVES FOR THE NEXT SEQUENCE,MASTER NOT AHEAD OF SLAVE
1238                                    ;*****************************************************************************
1239 007706  012767  000632  011724            MOV     #632,TOCK       ;SET CLOCK FOR 5 SEC
1240 007714  004767  011660                    JSR     PC,WAIT
1241 007720  000407                            BR      MASTER          ;GO TO MASTER CODE AFTER SYNC WAIT
1242
1243
1244 007722  026705  171716          DISPAT: CMP     PORTNO,R5       ;MY TURN AS MASTER?
1245 007726  001404                            BEQ     MASTER          ;BR=YES
1246 007730  004767  016442                    JSR     PC,CNTLC        ;IS IT CONTROL C?
1247 007734  000167  000114                    JMP     SLAVE           ;GO BE SLAVE
1248
1249                                    ;MASTER  TESTS DISPATCHING
1250
1251 007740                          MASTER:
1252 007740  004767  016432                    JSR     PC,CNTLC        ;IS IT CONTROL C?
1253 007744  004767  012236                    JSR     PC,MASMEN       ;GO MAKE MASTER MENU
1254 007750  005067  171746          1$:       CLR     CURFUN          ;INSURE NO EXTRANEOUS BITS PRESENT
1255 007754  116467  000001  171740            MOVB    1(R4),CURFUN    ;GET CURRANT FUNCTION
1256 007762  005067  171736                    CLR     CURID           ;CLEAN IT OUT
1257 007766  111467  171732                    MOVB    (R4),CURID      ;GET CURRENT ID
1258 007772  000241                            CLC
1259 007774  006167  171722                    ROL     CURFUN          ;DOUBLE IT
1260 010000  012703  012524                    MOV     #TABM-2,R3      ;LOAD BASE ADDRESS
1261 010004  066703  171712                    ADD     CURFUN,R3       ;ADD OFFSET
1262 010010  011303                            MOV     @R3,R3          ;GET ACTUAL ADDRESS
1263 010012  004713                            JSR     PC,@R3          ;GO DO IT
1264 010014  005724                            TST     (R4)+           ;STEP THRU MENU
1265 010016  022714  177777                    CMP     #-1,(R4)        ;DONE MENU?
1266 010022  001352                            BNE     1$              ;BR=NO
1267 010024  004767  012436                    JSR     PC,SEQSTP       ;GO STEP SEQUENCER
1268 010030  005367  171672                    DEC     CYCLE           ;COUNT
1269 010034  001402                            BEQ     2$              ;BR IF DONE
1270 010036  000167  177660                    JMP     DISPAT          ;GO START AGAIN
1271 010042  012767  007556  011360  2$:       MOV     #MULPRT,$RTNAD  ;SET UP RETURN
1272 010050  000167  011256                    JMP     $EOP+2          ;GO DO END PASS
```

```
1274                                    ;       SLAVE TEST DISPATCHING
1275
1276 010054  004767  012214            SLAVE:  JSR     PC,SALMEN       ;GO MAKE SLAVE MENU
1277                                    ;       JSR     PC,SYNCUP       ;GO SYNCUP! 2 MINUTE TIME OUT
1278 010060  005067  171636            1$:     CLR     CURFUN          ;CLEAN IT
1279 010064  116467  000001  171630            MOVB    1(R4),CURFUN    ;GET CURRENT FUNCTION
1280 010072  005067  171626                    CLR     CURID           ;CLEAN IT
1281 010076  111467  171622                    MOVB    (R4),CURID      ;GET CURRENT ID
1282 010102  000241                            CLC
1283 010104  006167  171612                    ROL     CURFUN          ;DOUBLE IT
1284 010110  012703  012524                    MOV     #TABM-2,R3      ;LOAD BASE ADDRESS
1285 010114  066703  171602                    ADD     CURFUN,R3       ;ADD OFFSET
1286 010120  011303                            MOV     @R3,R3          ;GET ACTUAL ADDRESS
1287 010122  004713                            JSR     PC,@R3          ;GO DO IT
1288 010124  005724                            TST     (R4)+           ;STEP THRU MENU
1289 010126  022714  177777                    CMP     #-1,(R4)        ;DONE MENU
1290 010132  001352                            BNE     1$              ;BR=NO
1291 010134  004767  012326                    JSR     PC,SEQSTP       ;GO STEP SEQUENCER
1292 010140  005367  171562                    DEC     CYCLE           ;COUNT
1293 010144  001402                            BEQ     2$              ;BRIF DONE
1294 010146  000167  177550                    JMP     DISPAT          ;GO START AGAIN
1295 010152  012767  007556  011250  2$:      MOV     #MULPRT,$RTNAD  ;SET UP RETURN
1296 010160  000167  011146                    JMP     $EOP+2          ;GO DO END PASS
```

```
                                        ;*****************************************************************
                                        ;EXTERNAL CONNECT REQUEST REJECTED                (MASTER_TEST_1)
                                        ;*****************************************************************
                                        FUNCT1:
                                                JSR     PC,CNTLC        ;IS IT CONTROL C?

                                                MOV     #100000,R0      ;KILL TIME FOR OTHERS TO SEE TRANSISTIONS
                                                DEC     R0              ;<<===
                                                BNE     .-2             ;<<==

                                                JSR     PC,CONNEC       ;GO CONNECT
                                                JSR     PC,CONID        ;GO CONVERT SLAVE ID
                                                MOV     #1$,@VEC0       ;SET RETURN
                                                MOVB    #101,@CSR       ;ENABLE INTERUPT

                                                CLR     R0              ;SET TIMER
                                                MOV     #60,R1          ;MULTIPLIER<<==========
                                        5$:     INC     R0              ;TIME OUT<<=======
                                                BNE     .-2             ;BR IF NOT DONE
                                                DEC     R1              ;ALLOW TIME FOR HDW DELAY<<========
                                                BNE     5$              ;AFTER CONNECTING TO THE SWITCHED BUS<<======

                                                TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <200>/ERROR1A! NO EXTERNAL REQUEST INTERUPT DETECTED.  CSR= /
                                        64$:
                                                MOV     @CSR,BAD        ;SAVE CSR
                                                MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
                                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                                HALT
                                                BR      .-2
                                        1$:
                                                MOV     (SP)+,R0        ;;POP STACK INTO R0
                                                MOV     (SP)+,R0        ;;POP STACK INTO R0
                                                MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
                                                BIT     #10000,BAD      ;MAKE SURE EXT INT IS SET
                                                BNE     2$              ;BR = OK
                                                TYPE    ,67$            ;;TYPE ASCIZ STRING
                                                BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  <200>/ERROR1B! EXTERNAL INTERUPT NOT SET.  CSR= /
                                        66$:
                                                MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
                                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                                HALT
                                                BR      .-2
                                        2$:     BIT     COVID,BAD       ;CORRECT SLAVE ID?
                                                BNE     3$              ;BR=YES
                                                TYPE    ,69$            ;;TYPE ASCIZ STRING
                                                BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  <200>/ERROR1C! EXTERNAL REQUEST FROM WRONG PORT.  CSR= /
                                        68$:
                                                MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
                                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                                HALT
                                                BR      .-2
                                        3$:     MOV     #101,@CSR       ;CLEAR EXT INT REJECT THE REQUEST FROM SLAVE
                                                MOV     #OOPS,@VEC0     ;RESET RETURN
```

SEQ 0044

```
 1341
 1342 010626  012700  100000          MOV     #100000,R0      ;KILL SOME TIME<<===
 1343 010632  005300                  DEC     R0              ;<<===
 1344 010634  001376                  BNE     .-2
 1345
 1346 010636  005077  170760          CLR     @CSR            ;DISC SWB TO NEUTRAL STATE<<===========
 1347
 1348 010642  012700  001600          MOV     #1600,R0        ;KILL SOME TIME<<===
 1349 010646  005300                  DEC     R0              ;SO OTHERS CAN SEE XSION<<===
 1350 010650  001376                  BNE     .-2             ;OF SWB.ACT BIT 0=>1  <<===
 1351
 1352 010652  005067  167120          CLR     PSW             ;LOWER PRIORITY
 1353 010656  000207                  RTS     PC
```

```
                                ;*****************************************************************
                                ;EXTERNAL REQUEST IGNORED                (MASTER_TEST_2)
                                ; (MASTER FAILS OVER TO SLAVE)
                                ;*****************************************************************

010660                          FUNCT2:
010660  004767  015512                  JSR     PC,CNTLC        ;IS IT CONTROL C?

                                ;*************************************************************************
                                ;ALLOW TIME FOR OTHERS TO DISPATCH AND SEE THE SWB.ACT. GO 0 TO 1
010664  012700  100000                  MOV     #100000,R0      ;STALL TIME FOR OTHERS TO SEE SWB.ACT.=0<<=====
                                                                ;THEN =1 FOR THE WINK OF SWB.ACT. <<========
010670  005300                          DEC     R0              ;DO SOME TIME HERE <<==========
010672  001376                          BNE     .-2             ;<<========
                                ;*************************************************************************
010674  004767  010552                  JSR     PC,CONNEC       ;GO CONNECT
010700  004767  010604                  JSR     PC,CONID        ;GO CONVERT SLAVE ID
010704  012777  011206  170712          MOV     #1$,@VEC0       ;SET RETURN
010712  005767  171040                  TST     TRIN            ;CAN WE TEST RESET IN NEUTRAL?
010716  001052                          BNE     6$              ;BR=NO
010720  016777  170714  170710          MOV     DEVRW,@DEVAD    ;LOAD READ/WRITE BITS
010726  026777  170706  170702          CMP     DEVRW,@DEVAD    ;LOADED?
010734  001443                          BEQ     6$              ;BR=YES
010736  104401  010744                  TYPE    ,65$            ;;TYPE ASCIZ STRING
010742  000432                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>/ERROR2F! COULDN'T LOAD EXTERNAL DEVICE REGISTER.../
011030                          64$:
011030  016746  170602                  MOV     DEVAD,-(SP)     ;;SAVE DEVAD FOR TYPEOUT
011034  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
011036  012767  177777  170712          MOV     #-1,TRIN        ;SET FLAG FOR NO TEST

011044  112777  000101  170550  6$:     MOVB    #101,@CSR       ;ENABLE INTERUPT
                                ;THIS IS ABOUT 4 SECS.
011052  012701  000060                  MOV     #60,R1          ;MUST ALLOW SUFFICIENT TIME FOR HRDW DELAY<===
011056  005000                          CLR     R0              ;SET TIMER
011060  005200                  15$:    INC     R0              ;TIME OUT
011062  001376                          BNE     .-2             ;BR IF NOT DONE
011064  005301                          DEC     R1              ;<<==========
011066  001374                          BNE     15$             ;NE= ADD ANOTHER LOOP<<=====

011070  104401  011076                  TYPE    ,67$            ;;TYPE ASCIZ STRING
011074  000434                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <200>/ERROR2A! NO EXTERNAL REQUEST INTERUPT DETECTED.  CSR= /
011166                          66$:
011166  017767  170430  167732          MOV     @CSR,BAD        ;SAVE CSR
011174  016746  167726                  MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
011200  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
011202  000000                          HALT
011204  000776                          BR      .-2
011206                          1$:
011206  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
011210  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
011212  017767  170404  167706          MOV     @CSR,BAD        ;STORE CONTENTS OF CSR
011220  032767  010000  167700          BIT     #10000,BAD      ;MAKE SURE EXT INT IS SET
011226  001036                          BNE     2$              ;BR = OK
011230  104401  011236                  TYPE    ,69$            ;;TYPE ASCIZ STRING
011234  000426                          BR      68$             ;;GET OVER THE ASCIZ
```

```
                                        ;;69$:  .ASCIZ  <200>/ERROR2B! EXTERNAL INTERUPT NOT SET.  CSR= /
     011312                             68$:
1401 011312  016746  167610                    MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
     011316  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1402 011320  000000                            HALT
1403 011322  000776                            BR      .-2
1404 011324  036767  170400  167574  2$:      BIT     COVID,BAD       ;CORRECT SLAVE ID?
1405 011332  001042                            BNE     3$              ;BR=YES
1406 011334  104401  011342                    TYPE    ,71$            ;;TYPE ASCIZ STRING
     011340  000432                            BR      70$             ;;GET OVER THE ASCIZ
                                        ;;71$:  .ASCIZ  <200>/ERROR2C! EXTERNAL REQUEST FROM WRONG PORT.  CSR= /
     011426                             70$:
1407 011426  016746  167474                    MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
     011432  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1408 011434  000000                            HALT
1409 011436  000776                            BR      .-2
1410 011440  012777  011556  170156  3$:      MOV     #4$,@VEC0       ;SET RETURN
1411 011446  005067  166324                    CLR     PSW             ;LOWER PRIORITY
1412 011452  004767  010110                    JSR     PC,WAIT05       ;WAIT
1413 011456  104401  011464                    TYPE    ,73$            ;;TYPE ASCIZ STRING
     011462  000430                            BR      72$             ;;GET OVER THE ASCIZ
                                        ;;73$:  .ASCIZ  <200>/ERROR2D! SLAVE DIDN'T TAKE SHARED BUSS. CSR= /
     011544                             72$:
1414 011544  017746  170052                    MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     011550  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1415 011552  000000                            HALT
1416 011554  000776                            BR      .-2
1417 011556                             4$:
     011556  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
     011560  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
     011562  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
1418 011564  005777  170032                    TST     @CSR            ;TIMEOUT SET?
1419 011570  100436                            BMI     5$              ;BR=YES
1420 011572  104401  011600                    TYPE    ,75$            ;;TYPE ASCIZ STRING
     011576  000423                            BR      74$             ;;GET OVER THE ASCIZ
                                        ;;75$:  .ASCIZ  <200>/ERROR2E! NO TMO ON INTERUPT.  CSR= /
     011646                             74$:
1421 011646  017767  167750  167252            MOV     @CSR,BAD        ;SAVE CSR
1422 011654  016746  167246                    MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
     011660  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1423 011662  000000                            HALT
1424 011664  000776                            BR      .-2
1425 011666  012777  022600  167730  5$:      MOV     #00PS,@VEC0     ;RESET RETURN
1426 011674  005077  167722                    CLR     @CSR            ;CLEAR CSR
1427 011700  005067  166072                    CLR     PSW             ;LOWER PRIORITY
1428 011704  000207                            RTS     PC              ;EXIT
```

```
 1430
 1431                                   ;*****************************************************************
 1432                                   ;SEND CONNECT REQUEST EXPECT REJECTION          (SLAVE_TEST_3)
 1433                                   ;*****************************************************************
 1434
 1435 011706                           FUNCT3:
 1436 011706  004767  014464                   JSR     PC,CNTLC        ;IS IT CONTROL C?
 1437 011712  004767  007724                   JSR     PC,SYNCUP       ;WAIT FOR MASTER TO START
 1438                                   ;*************************************************************************************
 1439                                   ;ALLOW 400 + M.S. FOR HRDW DELAY TO PASS SO EXT REQ. CAN BE SEEN BY MASTER
 1440
 1441 011716  012701  000010                   MOV     #10,R1          ;STALL FOR 300 + MS<<========
 1442 011722  005000                           CLR     R0              ;<<============
 1443 011724  005200                   15$:    INC     R0              ;SPIN LOOP= 65KX300X10-6 OR 300+M.S.<<====
 1444 011726  001376                           BNE     .-2
 1445 011730  005301                           DEC     R1              ;<<=====
 1446 011732  001374                           BNE     15$             ;DO ANOTHER LOOP=NE
 1447                                   ;*************************************************************************************
 1448 011734  012777  011752  167662           MOV     #3$,@VEC0       ;SET RETURN
 1449 011742  012777  000101  167652           MOV     #101,@CSR       ;SET IE + REQ
 1450 011750  000001                           WAIT                    ;WAIT FOR INTERUPT
 1451 011752                           3$:
      011752  012600                           MOV     (SP)+,R0        ;;POP STACK INTO R0
      011754  012600                           MOV     (SP)+,R0        ;;POP STACK INTO R0
 1452 011756  005777  167640                   TST     @CSR            ;TIMEOUT?
 1453 011762  100435                           BMI     4$              ;BR=YES
 1454 011764  017767  167632  167134           MOV     @CSR,BAD        ;SAVE CSR
 1455 011772  104401  012000                   TYPE    ,65$            ;;TYPE ASCIZ STRING
      011776  000422                           BR      64$             ;;GET OVER THE ASCIZ
      012044                           ;;65$:   .ASCIZ  <200>/ERROR3B! NO TIMEOUT DETECTED. CSR=/
      012044                           64$:
 1456 012044  016746  167056                   MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
      012050  104402                           TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1457 012052  000000                           HALT
 1458 012054  000776                           BR      .-2
 1459 012056  005077  167540           4$:     CLR     @CSR            ;CLEAR + DISCONNECT
 1460 012062  012777  022600  167534           MOV     #OOPS,@VEC0     ;RESET RETURN
 1461 012070  005067  165702                   CLR     PSW             ;LOWER PRIORITY
 1462 012074  000207                           RTS     PC              ;EXIT
```

```
                        ;*****************************************************************************
                        ;SEND CONNECT REQUEST EXPECT CONNECTION                    (SLAVE_TEST_4)
                        ;*****************************************************************************

012076                  FUNCT4:
012076  004767  014274          JSR     PC,CNTLC        ;IS IT CONTROL C?
012102  004767  007534          JSR     PC,SYNCUP       ;WAIT FOR MASTER
                        ;*****************************************************************************
                        ;ALLOW 400 + M.S. FOR HRDW DELAY TO PASS SO EXT REQ. CAN BE SEEN BY MASTER

012106  012701  000010          MOV     #10,R1          ;STALL FOR 300 + MS<<========
012112  005000                  CLR     R0              ;<<============
012114  005200          15$:    INC     R0              ;SPIN LOOP= 65KX300X10-6 OR 300+M.S.<<====
012116  001376                  BNE     .-2
012120  005301                  DEC     R1              ;<<=====
012122  001374                  BNE     15$             ;DO ANOTHER LOOP=NE
                        ;*****************************************************************************
                        ;       MOV     TIMA,TEMP1      ;SET TIMER
                        ;       ASL     TEMP1
                        ;21$:   BIT     #20000,@CSR     ;SWB ACT SET?
                        ;       BEQ     FUNCT4  ;BR=NO
                        ;       DEC     TEMP1
                        ;       BNE     21$

012124  012777  012142  167472        MOV     #3$,@VEC0       ;SET RETURN
012132  012777  000101  167462        MOV     #101,@CSR       ;SET IE + REQ
012140  000001                        WAIT                    ;WAIT FOR INTERUPT
012142                  3$:
012142  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
012144  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
012146  105777  167450                TSTB    @CSR            ;CONNECTED?
012152  100437                        BMI     4$              ;BR=YES
012154  017767  167442  166744        MOV     @CSR,BAD        ;SAVE CSR
012162  104401  012170                TYPE    ,65$            ;;TYPE ASCIZ STRING
012166  000424                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>/ERROR4B! CONNECT REQUEST FAILED. CSR=/
012240                          64$:
012240  016746  166662                MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
012244  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012246  000000                        HALT
012250  000776                        BR      .-2
012252  005767  167500          4$:   TST     TRIN            ;CAN WE TEST RESET IN NEUTRAL
012256  001036                        BNE     5$              ;BR=NO
012260  036777  167356  167350        BIT     DEVRC,@DEVAD    ;RESET CONDITIONS MET?
012266  001432                        BEQ     5$              ;BR=YES
012270  104401  012276                TYPE    ,67$            ;;TYPE ASCIZ STRING
012274  000424                        BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <200>/ERROR4C! NO RESET IN NEUTRAL DETECTED./
012346                          66$:
012346  012767  177777  167402        MOV     #-1,TRIN        ;CANCEL FURTHER TESTING OF RESET IN NEUTRAL
012354  005077  167242          5$:   CLR     @CSR            ;CLEAR + DISCONNECT
012360  012777  022600  167236        MOV     #OOPS,@VEC0     ;RESET RETURN
012366  005067  165404                CLR     PSW             ;LOWER PRIORITY
012372  000207                        RTS     PC              ;EXIT
```

```
                                ;*****************************************************************************
                                ;OBSERVE REQUEST INDICATORS                                        (SLAVE_TEST_5)
                                ;*****************************************************************************

012374                          FUNCT5:
012374  004767  013776                JSR     PC,CNTLC        ;IS IT CONTROL C?
012400  004767  007104                JSR     PC,CONID        ;GO CONVERT PORT ID
012404  005000                        CLR     R0              ;SET TIMERS
012406  012701  000400                MOV     #400,R1
012412  036777  167312  167202  1$:   BIT     COVID,@CSR      ;LOOK FOR REQUEST
012420  001035                        BNE     2$              ;BR=FOUND IT
012422  005200                        INC     R0              ;TIME OUT
012424  001372                        BNE     1$
012426  005301                        DEC     R1
012430  001370                        BNE     1$
012432  104401  012440                TYPE    ,65$            ;;TYPE ASCIZ STRING
012436  000424                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>/ERROR5A! NO REQUEST ACTIVITY DETECTED./
012510                          64$:
012510  000000                        HALT
012512  000776                        BR      .-2
012514  036777  167210  167100  2$:   BIT     COVID,@CSR      ;DID IT GO AWAY?
012522  001374                        BNE     2$              ;BR=NOT YET
012524  000207                        RTS     PC              ;EXIT

012526  010164                TABM:   FUNCT1                  ;DISPATCH TABLE FOR MULTIPORT TEST
012530  010660                        FUNCT2
012532  011706                        FUNCT3
012534  012076                        FUNCT4
012536  012374                        FUNCT5
```

```
                                        .SBTTL
                                        .SBTTL  BIPORT TEST IN DT03 MODE
                                        .SBTTL
012540                                  MULDT3:
012540  004767  013632                          JSR     PC,CNTLC        ;IS IT CONTROL C?
012544  012767  000000  165224                  MOV     #0,PSW          ;LOWER PRIORITY
012552  005077  167044                          CLR     @CSR            ;START WITH A CLEAR REGISTER
012556  012767  002000  167142                  MOV     #2000,CYCLE     ;SET SUB COUNT
012564  005067  006774                          CLR     TRIG            ;CLEAR TRIGGER
012570  032777  020000  167024          1$:     BIT     #20000,@CSR     ;ACTIVE?
012576  001401                                  BEQ     11$             ;BR=NO
012600  000464                                  BP      21$             ;GO BE SLAVE
012602  012777  012726  167014          11$:    MOV     #2$,@VEC0       ;SET RETURN
012610  012777  000101  167004                  MOV     #101,@CSR       ;SEND REQUEST
012616  004767  006730                          JSR     PC,WAIT20       ;GO WAIT
012622  104401  012630                          TYPE    ,65$            ;;TYPE ASCIZ STRING
012626  000432                                  BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <200>/ERROR6! NO INTERUPT AFTER CONNECT REQUEST.  CSR= /
012714                                  64$:
012714  017746  166702                          MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
012720  104402                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012722  000000                                  HALT
012724  000776                                  BR      .-2
012726  012706  001100          2$:     MOV     #STACK,SP       ;RESET STACK
012732  005067  165040                          CLR     PSW             ;LOWER PRIORITY
012736  005777  166660                          TST     @CSR            ;TIMEOUT?
012742  100007                                  BPL     3$              ;BR=NO
012744  000240                                  NOP
012746  000240                                  NOP
012750  000240                                  NOP
012752  005067  166766          21$:    CLR     ONOFF           ;INDIACATE FIRST SLAVE
012756  000167  001166                          JMP     SECT4           ;GO TO SECTION 4
012762  012767  177777  166754  3$:     MOV     #-1,ONOFF       ;INDIACATE FIRST MASTER

                                        ;SECTION 2 : WAIT FOR EXTERNAL REQUEST THEN RELEASE SWITCH

012770  005767  166762          SECT2:  TST     TRIN            ;CAN WE TEST RESET IN NEUTRAL
012774  001052                          BNE     2$              ;BR=NO
012776  016777  166636  166632          MOV     DEVRW,@DEVAD    ;LOAD DEVICE REGISTER
013004  026777  166630  166624          CMP     DEVRW,@DEVAD    ;OK?
013012  001443                          BEQ     2$              ;BR=YES
013014  104401  013022                  TYPE    ,65$            ;;TYPE ASCIZ STRING
013020  000432                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>/ERROR7A! COULDN'T LOAD EXTERNAL DEVICE REGISTER../
013106                          64$:
013106  016746  166524                  MOV     DEVAD,-(SP)     ;;SAVE DEVAD FOR TYPEOUT
013112  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
013114  012767  177777  166634          MOV     #-1,TRIN        ;CANCEL FURTHER TESTING OF RESET IN NEUTRAL
013122  012777  013246  166474  2$:     MOV     #1$,@VEC0       ;SET RETURN
013130  012767  000632  006502          MOV     #632,TOCK       ;SET DELAY FOR 5 SECONDS
013136  004767  006436                  JSR     PC,WAIT
013142  104401  013150                  TYPE    ,67$            ;;TYPE ASCIZ STRING
013146  000432                          BR      66$             ;;GET OVER THE ASCIZ
```

```
                                        ;;67$:  .ASCIZ  <200>/ERROR7! NO EXTERNAL INTERUPT REQUEST SEEN.  CSR= /
     013234                             66$:
1590 013234  017746  166362                    MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     013240  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1591 013242  000000                            HALT
1592 013244  000776                            BR      .-2
1593 013246  012706  001100               1$:  MOV     #STACK,SP       ;RESET STACK
1594 013252  016767  166474  165644            MOV     KONST,GOOD
1595 013260  062767  030301  165636            ADD     #30301,GOOD
1596 013266  026777  165632  166326            CMP     GOOD,@CSR       ;MAKE SURE IT'S AN EXTERNAL REQUEST
1597 013274  001435                            BEQ     SECT2A          ;BR=OK
1598 013276  104401  013304                    TYPE    ,69$            ;;TYPE ASCIZ STRING
     013302  000425                            BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  <200>/ERROR8! UNEXPECTED INTERUPT TYPE.  CSR= /
     013356                             68$:
1599 013356  017746  166240                    MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     013362  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1600 013364  000000                            HALT
1601 013366  000776                            BR      .-2
1602 013370  000401                    SECT2A: BR      1$              ;RELEASE TYPE SWITCHBACK
1603 013372  000420                            BR      SECT2B          ;GO DO PASSIVE RELEASE
1604 013374  012767  000240  177766       1$:  MOV     #240,SECT2A     ;TURN SWITCHBACK
1605 013402  005077  166214            SECT2D: CLR     @CSR            ;DROP SWITCH ACTIVELY
1606 013406  005067  164364                    CLR     PSW             ;LOWER PRIORITY
1607 013412  105777  166204                    TSTB    @CSR            ;DROPPED?
1608 013416  100775                            BMI     .-4             ;BR=NO
1609 013420  032777  020000  166174            BIT     #20000,@CSR     ;PICKED UP?
1610 013426  001774                            BEQ     .-6             ;BR=NO
1611 013430  000167  000230                    JMP     SECT2C          ;GO TO EXIT
1612 013434  012767  000401  177726  SECT2B:   MOV     #401,SECT2A     ;TURN SWITCHBACK
1613 013442  012777  013556  166154            MOV     #1$,@VEC0       ;SET RETURN
1614 013450  005067  164322                    CLR     PSW             ;LOWER PRIORITY
1615 013454  004767  006072                    JSR     PC,WAIT20       ;GO WAIT FOR TIMEOUT
1616 013460  104401  013466                    TYPE    ,65$            ;;TYPE ASCIZ STRING
     013464  000427                            BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <200>/ERROR9! NO TIMEOUT INTERUPT DETECTED. CSR= /
     013544                             64$:
1617 013544  017746  166052                    MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     013550  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1618 013552  000000                            HALT
1619 013554  000776                            BR      .-2
1620 013556  012706  001100               1$:  MOV     @STACK,SP       ;RESET STACK
1621 013562  005777  166034                    TST     @CSR            ;MAKE SURE IT'S A TIMEOUT
1622 013566  100705                            BMI     SECT2D          ;BR=OK
1623 013570  104401  013576                    TYPE    ,67$            ;;TYPE ASCIZ STRING
     013574  000426                            BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  <200>/ERROR10! INTERUPT NOT CAUSED BY TMO. CSR= /
     013652                             66$:
1624 013652  017746  165744                    MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     013656  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1625 013660  000000                            HALT
1626 013662  000776                            BR      .-2
1627 013664  005077  165732            SECT2C: CLR     @CSR            ;CLEAR CSR
1628 013670  005067  164102                    CLR     PSW             ;LOWER PRIORITY
1629
1630                                    ;SECTION 3 : SEND REQUEST EXPECT REJECTION
1631
```

```
1632 013674  012767  000001  005736  SECT3:  MOV     #1,TOCK          ;SET MINIMUM TIME
1633 013702  004767  005672                  JSR     PC,WAIT          ;GO WAIT
1634 013706  012777  014016  165710          MOV     #1$,@VEC0        ;SET RETURN
1635 013714  012777  000101  165700          MOV     #101,@CSR        ;SEND REQUEST
1636 013722  004767  005624                  JSR     PC,WAIT20        ;GO WAIT FOR TIMEOUT
1637 013726  104401  013734                  TYPE    ,65$             ;;TYPE ASCIZ STRING
     013732  000424                          BR      64$              ;;GET OVER THE ASCIZ
                                     ;;65$:  .ASCIZ  <200>/ERROR11! NO INTERUPT DETECTED.  CSR= /
     014004                          64$:
1638 014004  017746  165612                  MOV     @CSR,-(SP)       ;;SAVE @CSR FOR TYPEOUT
     014010  104402                          TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1639 014012  000000                          HALT
1640 014014  000776                          BR      .-2
1641 014016  012706  001100          1$:     MOV     #STACK,SP        ;RESET STACK
1642 014022  005777  165574                  TST     @CSR             ;TIMEOUT?
1643 014026  100437                          BMI     2$               ;BR=YES
1644 014030  104401  014036                  TYPE    ,67$             ;;TYPE ASCIZ STRING
     014034  000427                          BR      66$              ;;GET OVER THE ASCIZ
                                     ;;67$:  .ASCIZ  <200>/ERROR12! INTERUPT NOT CAUSED BY TMO.  CSR= /
     014114                          66$:
1645 014114  017746  165502                  MOV     @CSR,-(SP)       ;;SAVE @CSR FOR TYPEOUT
     014120  104402                          TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1646 014122  000000                          HALT
1647 014124  000776                          BR      .-2
1648 014126  005077  165470          2$:     CLR     @CSR             ;CLEAR CSR
1649 014132  005067  163640                  CLR     PSW              ;LOWER PRIORITY
1650 014136  005767  005422                  TST     TRIG             ;IS TRIGGER SET?
1651 014142  001402                          BEQ     SECT4            ;BR=NO
1652 014144  000167  000644                  JMP     MULEND           ;FINISHED A SUB PASS
1653
1654                                 ;SECTION 4 : SEND REQUEST EXPECT CONNECTION
1655
1656 014150  012777  014260  165446  SECT4:  MOV     #1$,@VEC0        ;SET RETURN
1657 014156  012777  000101  165436          MOV     #101,@CSR        ;SEND REQUEST
1658 014164  004767  005362                  JSR     PC,WAIT20        ;GO WAIT FOR TIMEOUT
1659 014170  104401  014176                  TYPE    ,65$             ;;TYPE ASCIZ STRING
     014174  000424                          BR      64$              ;;GET OVER THE ASCIZ
                                     ;;65$:  .ASCIZ  <200>/ERROR13! NO INTERUPT DETECTED.  CSR= /
     014246                          64$:
1660 014246  017746  165350                  MOV     @CSR,-(SP)       ;;SAVE @CSR FOR TYPEOUT
     014252  104402                          TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1661 014254  000000                          HALT
1662 014256  000776                          BR      .-2
1663 014260  012706  001100          1$:     MOV     #STACK,SP        ;RESET STACK
1664 014264  016767  165462  164632          MOV     KONST,GOOD
1665 014272  062767  020301  164624          ADD     #20301,GOOD
1666 014300  026777  164620  165314          CMP     GOOD,@CSR        ;CONNECTED?
1667 014306  001432                          BEQ     SECT5            ;BR=YES
1668 014310  104401  014316                  TYPE    ,67$             ;;TYPE ASCIZ STRING
     014314  000422                          BR      66$              ;;GET OVER THE ASCIZ
                                     ;;67$:  .ASCIZ  <200>/ERROR14! COULD NOT CONNECT.  CSR= /
     014362                          66$:
1669 014362  017746  165234                  MOV     @CSR,-(SP)       ;;SAVE @CSR FOR TYPEOUT
     014366  104402                          TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1670 014370  000000                          HALT
1671 014372  000776                          BR      .-2
1672
```

```
                                        ;SECTION 5 :DETECT EXTERNAL REQUEST AND REJECT IT

1673
1674
1675 014374  005767  165356            SECT5:  TST     TRIN            ;CAN WE TEST RESET IN NEUTRAL?
1676 014400  001041                            BNE     3$              ;BR=NO
1677 014402  036777  165234  165226            BIT     DEVRC,@DEVAD    ;RESET CONDITION MET
1678 014410  001435                            BEQ     3$              ;BR=YES
1679 014412  104401  014420                    TYPE    ,65$            ;;TYPE ASCIZ STRING
     014416  000427                            BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <200>/ERROR15A! COULDN'T DETECT RESET IN NEUTRAL./
     014476                             64$:
1680 014476  012767  177777  165252            MOV     #-1,TRIN        ;CANCEL FURTHER TESTING OF RESET IN NEUTRAL
1681 014504  012777  014622  165112    3$:     MOV     #1$,@VEC0       ;SET RETURN
1682 014512  005067  163260                    CLR     PSW             ;LOWER PRIORITY
1683 014516  004767  005030                    JSR     PC,WAIT20       ;GO WAIT
1684 014522  104401  014530                    TYPE    ,67$            ;;TYPE ASCIZ STRING
     014526  000430                            BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  <200>/ERROR15! NO EXTERNAL REQUEST DETECTED.  CSR= /
     014610                             66$:
1685 014610  017746  165006                    MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     014614  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1686 014616  000000                            HALT
1687 014620  000776                            BR      .-2
1688 014622  012706  001100            1$:     MOV     #STACK,SP       ;RESET STACK
1689 014626  016767  165120  164270            MOV     KONST,GOOD
1690 014634  062767  030301  164262            ADD     #30301,GOOD
1691 014642  026777  164256  164752            CMP     GOOD,@CSR       ;EXT. INT. ?
1692 014650  001433                            BEQ     2$              ;BR=OK
1693 014652  104401  014660                    TYPE    ,69$            ;;TYPE ASCIZ STRING
     014656  000423                            BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  <200>/ERROR16! UNEXPECTED INTERUPT.  CSR= /
     014726                             68$:
1694 014726  017746  164670                    MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     014732  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1695 014734  000000                            HALT
1696 014736  000776                            BR      .-2
1697 014740  012777  000101  164654    2$:     MOV     #101,@CSR       ;CLEAR EXT. INT.
1698 014746  012777  022600  164650            MOV     #OOPS,@VEC0     ;RESET RETURN
1699 014754  005067  163016                    CLR     PSW             ;LOWER PRIORITY
1700
1701 014760  005367  164742            DT3SP:  DEC     CYCLE           ;COUNT A SUBPASS
1702 014764  001402                            BEQ     1$              ;BR=DONE
1703 014766  000167  175776                    JMP     SECT2           ;CONTINUE
1704 014772  005767  164746            1$:     TST     ONOFF           ;FIRST MASTER OR SLAVE
1705 014776  001401                            BEQ     2$              ;BR=SLAVE
1706 015000  000405                            BR      MULEND          ;MASTER REPORTS ENDPASS
1707 015002  012767  177777  004554    2$:     MOV     #-1,TRIG        ;SET TRIGGER FOR SLAVE
1708 015010  000167  175754                    JMP     SECT2           ;CONTINUE SLAVE AT SECT2
1709
1710 015014  005767  164724            MULEND: TST     ONOFF           ;WAS THIS FIRST SLAVE?
1711 015020  001405                            BEQ     1$              ;BR=YES
1712 015022  012767  012540  004400            MOV     #MULDT3,$RTNAD  ;SET RETURN
1713 015030  000167  004276                    JMP     $EOP+2          ;GO DO ENDPASS
1714 015034  012767  015046  004366    1$:     MOV     #2$,$RTNAD      ;SET RETURN
1715 015042  000167  004264                    JMP     $EOP+2          ;GO DO ENDPASS
1716 015046  004767  004500            2$:     JSR     PC,WAIT20       ;WAIT 2 SECONDS
1717 015052  000167  175462                    JMP     MULDT3          ;GO TO IT!
```

```
 1719
 1720                                        .SBTTL
 1721                                        .SBTTL MANUAL INTERVENTION TEST
 1722
 1723 015056                                MANIN:
 1724 015056  004767  011314                        JSR     PC,CNTLC        ;IS IT CONTROL C?
 1725 015062  005767  164546                        TST     DT03            ;CHECK DT03 MODE
 1726 015066  001402                                BEQ     MANMOD          ;BR=DT07 MODE
 1727 015070  000167  001060                        JMP     PWROK           ;SKIP OVER MANUAL MODE TEST IF DT03
 1728 015074                                MANMOD:
 1729 015074  004767  011276                        JSR     PC,CNTLC        ;IS IT CONTROL C?
 1730 015100  005077  164516                        CLR     @CSR            ;CLEAR CSR
 1731 015104  104401  015112                        TYPE    ,65$            ;;TYPE ASCIZ STRING
      015110  000427                                BR      64$             ;;GET OVER THE ASCIZ
      015170                                ;;65$:  .ASCIZ  <200>/PUT THIS DT07 IN MANUAL MODE THEN TYPE <CR>./
                                            64$:
 1732 015170  104410                                RDCHR
 1733 015172  012600                                MOV     (SP)+,R0        ;;POP STACK INTO R0
 1734 015174  032777  002000  164420        1$:     BIT     #2000,@CSR      ;MANUAL MODE SET?
 1735 015202  001042                                BNE     2$              ;BR=YES
 1736 015204  104401  015212                        TYPE    ,67$            ;;TYPE ASCIZ STRING
      015210  000430                                BR      66$             ;;GET OVER THE ASCIZ
      015272                                ;;67$:  .ASCIZ  <200>/ERROR1!  MANUAL MODE (BIT10) IS NOT SET. CSR= /
                                            66$:
 1737 015272  017746  164324                        MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
      015276  104402                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1738 015300  000000                                HALT
 1739 015302  004767  011070                        JSR     PC,CNTLC        ;IS IT CONTROL C?
 1740 015306  000672                                BR      MANMOD          ;GO TRY AGAIN
 1741 015310  012777  000001  164304        2$:     MOV     #1,@CSR         ;TRY TO SET REQ IN MAN.MOD.
 1742 015316  032777  000001  164276                BIT     #1,@CSR         ;SET?
 1743 015324  001445                                BEQ     3$              ;BR=NO
 1744 015326  104401  015334                        TYPE    ,69$            ;;TYPE ASCIZ STRING
      015332  000433                                BR      68$             ;;GET OVER THE ASCIZ
      015422                                ;;69$:  .ASCIZ  <200>/ERROR2! REQ BIT NOT DISABLED IN MANUAL MODE.  CSR= /
                                            68$:
 1745 015422  017746  164174                        MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
      015426  104402                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1746 015430  000000                                HALT
 1747 015432  004767  010740                        JSR     PC,CNTLC        ;IS IT CONTROL C?
 1748 015436  000724                                BR      2$              ;TRY AGAIN
 1749 015440                                3$:
      015440  104401  015446                        TYPE    ,71$            ;;TYPE ASCIZ STRING
      015444  000426                                BR      70$             ;;GET OVER THE ASCIZ
      015522                                ;;71$:  .ASCIZ  <200>/CONNECT THIS DT07 MANUALY THEN TYPE <CR>./
                                            70$:
 1750 015522  104410                                RDCHR
 1751 015524  012600                                MOV     (SP)+,R0        ;;POP STACK INTO R0
 1752 015526  032777  000001  164066                BIT     #1,@CSR         ;IS REQ SET NOW?
 1753 015534  001041                                BNE     4$              ;BR=YES
 1754 015536  104401  015544                        TYPE    ,73$            ;;TYPE ASCIZ STRING
      015542  000431                                BR      72$             ;;GET OVER THE ASCIZ
      015626                                ;;73$:  .ASCIZ  <200>/ERROR3! REQ. BIT NOT SET WHEN CONNECTED.  CSR= /
                                            72$:
 1755 015626  017746  163770                        MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
      015632  104402                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1756 015634  000000                                HALT
```

```
                                        BR      3$
                                4$:     RESET                   ;RESET THE BUSS
1759 015642  032777  000001  163752     BIT     #1,@CSR         ;REQ STILL THERE?
1760 015650  001034                     BNE     5$              ;BR=YES
1761 015652  104401  015660             TYPE    ,75$            ;;TYPE ASCIZ STRING
     015656  000424                     BR      74$             ;;GET OVER THE ASCIZ
                                ;;75$:  .ASCIZ  <200>/ERROR4! RESET CLEARED REQ. BIT.  CSR= /
     015730                     74$:
1762 015730  017746  163666             MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     015734  104402                     TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1763 015736  000000                     HALT
1764 015740  000737                     BR      4$
1765 015742                     5$:
     015742  104401  015750             TYPE    ,77$            ;;TYPE ASCIZ STRING
     015746  000432                     BR      76$             ;;GET OVER THE ASCIZ
                                ;;77$:  .ASCIZ  <200>/PUT THIS DT07 IN PROGRAMMABLE MODE THEN TYPE <CR>./
     016034                     76$:
1766 016034  104410                     RDCHR
1767 016036  012600                     MOV     (SP)+,R0        ;;POP STACK INTO R0
1768 016040  032777  002000  163554     BIT     #2000,@CSR      ;MAN.MOD. GONE?
1769 016046  001442                     BEQ     PWROK           ;BR=YES
1770 016050  104401  016056             TYPE    ,79$            ;;TYPE ASCIZ STRING
     016054  000432                     BR      78$             ;;GET OVER THE ASCIZ
                                ;;79$:  .ASCIZ  <200>/ERROR5!  MANUAL MODE (BIT10) DIDN'T CLEAR.  CSR= /
     016142                     78$:
1771 016142  017746  163454             MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     016146  104402                     TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1772 016150  000000                     HALT
1773 016152  000673                     BR      5$
1774 016154                     PWROK:
1775 016154  004767  010216             JSR     PC,CNTLC        ;IS IT CONTROL C?
1776 016160  005077  163436             CLR     @CSR            ;CLEAR DEVICE
1777 016164  104401  016172             TYPE    ,65$            ;;TYPE ASCIZ STRING
     016170  000422                     BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>/POWER UP ALL DT07S THEN TYPE <CR>./
     016236                     64$:
1778 016236  104410                     RDCHR
1779 016240  012600                     MOV     (SP)+,R0        ;;POP STACK INTO R0
1780 016242  032777  004000  163352 1$:  BIT     #4000,@CSR      ;PWROK OK?
1781 016250  001034                     BNE     2$              ;BR=YES
1782 016252  104401  016260             TYPE    ,67$            ;;TYPE ASCIZ STRING
     016256  000424                     BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <200>/ERROR6! PWR.OK (BIT11) NOT SET.  CSR= /
     016330                     66$:
1783 016330  017746  163266             MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     016334  104402                     TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1784 016336  000000                     HALT
1785 016340  000705                     BR      PWROK
1786 016342  005067  161430     2$:     CLR     PSW             ;LOWER PRIORITY
1787 016346  004767  003100             JSR     PC,CONNEC       ;GO CONNECT
1788 016352  104401  016360             TYPE    ,69$            ;;TYPE ASCIZ STRING
     016356  000423                     BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <200>/POWER DOWN OTHER CPU THEN TYPE <CR>./
     016426                     68$:
1789 016426  104410                     RDCHR
1790 016430  012600                     MOV     (SP)+,R0        ;;POP STACK INTO R0
1791 016432  032777  004000  163162     BIT     #4000,@CSR      ;PWROK NOT OK?
```

```
 1792 016440  001441                        BEQ     3$              ;BR=OK
 1793 016442  104401  016450                TYPE    ,71$            ;;TYPE ASCIZ STRING
      016446  000427                        BR      70$             ;;GET OVER THE ASCIZ
                                    ;;71$:  .ASCIZ  <200>/ERROR7! PWR. OK (BIT11) NOT CLEARED.  CSR= /
      016526                        70$:
 1794 016526  017746  163070                MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
      016532  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1795 016534  000000                        HALT
 1796 016536  004767  007634                JSR     PC,CNTLC        ;IS IT CONTROL C?
 1797 016542  000677                        BR      2$
 1798 016544  032777  000200  163050  3$:   BIT     #200,@CSR       ;STILL CONNECTED?
 1799 016552  001032                        BNE     4$              ;BR=YES
 1800 016554  104401  016562                TYPE    ,73$            ;;TYPE ASCIZ STRING
      016560  000421                        BR      72$             ;;GET OVER THE ASCIZ
                                    ;;73$:  .ASCIZ  <200>/ERROR8! LOST SHARED BUSS.  CSR= /
      016624                        72$:
 1801 016624  017746  162772                MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
      016630  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1802 016632  000000                        HALT
 1803 016634  000167  177314                JMP     PWROK
 1804 016640                        4$:
      016640  104401  016646                TYPE    ,75$            ;;TYPE ASCIZ STRING
      016644  000433                        BR      74$             ;;GET OVER THE ASCIZ
                                    ;;75$:  .ASCIZ  <200>/RESTORE ALL DT07S TO ORIGINAL STATE THEN TYPE <CR>./
      016734                        74$:
 1805 016734  104410                        RDCHR
 1806 016736  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 1807
 1808 016740                        SWBPWF:
 1809 016740  004767  007432                JSR     PC,CNTLC        ;IS IT CONTROL C?
 1810 016744  005077  162652                CLR     @CSR            ;CLEAR DEVICE
 1811 016750  013746  000024                MOV     @#PWRVEC,-(SP)          ;;PUSH @#PWRVEC ON STACK
 1812 016754  012737  017164  000024        MOV     #11$,@#PWRVEC   ;SET RETURN
 1813 016762  012777  000100  162632        MOV     #100,@CSR       ;SET I.E.
 1814 016770  005067  161002          1$:   CLR     PSW             ;LOWER PRIORITY
 1815 016774  104401  017002                TYPE    ,65$            ;;TYPE ASCIZ STRING
      017000  000426                        BR      64$             ;;GET OVER THE ASCIZ
                                    ;;65$:  .ASCIZ  <200>/POWER DOWN THE SHARED BUSS THEN TYPE <CR>./
      017056                        64$:
 1816 017056  104410                        RDCHR
 1817 017060  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 1818 017062  032777  040000  162532        BIT     #40000,@CSR     ;SWB PWF SET?
 1819 017070  001112                        BNE     2$              ;BR=YES
 1820 017072  104401  017100                TYPE    ,67$            ;;TYPE ASCIZ STRING
      017076  000425                        BR      66$             ;;GET OVER THE ASCIZ
                                    ;;67$:  .ASCIZ  <200>/ERROR8!  SWB PWF (BIT14) NOT SET.  CSR= /
      017152                        66$:
 1821 017152  017746  162464                MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
      017156  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1822 017160  000000                        HALT
 1823 017162  000702                        BR      1$
 1824 017164  012706  001100          11$:  MOV     #STACK,SP       ;RESET STACK
 1825 017170  005746                        TST     -(SP)           ;POSITION STACK
 1826 017172  012637  000024                MOV     (SP)+,@#PWRVEC          ;;POP STACK INTO @#PWRVEC
 1827 017176  104401  017204                TYPE    ,69$            ;;TYPE ASCIZ STRING
      017202  000440                        BR      68$             ;;GET OVER THE ASCIZ
                                    ;;69$:  .ASCIZ  <200>/ERROR9!  TRAP THRU POWER FAIL VECTOR WHEN DISCONNECTED.  CSR= /
```

```
                017304
1828 017304     017746  162312          68$:
     017310     104402                          MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
1829 017312     000000                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1830 017314     000611                          HALT
1831 017316                                     BR      SWBPWF
                017316  104401  017324  2$:
                017322  000423                  TYPE    ,71$            ;;TYPE ASCIZ STRING
                                                BR      70$             ;;GET OVER THE ASCIZ
                                        ;;71$:  .ASCIZ  <200>/POWER UP SHARED BUSS THEN TYPE <CR>/
                017372                  70$:
1832 017372     104410                          RDCHR
1833 017374     012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
1834 017376     032777  040000  162216          BIT     #40000,@CSR     ;DID IT CLEAR?
1835 017404     001441                          BEQ     3$              ;BR=YES
1836 017406     104401  017414                  TYPE    ,73$            ;;TYPE ASCIZ STRING
     017412     000430                          BR      72$             ;;GET OVER THE ASCIZ
                                        ;;73$:  .ASCIZ  <200>/ERROR10!  SWB PWF (BIT14) DIDN'T CLEAR.  CSR= /
                017474                  72$:
1837 017474     017746  162122                  MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     017500     104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1838 017502     000000                          HALT
1839 017504     000167  177260                  JMP     1$
1840            017510                  ERREN=.
1841 017510     005077  162106          3$:     CLR     @CSR    ;CLEAR IT
1842 017514     012737  017746  000024         MOV     #PFVSEV,@#PWRVEC        ;SET RETURNS
1843 017522     012777  020602  162074         MOV     #PWFSER,@VECO
1844 017530     005067  160242                 CLR     PSW
1845 017534     004767  001712                 JSR     PC,CONNEC       ;GO CONNECT
1846 017540     052777  000100  162054         BIS     #100,@CSR       ;SET I.E.
1847 017546     104401  017554                 TYPE    ,75$            ;;TYPE ASCIZ STRING
     017552     000423                         BR      74$             ;;GET OVER THE ASCIZ
                                        ;;75$:  .ASCIZ  <200>/POWER THE SHARED BUSS DOWN THEN UP./
                017622                  74$:
1848 017622     012767  011470  002010         MOV     #11470,TOCK     ;SET TIMER FOR 1 MIN.
1849 017630     004767  001744                 JSR     PC,WAIT
1850 017634     104401  017642                 TYPE    ,77$            ;;TYPE ASCIZ STRING
     017640     000435                         BR      76$             ;;GET OVER THE ASCIZ
                                        ;;77$:  .ASCIZ  <200>/ERROR11!  NO EVIDENCE OF POWER FAILURE DETECTED.  CSR= /
                017734                  76$:
1851 017734     017746  161662                 MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
     017740     104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1852 017742     000000                         HALT
1853 017744     000661                         BR      3$
1854
1855 017746                             PFVSEV:
1856 017746     004767  006424                 JSR     PC,CNTLC        ;IS IT CONTROL C?
1857 017752     017767  161644  161146         MOV     @CSR,BAD        ;SAVE CSR CONTENTS
1858 017760     042767  000400  161140         BIC     #400,BAD        ;MASK BRQ IF PRESENT
1859 017766     042777  000100  161626         BIC     #100,@CSR       ;DROP I.E.
1860 017774     012737  020004  000024         MOV     #1$,@#PWRVEC    ;POINT TO POWER UP ROUTINE
1861 020002     000000                         HALT    ;IF YOU STOPPED HERE NO POWER UP TRAP OCCURED
1862 020004     017767  161612  162662  1$:    MOV     @CSR,TEMP2      ;STORE CSR
1863 020012     016767  161734  161104         MOV     KONST,GOOD      ;ASSEMBLE GOOD DATA
1864 020020     062767  060301  161076         ADD     #60301,GOOD
1865 020026     026767  161072  161072         CMP     GOOD,BAD        ;CORRECT AT PWR DOWN?
1866 020034     001465                         BEQ     2$              ;BR=YES
1867 020036     104401  020044                 TYPE    ,65$            ;;TYPE ASCIZ STRING
```

```
                020042  000433                          BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  <200>/ERROR12!  CSR INCORRECT @ POWER DOWN TIME.  CSR WAS /
                020132
                                                64$:
1868            020132  016746  160770                  MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
                020136  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1869            020140  104401  020146                  TYPE    ,67$            ;;TYPE ASCIZ STRING
                020144  000410                          BR      66$             ;;GET OVER THE ASCIZ
                                                ;;67$:  .ASCIZ  <200>/IT SHOULD BE /
                020166
                                                66$:
1870            020166  016746  160732                  MOV     GOOD,-(SP)      ;;SAVE GOOD FOR TYPEOUT
                020172  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1871            020174  000000                          HALT
1872            020176  012706  001100          11$:    MOV     #STACK,SP       ;RESET STACK
1873            020202  005746                          TST     -(SP)           ;POSITION STACK
1874            020204  000167  177300                  JMP     ERREN
1875            020210  016767  162460  160710  2$:     MOV     TEMP2,BAD       ;GET PWR UP DATA
1876            020216  042767  000400  160702          BIC     #400,BAD        ;MASK BRQ IF PRESENT
1877            020224  162767  020301  160672          SUB     #20301,GOOD     ;CORRECT GOOD DATA
1878            020232  026767  160666  160666          CMP     GOOD,BAD        ;CORRECT @ PWR UP?
1879            020240  001460                          BEQ     3$              ;BR=YES
1880            020242  104401  020250                  TYPE    ,69$            ;;TYPE ASCIZ STRING
                020246  000432                          BR      68$             ;;GET OVER THE ASCIZ
                                                ;;69$:  .ASCIZ  <200>/ERROR13!  CSR INCORRECT @ POWER UP TIME. CSR WAS /
                020334
                                                68$:
1881            020334  016746  160566                  MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
                020340  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1882            020342  104401  020350                  TYPE    ,71$            ;;TYPE ASCIZ STRING
                020346  000410                          BR      70$             ;;GET OVER THE ASCIZ
                                                ;;71$:  .ASCIZ  <200>/IT SHOULD BE /
                020370
                                                70$:
1883            020370  016746  160530                  MOV     GOOD,-(SP)      ;;SAVE GOOD FOR TYPEOUT
                020374  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1884            020376  000000                          HALT
1885            020400  000676                          BR      11$
1886            020402  032777  040000  161212  3$:     BIT     #40000,@CSR     ;DID SWB PWF CLEAR?
1887            020410  001442                          BEQ     4$              ;BR=YES
1888            020412  104401  020420                  TYPE    ,73$            ;;TYPE ASCIZ STRING
                020416  000430                          BR      72$             ;;GET OVER THE ASCIZ
                                                ;;73$:  .ASCIZ  <200>/ERROR14!  SWB PWF (BIT14) DIDN'T CLEAR.  CSR= /
                020500
                                                72$:
1889            020500  017746  161116                  MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
                020504  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1890            020506  000000                          HALT
1891            020510  004767  005662                  JSR     PC,CNTLC        ;IS IT CONTROL C?
1892            020514  000630                          BR      11$
1893            020516                          4$:
                020516  104401  020524                  TYPE    ,75$            ;;TYPE ASCIZ STRING
                020522  000420                          BR      74$             ;;GET OVER THE ASCIZ
                                                ;;75$:  .ASCIZ  <200>/THIS DT07 TRAPS ON POWER FAIL!/
                020564
                                                74$:
1894                    020564                  POSTSV=.
1895            020564  012706  001100                  MOV     #STACK,SP       ;RESET STACK
1896            020570  005746                          TST     -(SP)           ;POSITION STACK
1897            020572  012637  000024                  MOV     (SP)+,@#PWRVEC          ;;POP STACK INTO @#PWRVEC
1898            020576  000167  000454                  JMP     MANEND
1899
1900            020602  017767  161014  160316  PWFSER: MOV     @CSR,BAD        ;SAVE CSR
```

```
 1901 020610  042767  000400  160310          BIC     #400,BAD        ;MASK BRQ IF PRESENT
 1902 020616  016767  161130  160300          MOV     KONST,GOOD      ;ASSEMBLE GOOD DATA
 1903 020624  062767  060301  160272          ADD     #60301,GOOD
 1904 020632  026767  160266  160266          CMP     GOOD,BAD        ;CORRECT @PWR DOWN?
 1905 020640  001457                          BEQ     1$              ;BR=YES
 1906 020642  104401  020650                  TYPE    ,65$            ;;TYPE ASCIZ STRING
      020646  000427                          BR      64$             ;;GET OVER THE ASCIZ
                                     ;;65$:   .ASCIZ  <200>/ERROR15! CSR INCORRECT @ POWER DOWN.  WAS /
      020726                         64$:
 1907 020726  016746  160174                  MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
      020732  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1908 020734  104401  020742                  TYPE    ,67$            ;;TYPE ASCIZ STRING
      020740  000406                          BR      66$             ;;GET OVER THE ASCIZ
                                     ;;67$:   .ASCIZ  <200>/SHOULD BE /
      020756                         66$:
 1909 020756  016746  160142                  MOV     GOOD,-(SP)      ;;SAVE GOOD FOR TYPEOUT
      020762  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1910 020764  000000                          HALT
 1911 020766  012706  001100          11$:    MOV     #STACK,SP       ;RESET STACK
 1912 020772  005746                          TST     -(SP)           ;POSITION STACK
 1913 020774  000167  176510                  JMP     ERREN
 1914 021000  012767  000632  000632  1$:     MOV     #632,TOCK       ;SET TIMER FOR 5SEC.
 1915 021006  004767  000566                  JSR     PC,WAIT ;GO WAIT
 1916 021012  017767  160604  160106          MOV     @CSR,BAD        ;SAVE CSR
 1917 021020  042767  000400  160100          BIC     #400,BAD        ;MASK BRQ IF PRESENT
 1918 021026  162767  060201  160070          SUB     #60201,GOOD     ;CORRECT DATA
 1919 021034  026767  160064  160064          CMP     GOOD,BAD        ;CORRECT NOW?
 1920 021042  001455                          BEQ     2$              ;BR=YES
 1921 021044  104401  021052                  TYPE    ,69$            ;;TYPE ASCIZ STRING
      021050  000431                          BR      68$             ;;GET OVER THE ASCIZ
                                     ;;69$:   .ASCIZ  <200>/ERROR16! CSR INCORRECT AFTER POWER UP. CSR WAS /
      021134                         68$:
 1922 021134  017746  160462                  MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
      021140  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1923 021142  104401  021150                  TYPE    ,71$            ;;TYPE ASCIZ STRING
      021146  000406                          BR      70$             ;;GET OVER THE ASCIZ
                                     ;;71$:   .ASCIZ  <200>/SHOULD BE /
      021164                         70$:
 1924 021164  016746  157734                  MOV     GOOD,-(SP)      ;;SAVE GOOD FOR TYPEOUT
      021170  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1925 021172  000000                          HALT
 1926 021174  000674                          BR      11$
 1927 021176                         2$:
      021176  104401  021204                  TYPE    ,73$            ;;TYPE ASCIZ STRING
      021202  000423                          BR      72$             ;;GET OVER THE ASCIZ
                                     ;;73$:   .ASCIZ  <200>/THIS DT07 INTERRUPTS ON POWER FAIL!/
      021252                         72$:
 1928 021252  000167  177306                  JMP     POSTSV
 1929
 1930 021256                         MANEND:
      021256  104401  021264                  TYPE    ,65$            ;;TYPE ASCIZ STRING
      021262  000420                          BR      64$             ;;GET OVER THE ASCIZ
                                     ;;65$:   .ASCIZ  <200>/MANUAL INTERVENTION TEST DONE./
      021324                         64$:
 1931 021324  000000                          HALT
 1932 021326  000776                          BR      .-2
```

SEQ 0060

```
1934                                    ;       NEWTST  <<INSERT NEXT TEST HERE>>
```

```
 1936                                   .SBTTL  END OF PASS ROUTINE
                                        ;;*****************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO RESTRT
 021330                                 $EOP:
 021330  000004                                 SCOPE
 021332  005067  157544                        CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 021336  005067  157630                        CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 021342  005267  157646                        INC     $PASS           ;;INCREMENT THE PASS NUMBER
 021346  042767  100000  157640              BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 021354  005327                              DEC     (PC)+           ;;LOOP?
 021356  000001                       $EOPCT: .WORD   1
 021360  003022                              BGT     $DOAGN          ;;YES
 021362  012737                              MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 021364  000001                       $ENDCT: .WORD   1
 021366  021356                              $EOPCT
 021370  104401  021435                      TYPE    ,$ENDMG         ;;TYPE "END PASS #"
 021374  016746  157614                      MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 021400  104405                              TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 021402  104401  021432                      TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
 021406  013700  000042               $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 021412  001405                              BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 021414  000005                              RESET                   ;;CLEAR THE WORLD
 021416  004710                       $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 021420  000240                              NOP                     ;;SAVE ROOM
 021422  000240                              NOP                     ;;FOR
 021424  000240                              NOP                     ;;ACT11
 021426                                 $DOAGN:
 021426  000137                              JMP     @(PC)+          ;;RETURN
 021430  002526                       $RTNAD: .WORD   RESTRT
 021432     377     377     000       $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
 021435     015     012     105       $ENDMG: .ASCIZ  <15><12>/END PASS #/
 021440     116     104     040
 021443     120     101     123
 021446     123     040     043
 021451     000
```

```
                                        .SBTTL USER ROUTINES
                
021452  012767  000240  000006  CONNEC: MOV     #240,2$         ;SET SWITCH
021460  105777  160136          1$:     TSTB    @CSR            ;ALL READY CONNECTED?
021464  100410                          BMI     3$              ;BR = YES
021466  000774                  2$:     BR      1$              ;CONTINUE
021470  012777  000001  160124          MOV     #1,@CSR         ;CONNECT
021476  012767  000774  177762          MOV     #774,2$         ;RESTORE SWITCH
021504  000765                          BR      1$              ;BR UNTIL CONNECTED
021506  000207                  3$:     RTS     PC              ;EXIT

021510  012767  000004  160212  CONID:  MOV     #4,COVID        ;CONVERT SLAVE ID
021516  016700  160202                  MOV     CURID,R0        ;SAVE CURRANT ID
021522  001405                  1$:     BEQ     2$              ;BR IF DONE
021524  000241                          CLC
021526  006167  160176                  ROL     COVID           ;SHIFT ID
021532  005300                          DEC     R0              ;COUNT DOWN TO ZERO
021534  000772                          BR      1$
021536  000207                  2$:     RTS     PC              ;EXIT

021540  012767  000122  000072  WAIT10: MOV     #122,TOCK               ;SET CLOCK
021546  000167  000026                  JMP     WAIT
021552  012767  000244  000060  WAIT20: MOV     #244,TOCK               ;SET CLOCK
021560  000167  000014                  JMP     WAIT
021564  000000                  TRIG:   000
021566  012767  000051  000044  WAIT05: MOV     #51,TOCK                ;SET CLOCK
021574  000167  000000                  JMP     WAIT
021600  016767  160142  000030  WAIT:   MOV     TIMA,TICK       ;SET BASE TIMER
021606  005777  160010          1$:     TST     @CSR
021612  000400                          BR      .+2
021614  005367  000016                  DEC     TICK
021620  001372                          BNE     1$
021622  005367  000012                  DEC     TOCK
021626  001364                          BNE     WAIT
021630  005267  000004                  INC     TOCK
021634  000207                          RTS     PC
021636  000000                  TICK:   0
021640  000000                  TOCK:   0
                
                
                
                
021642  005000                  SYNCUP: CLR     R0              ;SET DELAY TIME
021644  012701  000400                  MOV     #400,R1
021650  032777  020000  157744  1$:     BIT     #20000,@CSR             ;BUSS ACTIVE?
021656  001434                          BEQ     2$              ;BR=NO
021660  005200                          INC     R0              ;TIME OUT
021662  001372                          BNE     1$
021664  005301                          DEC     R1
021666  001370                          BNE     1$
021670  104401  021676                  TYPE    ,65$            ;;TYPE ASCIZ STRING
021674  000420                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>/SYNCRONIZATION FAILURE!  CSR= /
021736                          64$:
021736  017746  157660                  MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
021742  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
 1991 021744  000000                        HALT
 1992 021746  000776                        BR      .-2
 1993 021750  005000                2$:     CLR     R0              ;RESET DELAY TIME
 1994 021752  012701  000000               MOV     #0,R1
 1995 021756  032777  020000  157636  3$:  BIT     #20000,@CSR     ;BUSS ACTIVE?
 1996 021764  001043                        BNE     4$              ;BR=YES
 1997 021766  005200                        INC     R0              ;TIME OUT
 1998 021770  001372                        BNE     3$
 1999 021772  005301                        DEC     R1
 2000 021774  001370                        BNE     3$
 2001 021776  104401  022004               TYPE    ,67$            ;;TYPE ASCIZ STRING
      022002  000427                        BR      66$             ;;GET OVER THE ASCIZ
                                    ;;67$:  .ASCIZ  <200>/MASTER PORT DIDN'T TAKE SHARED BUSS!  CSR= /
      022062                        66$:
 2002 022062  017746  157534               MOV     @CSR,-(SP)      ;;SAVE @CSR FOR TYPEOUT
      022066  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 2003 022070  000000                        HALT
 2004 022072  000776                        BR      .-2
 2005 022074  000207                4$:     RTS     PC              ;EXIT
 2006
 2007
 2008
 2009                                       ;CONSTRUCT SEQUENCER
 2010 022076  012701  001644        SEQMAK: MOV     #PORTNO,R1      ;STORE ADDRESS OF DATA
 2011 022102  012702  001732                MOV     #TEMP1,R2       ;SET TEMPORARY STORAGE POINTER
 2012 022106  012703  001660                MOV     #SEQ,R3         ;LOAD FIRST SEQUENCER ADDRESS
 2013 022112  005067  157540                CLR     EXPTCT          ;CLEAR PORT COUNT
 2014 022116  012122                1$:     MOV     (R1)+,(R2)+     ;FILL TEMP
 2015 022120  005267  157532                INC     EXPTCT          ;COUNT THEM
 2016 022124  005711                        TST     @R1             ;LAST ONE
 2017 022126  100373                        BPL     1$              ;BR=NO
 2018 022130  012712  177777               MOV     #-1,@R2         ;TERMINATE DATA
 2019 022134  005001                        CLR     R1              ;LOAD DETECTOR
 2020 022136  012702  001732        2$:     MOV     #TEMP1,R2       ;RESET DATA POINTER
 2021 022142  020112                3$:     CMP     R1,(R2)         ;ID FOUND?
 2022 022144  001407                        BEQ     5$              ;BR=YES
 2023 022146  005722                4$:     TST     (R2)+           ;STEP AND TEST FOR LAST
 2024 022150  100374                        BPL     3$              ;BR IF NOT DONE
 2025 022152  005201                        INC     R1              ;STEP DETECTOR
 2026 022154  022701  000004               CMP     #4,R1           ;CHECK FOR DONE
 2027 022160  001366                        BNE     2$              ;BR IF NOT DONE
 2028 022162  000404                        BR      6$              ;FINISH UP
 2029 022164  011223                5$:     MOV     @R2,(R3)+       ;LOAD SEQUENCER
 2030 022166  012712  000400               MOV     #400,@R2        ;ERASE ENTRY
 2031 022172  000765                        BR      4$              ;CONTINUE
 2032 022174  005367  157456        6$:     DEC     EXPTCT          ;CORRECT EXTERNAL PORT COUNT
 2033 022200  012713  177777               MOV     #-1,@R3         ;TERMINATE SEQUENCER
 2034 022204  000207                        RTS     PC              ;EXIT
 2035
 2036
 2037
 2038                                       ;ROUTINE TO ASSEMBLE MENUS
 2039 022206  012767  177777  157530  MASMEN: MOV     #-1,ONOFF       ;SET MASTER/SLAVE INDICATOR
 2040 022214  004767  000462                JSR     PC,FLUSHM       ;GO FLUSH MENU
 2041 022220  012701  001676                MOV     #MENU,R1        ;SET POINTER
 2042 022224  117721  157444        1$:     MOVB    @NEXENT,(R1)+   ;LOAD MENU WITH SLAVE ID
 2043 022230  112721  000001                MOVB    #1,(R1)+        ;LOAD FUNCTION #1
```

```
2044 022234  004767  000302                JSR     PC,STPNEX       ;GET NEXT SLAVE ID
2045 022240  000771                        BR      1$              ;CONTINUE
2046 022242  004767  000274          2$:   JSR     PC,STPNEX       ;GO GET NEXT ID
2047 022246  000405                        BR      3$              ;BR=NOT FINISHED
2048 022250  012711  177777                MOV     #-1,@R1         ;TERMINATE MENU
2049 022254  012704  001676                MOV     #MENU,R4        ;STORE MENUS
2050 022260  000207                        RTS     PC              ;EXIT
2051 022262  117721  157406          3$:   MOVB    @NEXENT,(R1)+
2052 022266  112721  000002                MOVB    #2,(R1)+        ;LOAD FUNCTION #2
2053 022272  000763                        BR      2$              ;CONTINUE
2054
2055
2056 022274  005067  157444          SALMEN: CLR   ONOFF           ;CLR MASTER/SLAVE INDICATOR
2057 022300  004767  000376                JSR     PC,FLUSHM       ;GO FLUSH MENU
2058 022304  012701  001676                MOV     #MENU,R1        ;SET POINTER
2059 022310  027767  157360  157326  1$:   CMP     @NEXENT,PORTNO  ;IS THIS ME?
2060 022316  001010                        BNE     3$              ;BR=NO
2061 022320  117721  157350                MOVB    @NEXENT,(R1)+   ;LOAD SLAVE ID
2062 022324  112721  000003                MOVB    #3,(R1)+        ;LOAD FUNCTION 3
2063 022330  004767  000206          2$:   JSR     PC,STPNEX       ;GO GET NEXT ID
2064 022334  000765                        BR      1$              ;BR=NOT DONE
2065 022336  000405                        BR      4$              ;GO TO NEXT SET
2066 022340  117721  157330          3$:   MOVB    @NEXENT,(R1)+
2067 022344  112721  000005                MOVB    #5,(R1)+        ;LOAD FUNCTION 5
2068 022350  000767                        BR      2$              ;CONTINUE
2069 022352  004767  000164          4$:   JSR     PC,STPNEX       ;GO GET NEXT ID
2070 022356  000405                        BR      5$              ;GO CONTINUE
2071 022360  012711  177777                MOV     #-1,@R1         ;TERMINATE MENU
2072 022364  012704  001676                MOV     #MENU,R4        ;STORE MENUS
2073 022370  000207                        RTS     PC              ;EXIT
2074 022372  027767  157276  157244  5$:   CMP     @NEXENT,PORTNO  ;IS THIS ME?
2075 022400  001014                        BNE     7$              ;BR=NO
2076 022402  117721  157266                MOVB    @NEXENT,(R1)+
2077 022406  112721  000004                MOVB    #4,(R1)+        ;LOAD FUNCTION 4
2078 022412  004767  000124          6$:   JSR     PC,STPNEX       ;GO GET NEXT ID
2079 022416  000765                        BR      5$              ;CONTINUE
2080 022420  012711  177777                MOV     #-1,@R1         ;TERMINATE MENU
2081 022424  012704  001676                MOV     #MENU,R4        ;STORE MENUS
2082 022430  000207                        RTS     PC              ;EXIT
2083 022432  117721  157236          7$:   MOVB    @NEXENT,(R1)+
2084 022436  112721  000005                MOVB    #5,(R1)+        ;LOAD FUNCTION 5
2085 022442  000763                        BR      6$              ;CONTINUE
2086
2087
2088                                ;ROUTINE TO INIT SEQUENCER
2089
2090 022444  012767  001660  157220  SEQINT: MOV     #SEQ,SEQPTR     ;SET POINTER TO FIRST ENTRY
2091 022452  012767  001662  157214          MOV     #SEQ+2,NEXENT   ;GET NEXT ENTRY
2092 022460  017705  157206                MOV     @SEQPTR,R5      ;LOAD R5
2093 022464  000207                        RTS     PC              ;EXIT
2094 022466  062767  000002  157176  SEQSTP: ADD     #2,SEQPTR       ;STEP SEQUENCER
2095 022474  005777  157172                TST     @SEQPTR         ;CHECK FOR END
2096 022500  100761                        BMI     SEQINT          ;RELOAD IF DONE
2097 022502  017705  157164                MOV     @SEQPTR,R5
2098 022506  016767  157160  157160        MOV     SEQPTR,NEXENT   ;STEP NEXT ENTRY
2099 022514  062767  000002  157152        ADD     #2,NEXENT
2100 022522  005777  157146                TST     @NEXENT         ;OFF THE END?
```

```
2101 022526  100401                    BMI     2$              ;BR=YES
2102 022530  000207              1$:    RTS     PC              ;NO! EXIT
2103 022532  012767  001660  157134  2$:    MOV     #SEQ,NEXENT     ;CORRECT POINTER
2104 022540  000773                    BR      1$              ;EXIT
2105 022542  062767  000002  157124  STPNEX: ADD     #2,NEXENT       ;STEP NEXT ENTRY
2106 022550  005777  157120            TST     @NEXENT         ;OFF THE END?
2107 022554  100003                    BPL     1$              ;BR=NO
2108 022556  012767  001660  157110    MOV     #SEQ,NEXENT     ;RESET
2109 022564  027705  157104      1$:   CMP     @NEXENT,R5      ;BACK TO CURRENT DT07?
2110 022570  001002                    BNE     2$              ;BR=NO
2111 022572  062716  000002            ADD     #2,@SP          ;SET UP FINISHED RETURN
2112 022576  000207              2$:   RTS     PC              ;EXIT
2113
2114
2115 022600  011667  156322      OOPS: MOV     @SP,BAD         ;SAVE ERROR PC
2116 022604  104401  022612            TYPE    ,65$            ;;TYPE ASCIZ STRING
     022610  000427                    BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$: .ASCIZ  <200>/UNEXPECTED INTERUPT FROM DT07 AT ADDRESS.../
     022670                     64$:
2117 022670  016746  156232            MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
     022674  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2118 022676  000000                    HALT
2119 022700  000776                    BR      .-2
2120
2121 022702  012700  001676  FLUSHM: MOV     #MENU,R0        ;SET POINTER
2122 022706  012701  000013            MOV     #11.,R1         ;SET FLUSH COUNT
2123 022712  012720  177777      1$:   MOV     #-1,(R0)+       ;FLUSH WITH -1
2124 022716  005301                    DEC     R1              ;COUNT
2125 022720  001374                    BNE     1$              ;CONT.
2126 022722  000207                    RTS     PC              ;EXIT
2127 022724  012700  001634  FLUSHP: MOV     #DT03,R0        ;LOAD POINTER
2128 022730  012701  000017            MOV     #15.,R1         ;SET COUNT
2129 022734  012720  177777            MOV     #-1,(R0)+       ;FLUSH
2130 022740  005020                    CLR     (R0)+
2131 022742  012720  177777      1$:   MOV     #-1,(R0)+       ;FLUSH WITH -1
2132 022746  005301                    DEC     R1              ;COUNT
2133 022750  001374                    BNE     1$              ;CONT
2134 022752  000207                    RTS     PC              ;EXIT
2135
2136
2137 022754              DMODE:
2138 022754  012767  000340  155014  3$:    MOV     #340,PSW        ;RISE PRIORITY
2139 022762  012777  000001  156632        MOV     #1,@CSR         ;CONNECT
2140 022770  017767  156626  156130  4$:    MOV     @CSR,BAD        ;SAVE CSR
2141 022776  032767  000400  156122        BIT     #400,BAD        ;LOOK FOR BRQ
2142 023004  001010                    BNE     5$              ;BR=FOUND
2143 023006  032767  100000  156112        BIT     #100000,BAD     ;DID WE TIME OUT?
2144 023014  001765                    BEQ     4$              ;BR=NO
2145 023016  012767  177777  156610        MOV     #-1,DT03        ;INDIACATE DT03 MODE
2146 023024  000531                    BR      8$              ;EXIT
2147 023026  005067  156602      5$:   CLR     DT03            ;INDIACATE DT07 MODE
2148 023032  016700  156070            MOV     BAD,R0          ;SAVE IMAGE
2149 023036  042700  177703            BIC     #177703,R0      ;SAVE ONLY RQ0-3
2150 023042  000241                    CLC                     ;CONVERT RQ BIT
2151 023044  006000                    ROR     R0
2152 023046  006000                    ROR     R0
2153 023050  005002                    CLR     R2
```

```
                                                6$:     BIT     #1,R0           ;BIT SET?
                                                        BNE     7$              ;BR=YES
                                                        INC     R2              ;STEP INDIACATOR
                                                        CLC
                                                        ROR     R0
                                                        CMP     #4,R2           ;TOO HIGH?
                                                        BHI     6$              ;BR =NO
                                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                        BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  <200>/ERROR!  BRQ DETECTED WITHOUT ANY RQ. BIT SET.  CSR= /
                                                64$:
                                                        MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
                                                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                                        HALT                    ;FATAL ERROR BRQ SEEN BUT NO RQ
                                                        BR      .-2
                                                7$:     MOV     R2,PORTNO       ;LOAD PORT ID#
                                                        CLC
                                                        ROR     R0              ;MAKE SURE ONLY ONE RQ IS PRESENT
                                                        BEQ     8$              ;BR=OK
                                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                                        BR      66$             ;;GET OVER THE ASCIZ
                                                ;;67$:  .ASCIZ  <200>/ERROR!  MORE THAN ONE RQ. BIT SET.  CSR= /
                                                66$:
                                                        MOV     BAD,-(SP)       ;;SAVE BAD FOR TYPEOUT
                                                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                                        HALT
                                                        BR      .-2
                                                8$:     CLR     @CSR            ;CLEAR AND DISCONNECT AND EXIT
                                                        RTS     PC

                                                DTIME:  MOV     #340,PSW        ;LOCK OUT INTERUPTS
                                                        CLR     TIMA            ;CLEAR TIMERS
                                                        CLR     @CSR            ;CLEAR DEVICE
                                                1$:     TSTB    @CSR            ;MAKE SURE IT'S CLEAR
                                                        BMI     1$
                                                        INC     @CSR            ;SET REQ
                                                2$:     TST     @CSR            ;TMO SET?
                                                        BMI     3$              ;BR=YES
                                                        INC     TIMA            ;COUNT
                                                        BNE     2$
                                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                        BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  <200>/ERROR! TIMING TOO LONG!/
                                                64$:
                                                        HALT
                                                3$:     CLR     PSW             ;LOWER PRIORITY
                                                        MOV     #1,@CSR         ;CONNECT
                                                4$:     TSTB    @CSR            ;CONNECTED?
                                                        BPL     4$
                                                        CLR     TIMB            ;CLAER WATCHDOG TIMER
                                                        MOV     #340,PSW        ;RISE PRIORITY
                                                        DEC     @CSR            ;DROP REQ
                                                5$:     TST     @CSR            ;LOOK FOR TMO
                                                        BMI     6$              ;BR=FOUND
                                                        INC     TIMB
                                                        BNE     5$
```

```
                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                        BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <200>/ERROR! WATCHDOG TIMING TOO LONG!/
                                66$:
                                        HALT
                                6$:     MOV     TIMA,R0         ;COMPARE TIMING FOR PROPER RATIO
                                        MOV     TIMB,R1
                                        CLR     R2
                                        MOV     #25.,R3
                                7$:     ADD     R0,R2           ;MULTIPLY TIMA BY 25
                                        DEC     R3
                                        BNE     7$              ;BR=NOT DONE
                                8$:     SUB     R1,R2           ;DIVIDE TIMA BY TIMB
                                        INC     R3
                                        CMP     R1,R2           ;DONE?
                                        BLOS    8$              ;BR=NO
                                        CMP     R3,#33          ;TOO LOW?
                                        BHIS    9$              ;BR=NO
                                        TYPE    ,69$            ;;TYPE ASCIZ STRING
                                        BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <200>"ERROR! WATCHDOG/REQUEST TIMER RATIO TOO LOW!"
                                68$:
                                        HALT
                                        BR      .-2
                                9$:     CMP     R3,#44          ;TOO HIGH?
                                        BLOS    11$             ;BR=NO
                                        TYPE    ,71$            ;;TYPE ASCIZ STRING
                                        BR      70$             ;;GET OVER THE ASCIZ
                                ;;71$:  .ASCIZ  <200>"ERROR! WATCHDOG/REQUEST TIMER RATIO TOO HIGH!"
                                70$:
                                        HALT
                                        BR      .-2
                                11$:    CLR     @CSR            ;CLEAR AND DISCONNECT
                                        RTS     PC


                                        ;PROGRAM PARAMETER INPUT ROUTINE

                                GETADR: MOV     #-1,PARFLG      ;SET PARFLG
                                        CLR     USEPAR          ;CLEAR FLAG
                                        JSR     PC,FLUSHP       ;GO FLUSH OUT PARAMETER BLOCK
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>/DO YOU WANT TO CHANGE THE DEVICE ADDRESSES?(Y OR N)/
                                64$:
                                        RDCHR
                                        MOV     (SP)+,R0        ;STORE CHAR.
                                        MOVB    R0,ECHOB
                                        TYPE    ,ECHOB
                                        CMP     #'Y,R0          ;YES?
                                        BNE     10$             ;BR=NO
                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                        BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <200>/IN THE FOLLOWING THREE PRINTOUTS TYPE THE NEW DATA OR 0 IF NO CHANGE./
                                66$:
                                        TYPE    ,69$            ;;TYPE ASCIZ STRING
                                        BR      68$             ;;GET OVER THE ASCIZ
```

SEQ 0068

```
                024312
2241            024312  016746  155304          ;;69$:  .ASCIZ  <200>/DEVICE ADDRESS= /
                024316  104402          68$:
2242            024320  104401  034425                  MOV     CSR,-(SP)       ;;SAVE CSR FOR TYPEOUT
2243            024324  104412                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2244            024326  012600                          TYPE    ,SPACE
2245            024330  001402                          RDOCT
2246            024332  010067  154724                  MOV     (SP)+,R0        ;STORE INFO
2247            024336                                  BEQ     11$
                024336  104401  024344                  MOV     R0,$BASE
                024342  000411          11$:
                                                        TYPE    ,71$            ;;TYPE ASCIZ STRING
                024366                                  BR      70$             ;;GET OVER THE ASCIZ
2248            024366  016700  154664          ;;71$:  .ASCIZ  <200>/DEVICE VECTOR= /
2249            024372  042700  177000          70$:
2250            024376  010046                          MOV     $VECT1,R0
                024400  104402                          BIC     #177000,R0
2251            024402  104401  034425                  MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
2252            024406  104412                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2253            024410  012600                          TYPE    ,SPACE
2254            024412  001402                          RDOCT
2255            024414  010001                          MOV     (SP)+,R0
2256            024416  000404                          BEQ     12$
2257            024420  016701  154632                  MOV     R0,R1           ;SAVE IT TEMPORARILY
2258            024424  042701  177000                  BR      14$             ;JUMP OVER NEXT TWO LINES
2259            024430                          12$:    MOV     $VECT1,R1       ;SAVE OLD VECTOR ADDRESS
                024430  104401  024436                  BIC     #177000,R1      ;MASK OUT JUNK
                024434  000410          14$:
                                                        TYPE    ,73$            ;;TYPE ASCIZ STRING
                024456                                  BR      72$             ;;GET OVER THE ASCIZ
2260            024456  016700  154574          ;;73$:  .ASCIZ  <200>/BREAK LEVEL= /
2261            024462  042700  000777          72$:
2262            024466  000300                          MOV     $VECT1,R0
2263            024470  010046                          BIC     #777,R0
                024472  104402                          SWAB    R0
2264            024474  104401  034425                  MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
2265            024500  104412                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2266            024502  012600                          TYPE    ,SPACE
2267            024504  001405                          RDOCT
2268            024506  000300                          MOV     (SP)+,R0
2269            024510  060100                          BEQ     13$
2270            024512  010067  154540                  SWAB    R0
2271            024516  000405                          ADD     R1,R0
2272            024520  042767  000777  154530  13$:    MOV     R0,$VECT1
2273            024526  060167  154524                  BR      10$
2274            024532                          13$:    BIC     #777,$VECT1
2275            024532  000207                          ADD     R1,$VECT1
2276            024534                          10$:
2277            024534  004767  177250                  RTS     PC              ;RETURN
2278            024540  012767  024552  156124  PARIN:
2279                                                    JSR     PC,GETADR
2280            024546  000167  155210                  MOV     #REPAR,ROUTE    ;SET RETURN
2281                    024552
2282            024552  004767  176176                  JMP     START
2283            024556  004767  176534          REPAR=.
2284            024562  005767  155046                  JSR     PC,DMODE        ;GO GET MODE AND ID#
                                                        JSR     PC,DTIME        ;GO GET DEVICE TIMING
                                                        TST     DT03            ;WHAT MODE?
```

```
   2285 024566  001422                        BEQ     20$             ;BR=DT07
   2286 024570  104401  024576                TYPE    ,65$            ;;TYPE ASCIZ STRING
        024574  000416                        BR      64$             ;;GET OVER THE ASCIZ
                                     ;;65$:  .ASCIZ  <200>/THIS DT07 IS IN DT03 MODE./
        024632                       64$:
   2287 024632  000421                        BR      1$              ;CONTINUE AT 1$
   2288 024634                       20$:
        024634  104401  024642                TYPE    ,67$            ;;TYPE ASCIZ STRING
        024640  000413                        BR      66$             ;;GET OVER THE ASCIZ
                                     ;;67$:  .ASCIZ  <200>/THIS DT07 IS PORT# /
        024670                       66$:
   2289 024670  016746  154750                MOV     PORTNO,-(SP)    ;;SAVE PORTNO FOR TYPEOUT
        024674  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
   2290 024676                       1$:
        024676  104401  024704                TYPE    ,69$            ;;TYPE ASCIZ STRING
        024702  000447                        BR      68$             ;;GET OVER THE ASCIZ
                                     ;;69$:  .ASCIZ  <200>/TYPE THE ADDRESS OF AN EXTERNAL DEVICE REGISTER ON THE SHARED BUSS./<2
00>/ 0=NONE /
        025022                       68$:
   2291 025022  104412                        RDOCT
   2292 025024  012667  154606                MOV     (SP)+,DEVAD     ;STORE IN DEVAD
   2293 025030  001523                        BEQ     2$              ;SKIP OVER NEXT TWO QUESTIONS IF NO DEVICE
   2294 025032  104401  025040                TYPE    ,71$            ;;TYPE ASCIZ STRING
        025036  000442                        BR      70$             ;;GET OVER THE ASCIZ
                                     ;;71$:  .ASCIZ  <200> % TYPE AN OCTAL MASK OF READ/WRITE BITS IN ABOVE DEVICE REGISTER..%<20
0>
        025144                       70$:
   2295 025144  104412                        RDOCT
   2296 025146  012667  154466                MOV     (SP)+,DEVRW     ;STORE BIT PATTERN
   2297 025152  104401  025160                TYPE    ,73$            ;;TYPE ASCIZ STRING
        025156  000445                        BR      72$             ;;GET OVER THE ASCIZ
                                     ;;73$:  .ASCIZ  <200> % TYPE AN OCTAL MASK OF RESET-CLEARABLE BITS IN ABOVE DEVICE REGISTER.
..%<200>
        025272                       72$:
   2298 025272  104412                        RDOCT
   2299 025274  012667  154342                MOV     (SP)+,DEVRC     ;STORE BIT PATTERN
   2300 025300  005767  154330       2$:     TST     DT03            ;NO NEED TO CONTINUE IF DT03 MODE
   2301 025304  001067                        BNE     7$              ;BR=DT03
   2302 025306                       3$:4$:
        025306  104401  025314                TYPE    ,75$            ;;TYPE ASCIZ STRING
        025312  000427                        BR      74$             ;;GET OVER THE ASCIZ
                                     ;;75$:  .ASCIZ  <200>/HOW MANY OTHER PORTS ARE THERE TO BE TESTED?/
        025372                       74$:
   2303 025372  104412                        RDOCT
   2304 025374  012667  154256                MOV     (SP)+,EXPTCT    ;STORE EXTERNAL PORT COUNT
   2305 025400  001431                        BEQ     7$              ;GO TO 7$
   2306 025402  012700  001646       5$:     MOV     #EXPRT1,R0      ;SET POINTER
   2307 025406  016701  154244                MOV     EXPTCT,R1
   2308 025412                       6$:
        025412  104401  025420                TYPE    ,77$            ;;TYPE ASCIZ STRING
        025416  000416                        BR      76$             ;;GET OVER THE ASCIZ
                                     ;;77$:  .ASCIZ  <200>/TYPE IN EXTERNAL PORT ID../
        025454                       76$:
   2309 025454  104412                        RDOCT
   2310 025456  012620                        MOV     (SP)+,(R0)+     ;STORE IT
   2311 025460  005301                        DEC     R1              ;COUNT IT
   2312 025462  001353                        BNE     6$              ;CONTINUE LOAD IF NOT DONE
   2313 025464                       7$:
        025464  104401  025472                TYPE    ,79$            ;;TYPE ASCIZ STRING
        025470  000422                        BR      78$             ;;GET OVER THE ASCIZ
```

SEQ 0070

```
                025536
     2314 025536  104412
     2315 025540  012600
     2316 025542  001003
     2317 025544  104401  034427
     2318 025550  000745
     2319 025552  022700  000004
     2320 025556  001002
     2321 025560  000167  000644
     2322 025564  022700  000004
     2323 025570  101016
     2324 025572  104401  025600
          025576  000412

                025624
     2325 025624  000717
     2326 025626  022700  000002
     2327 025632  001044
     2328 025634  005767  154016
     2329 025640  001041
     2330 025642  005767  153766
     2331 025646  001036
     2332 025650  104401  025656
          025654  000432

                025742
     2333 025742  000650
     2334 025744  022700  000002
     2335 025750  001057
     2336 025752  104401  025760
          025756  000454

/ THE MULTIPORT TEST!/
                026110
     2337 026110  000241
     2338 026112  006100
     2339 026114  016016  026212
     2340 026120  104401  026126
          026124  000424

                026176
     2341 026176  104410
     2342 026200  012600
     2343 026202  000207
     2344 026204  000000
     2345 026206     000     200     000
     2346
     2347 026212  000000
     2348 026214  002566
     2349 026216  002600
     2350 026220  002636
     2351
     2352
     2353 026222
          026222  012600
     2354 026224  104401  026232
          026230  000422
```

```
;;79$:  .ASCIZ  <200>/SELECT TEST. (0 FOR HELP MESSAGE)/
78$:
        RDOCT
        MOV     (SP)+,R0
        BNE     8$
        TYPE    ,SELMS
        BR      7$
8$:     CMP     #4,R0   ;TEST FOR NPR?
        BNE     18$     ;NO:BR
        JMP     TSTNPR  ;GO CHECK FOR NPR LINE
18$:    CMP     #4,R0   ;VALID TEST?
        BHI     9$
        TYPE    ,81$            ;;TYPE ASCIZ STRING
        BR      80$             ;;GET OVER THE ASCIZ
;;81$:  .ASCIZ  <200>/INVALID SELECTION!/
80$:
        BR      7$
9$:     CMP     #2,R0           ;MULTIPORT?
        BNE     21$             ;BR=NO
        TST     EXPTCT          ;ONLY ONE PORT?
        BNE     21$             ;BR=NO
        TST     DT03            ;DT03 MODE?
        BNE     21$             ;BR=YES
        TYPE    ,83$            ;;TYPE ASCIZ STRING
        BR      82$             ;;GET OVER THE ASCIZ
;;83$:  .ASCIZ  <200>/CAN NOT RUN THE MULTIPORT TEST WITH ONLY ONE PORT!/
82$:
        BR      7$
21$:    CMP     #2,R0           ;MULTIPORT TEST?
        BNE     22$             ;BR=NO
        TYPE    ,85$            ;;TYPE ASCIZ STRING
        BR      84$             ;;GET OVER THE ASCIZ
;;85$:  .ASCIZ  <200>/THE DT07 WITH THE LOWEST PORT# MUST BE STARTED LAST WHEN RUNNING/<200>
84$:
22$:    CLC
        ROL     R0
        MOV     TX(R0),(SP)
        TYPE    ,87$            ;;TYPE ASCIZ STRING
        BR      86$             ;;GET OVER THE ASCIZ
;;87$:  .ASCIZ  <200>/TYPE <CR> WHEN READY TO CONTINUE TEST./
86$:
        RDCHR
        MOV     (SP)+,R0
        RTS     PC              ;EXIT
PARFLG: 000                     ;PARAMETERS PRESENT FLAG
ECHOB:  .BYTE   0,200,0         ;ECHO BYTE
        .EVEN
TX:     0
        ROUT1
        ROUT2
        ROUT4

TRAP4:
        MOV     (SP)+,R0        ;;POP STACK INTO R0
        TYPE    ,65$            ;;TYPE ASCIZ STRING
        BR      64$             ;;GET OVER THE ASCIZ
```

SEQ 0071

```
                                        ;;65$:  .ASCIZ  <200>/UNEXPECTED TRAP TO VECTOR 4 FROM../
     026276                             64$:
2355 026276  010046                             MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
     026300  104402                             TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2356 026302  000000                             HALT
2357 026304  000167  153452                     JMP     START
2358
2359 026310                             ERROUT:
2360 026310  016637  000004  000004             MOV     4(SP),@#ERRVEC  ;ADJUST STACK
2361 026316  005267  000014                     INC     FLAG
2362 026322  016666  000002  000004             MOV     2(SP),4(SP)     ;PS
2363 026330  011666  000002                     MOV     (SP),2(SP)      ;PC
2364 026334  000002                             RTI
2365
2366 026336  000000                     FLAG:   0
2367
2368 026340  010246                     CKADR:  MOV     R2,-(SP)        ;SAVE R2
2369 026342  016702  153270                     MOV     DEVAD,R2        ;DEVICE ADDRESS IN R2
2370 026346  013746  000004                     MOV     @#ERRVEC,-(SP)  ;SAVE OLD TRAP ADDRESS
2371 026352  012737  026310  000004             MOV     #ERROUT,@#ERRVEC        ;NEW TRAP ADDRESS
2372 026360  005067  177752                     CLR     FLAG            ;INITIALIZE FLAG
2373 026364  005712                             TST     (R2)            ;IS ADDRESS ADDRESSABLE?
2374 026366  000240                             NOP
2375 026370  005726                             TST     (SP)+           ;RESTORE OLD TRAP ADDRESS
2376 026372  010226                             MOV     R2,(SP)+        ;RESTORE R2
2377 026374  000207                             RTS     PC
2378
2379 026376                             CNTLC:
2380 026376  105777  152542                     TSTB    @$TKS           ;IS CHAR THERE?
2381 026402  100011                             BPL     1$              ;NO RETURN
2382 026404  117746  152536                     MOVB    @$TKB,-(SP)     ;PUT CHAR ON STACK
2383 026410  042716  177600                     BIC     #^C177,(SP)     ;STRIP OFF UNWANTED BITS
2384 026414  022726  000003                     CMP     #3,(SP)+        ;IS IT CONTROL C?
2385 026420  001002                             BNE     1$              ;NO RETURN
2386 026422  000167  154164                     JMP     ROUT3           ;GO TO START AGAIN
2387 026426  000207                     1$:     RTS     PC
2388
```

```
2390                                    ;
2391                                    ;CHECK OUT NPR LINE ON THE SWITCH BUS
2392                                    ;
2393 026430                            TSTNPR:
2394 026430  005067  000526                    CLR     PASS            ;CLEAR PASS COUNT
2395 026434  005067  000526                    CLR     NUM             ;CLEAR NUM
2396 026440  104401  026446                    TYPE    ,65$            ;;TYPE ASCIZ STRING
     026444  000417                            BR      64$             ;;GET OVER THE ASCIZ
     026504                             ;;65$: .ASCIZ  <200>/TEST NPR LINE ON SWITCH BUS/
                                        64$:
2397 026504  104401  026512                    TYPE    ,67$            ;;TYPE ASCIZ STRING
     026510  000415                            BR      66$             ;;GET OVER THE ASCIZ
     026544                             ;;67$: .ASCIZ  <200>/TYPE CONTROL C TO EXIT/<200>
                                        66$:
2398 026544                            NPR:
2399 026544  004767  177626                    JSR     PC,CNTLC        ;IS IT CONTROL C?
2400 026550  005267  000412                    INC     NUM             ;COUNT NUMBER OF ITERATION
2401 026554  012767  000003  000402            MOV     #3,BRCNT        ;COUNTER FOR BR LEVEL
2402 026562  012702  027236                    MOV     #BRLV,R2        ;POINTER TO BRLEVEL TABLE
2403 026566  012706  001100                    MOV     #STACK,SP       ;RESTORE STACK
2404 026572  012767  000000  151176            MOV     #0,PSW          ;DROP PRIORITY
2405 026600  012777  000001  153014            MOV     #REQ,@CSR       ;CONNECT
2406 026606  105777  153010            7$:     TSTB    @CSR            ;CONNECTED?
2407 026612  100421                            BMI     8$              ;BR IF YES
2408 026614  005777  153002                    TST     @CSR            ;TIME OUT?
2409 026620  100372                            BPL     7$              ;BR IF NOT
2410 026622  017767  152774  152276            MOV     @CSR,BAD        ;STORE CONTENT OF CSR
2411 026630  016767  153116  152266            MOV     KONST,GOOD      ;
2412 026636  062767  020201  152260            ADD     #20201,GOOD     ;STORE GOOD DATA
2413 026644  104006                            ERROR+6                 ;COUNTN'T CONNECT
2414 026646  005077  152750                    CLR     @CSR            ;CLEAR CSR
2415 026652  000167  153734                    JMP     ROUT3
2416 026656                            8$:
2417 026656  004767  177514                    JSR     PC,CNTLC        ;
2418 026662  022702  027246                    CMP     #BREND,R2       ;IS IT END OF TABLE?
2419 026666  001504                            BEQ     5$              ;YES:BR
2420 026670  012706  001100                    MOV     #STACK,SP       ;RESTORE STACK
2421 026674  012737  000340  177776            MOV     #340,@#PSW      ;LOCK OUT INTERRUPT
2422 026702  004767  000670                    JSR     PC,CLRREG       ;CLEAR ALL UBE REGISTER
2423 026706  005067  000300                    CLR     BUFF1           ;CLEAR BUFF LOCATION
2424 026712  012777  177777  000250            MOV     #177777,@BEBD   ;LOAD UBE DATA REG WITH TEST DATA
2425 026720  012777  027212  000246            MOV     #BUFF1,@BEBA    ;LOAD UBE ADDRESS REG WITH BUFF ADR
2426 026726  012777  177777  000236            MOV     #177777,@BECC   ;SET UBE TO DO ONE CYCLE
2427 026734  012777  003041  000234            MOV     #3041,@BECR1    ;HAVE UBE SEND DATA VIA NPR
2428 026742  000240                            NOP                     ;ALLOW UBE TO SET BUS
2429 026744  004767  000702                    JSR     PC,CRDY         ;CHECK READY SET
2430 026750  005704                            TST     R4              ;DID RDY SET?
2431 026752  001037                            BNE     1$              ;BRANCH TO ERROR IF RDY DIDN'T SET
2432 026754  005767  000232                    TST     BUFF1           ;WAS DATAO DONE?
2433 026760  001445                            BEQ     2$              ;BRANCH TO ERROR IF NPR NOT DONE
2434 026762  005067  000224                    CLR     BUFF1           ;CLEAR BUF LOCATION
2435 026766  005067  000222                    CLR     BUFF1+2         ;CLEAR BUF LOCATION PLUS 2
2436 026772  012777  027040  000204            MOV     #3$,@INTVEC     ;SETUP FOR INTERRUPT
2437 027000  012777  027212  000166            MOV     #BUFF1,@BEBA    ;LOAD TEST ADDRESS
2438 027006  012777  177777  000156            MOV     #177777,@BECC   ;SETUP UBE TO DO 1 CYCLE
2439 027014  005267  000144                    INC     BRCNT           ;COUNT BR LEVEL
2440 027020  012277  000152                    MOV     (R2)+,@BECR1    ;HAVE UBE DO DATO NPR+INT
```

```
2441
2442
2443 027024  005037  177776                CLR     @#PSW           ;WHEN DONE VIA DIFF BR LEVEL
2444 027030  004767  000616                JSR     PC,CRDY         ;BR LEVEL = (4,5,6,7)
2445 027034  104025                        ERROR+25                ;ALLOW UBE TO INTERRUPT
2446 027036  000443                        BR      4$              ;WAIT FOR INTERRUPT OR READY TO SET
2447                                                               ;RESTORE TRAP
2448 027040  005767  000150          3$:   TST     BUFF1+2         ;DID NPR WRITE MORE THAN ONE LOC?
2449 027044  001440                        BEQ     4$              ;GO TO END OF TEST
2450 027046  104026                        ERROR+26
2451 027050  000436                        BR      4$
2452
2453 027052  104024                1$:   ERROR+24                ;ERROR:NPR DIDN'T SET RDY
2454 027054  012777  006003  000114        MOV     #6003,@BECR1    ;HAVE UBE SET ITS READY
2455 027062  005037  177776                CLR     @#PSW           ;
2456 027066  004767  000560                JSR     PC,CRDY         ;WAIT TILL SET
2457 027072  000425                        BR      4$              ;RESTORE TRAP
2458 027074  104023                2$:   ERROR+23
2459 027076  000423                        BR      4$
2460 027100                        5$:
2461 027100  022767  001750  000060        CMP     #1000.,NUM
2462 027106  001013                        BNE     11$
2463 027110  005267  000046                INC     PASS
2464 027114  104401  027560                TYPE    .ENDMG
2465 027120  016746  000036                MOV     PASS,-(SP)
2466 027124  104405                        TYPDS
2467 027126  104401  027555                TYPE    .NULL
2468 027132  005067  000030                CLR     NUM
2469 027136  004767  000466          11$:  JSR     PC,RCATCH
2470 027142  000167  177376                JMP NPR
2471
2472 027146  004767  000456          4$:   JSR     PC,RCATCH       ;RESTORE TRAP
2473 027152  004767  177220                JSR     PC,CNTLC        ;
2474 027156  000167  177474                JMP     8$              ;
2475
2476
2477 027162  000000                PASS:   0
2478 027164  000000                BRCNT:  0
2479 027166  000000                NUM:    0
2480 027170  170000                BEBD:   .WORD   170000
2481 027172  170002                BECC:   .WORD   170002
2482 027174  170004                BEBA:   .WORD   170004
2483 027176  170006                BECR1:  .WORD   170006
2484 027200  170010                BECR2:  .WORD   170010
2485 027202  170014                BERE:   .WORD   170014
2486 027204  000510                INTVEC: .WORD   510
2487 027206  000000                        0
2488 027210  000000                        0
2489
2490 027212                        BUFF1:  .BLKW   11
2491 027234  000000                        0
2492
2493 027236  003143                BRLV:   3143                    ;DATO AT BR4
2494 027240  003145                        3145                    ;DATO AT BR5
2495 027242  003151                        3151                    ;DATO AT BR6
2496 027244  003161                        3161                    ;DATO AT BR7
2497 027246  000000                BREND:  0
```

```
 2498 027250  000000                          0
 2499
 2500 027252      105      122      122  EM23:   .ASCIZ  /ERROR:NPR DATO NOT DONE/
      027255      117      122      072
      027260      116      120      122
      027263      040      104      101
      027266      124      117      040
      027271      116      117      124
      027274      040      104      117
      027277      116      105      000
 2501 027302      105      122      122  EM24:   .ASCIZ  /ERROR:NPR DID NOT SET RDY/
      027305      117      122      072
      027310      116      120      122
      027313      040      104      111
      027316      104      040      116
      027321      117      124      040
      027324      123      105      124
      027327      040      122      104
      027332      131      000
 2502 027334      105      122      122  EM25:   .ASCIZ  /ERROR:UBE DID NOT INTERRUPT WHEN NPR FINISHED/
      027337      117      122      072
      027342      125      102      105
      027345      040      104      111
      027350      104      040      116
      027353      117      124      040
      027356      111      116      124
      027361      105      122      122
      027364      125      120      124
      027367      040      127      110
      027372      105      116      040
      027375      116      120      122
      027400      040      106      111
      027403      116      111      123
      027406      110      105      104
      027411      000
 2503 027412      105      122      122  EM26:   .ASCIZ  /ERROR:TWO LOC WRITTEN WHEN ONE NPR AND INT ON DONE ENABLE/
      027415      117      122      072
      027420      124      127      117
      027423      040      114      117
      027426      103      040      127
      027431      122      111      124
      027434      124      105      116
      027437      040      127      110
      027442      105      116      040
      027445      117      116      105
      027450      040      116      120
      027453      122      040      101
      027456      116      104      040
      027461      111      116      124
      027464      040      117      116
      027467      040      104      117
      027472      116      105      040
      027475      105      116      101
      027500      102      114      105
      027503      000
 2504 027504      105      122      122  EM27:   .ASCIZ  /ERROR:DEVICE DOESN'T EXIST ON SWITCH BUS/
      027507      117      122      072
```

```
        027512  104     105     126
        027515  111     103     105
        027520  040     104     117
        027523  105     123     116
        027526  047     124     040
        027531  105     130     111
        027534  123     124     040
        027537  117     116     040
        027542  123     127     111
        027545  124     103     110
        027550  040     102     125
        027553  123     000
 2505
 2506 027555  377     377     000  NULL:   .BYTE   -1,-1,0          ;NULL CHARACTER STRING
 2507 027560  015     012     105  ENDMG:  .ASCIZ  <15><12>/END PASS #/
        027563  116     104     040
        027566  120     101     123
        027571  123     040     043
        027574  000
 2508
 2509                                         .EVEN
 2510
 2511                                 ;SUBROUTINE TO TYPE PC OF ERROR MESSAGE
 2512
 2513                                 ;TERRPC:BIT     @SW13,@SWR      ;INHIBIT ERROR PRINT OUT
 2514                                 ;       BNE     1$              ;BRANCH IF YES
 2515                                 ;       TYPE    ,MSG15          ;TYPE PC OF ERROR MESSAGE
 2516                                 ;       MOV     $ERRPC,-(SP)    ;SAVE $ERRPC FOR TYPEOUT
 2517                                 ;       TYPOC
 2518                                 ;1$:    RTS     PC
 2519
 2520                                 ;SUBROUTINE TO CLEAR ALL UBE REG
 2521                                 ;
 2522 027576                         CLRREG:
 2523 027576  005077  177400                 CLR     @#BERE  ;CLEAR ERROR CONDITION
 2524 027602  005077  177372                 CLR     @#BECR2 ;CLEAR BECR2 REG
 2525 027606  005077  177364                 CLR     @#BECR1 ;CLEAR BECR1 REG
 2526 027612  005077  177356                 CLR     @#BEBA  ;CLEAR BEBA REG
 2527 027616  005077  177350                 CLR     @#BECC  ;CLEAR BECC REG
 2528 027622  005077  177342                 CLR     @#BEBD  ;CLEAR BEBD REG
 2529 027626  000207                         RTS     PC
 2530
 2531                                 ;SUBROUTINE TO RESTORE TRAP CATCHER TO UBE VECTOR AREA
 2532
 2533 027630  010546                 RCATCH: MOV     R5,-(SP)        ;SAVE R5 ON STACK
 2534 027632  016705  177346                 MOV     INTVEC,R5       ;GET INTVEC
 2535 027636  005725                         TST     (R5)+           ;CALCULATE INTVEC+2
 2536 027640  010577  177340                 MOV     R5,@INTVEC      ;PUT INTVEC+2 IN INTVEC
 2537 027644  005015                         CLR     (R5)            ;PUT HALT IN INTVEC+2
 2538 027646  012605                         MOV     (SP)+,R5        ;RESTORE R5
 2539 027650  000207                         RTS     PC
 2540
 2541                                 ;SUBROUTINE TO CHECK RDY BIT SET
 2542
 2543 027652  005004                 CRDY:   CLR     R4
 2544 027654  005005                         CLR     R5
 2545 027656  005205                 2$:     INC     R5      ;UPDATE COUNTER
```

```
2546 027660  105777  177312              TSTB    @BECR1  ;SEE IF RDY SET
2547 027664  100405                      BMI     1$      ;BRANCH IF SET
2548 027666  032705  000200              BIT     #200,R5 ;WAITED>100MICROSEC
2549 027672  001771                      BEQ     2$      ;CONTINUE TO LOOK FOR RDY IF R5 NOT=128
2550 027674  012704  000001              MOV     #1,R4   ;SET R4=1 TO INDICATE ERROR
2551 027700  000207                1$:   RTS     PC      ;RETURN
2552
2553
2554
```

```
                                .NLIST  MC,MD,CND
                                .SBTTL  TYPE ROUTINE
                                ;;*****************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*
027702  105767  151251          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
027706  100002                          BPL     1$              ;;BR IF YES
027710  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
027712  000430                          BR      3$              ;;LEAVE
027714  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
027716  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
027722  122767  000001  151276          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
027730  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
027732  132767  000100  151267          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
027740  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
027742  010067  000004                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
027746  004767  000220                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
027752  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
027754  132767  000040  151245  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
027762  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
027764  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
027766  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
027770  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
027772  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
027774  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
030000  000002                          RTI                     ;;RETURN
030002  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
030006  001430                          BEQ     8$
030010  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
030014  001006                          BNE     5$
030016  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
030020  104401                          TYPE                    ;;TYPE A CR AND LF
030022  001203                          $CRLF
030024  105067  000130                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
030030  000755                          BR      2$              ;;GET NEXT CHARACTER
030032  004767  000056          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
030036  126726  151114          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
030042  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
030044  016746  151104                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
030050  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
030054  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
030056  004767  000032                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
030062  105367  000072                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
030066  000770                          BR      7$              ;;LOOP
                                ;HORIZONTAL TAB PROCESSOR
```

```
030070  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
030074  004767  000014          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
030100  132767  000007  000052          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
030106  001372                          BNE     9$              ;;TAB STOP
030110  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
030112  000724                          BR      2$              ;;GET NEXT CHARACTER
030114  105777  151030      $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
030120  100375                          BPL     $TYPEC
030122  116677  000002  151022          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
030130  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
030136  001003                          BNE     1$              ;;BRANCH IF NO
030140  105067  000014                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
030144  000406                          BR      $TYPEX          ;;EXIT
030146  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
030154  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
030156  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
030160  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
030162  000207                  $TYPEX: RTS     PC
2558                            .SBTTL  APT COMMUNICATIONS ROUTINE
                                ;;**************************************************************
030164  112767  000001  000236  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
030172  112767  000001  000226  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
030200  000403                          BR      $ATYC
030202  112767  000001  000220  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
030210                          $ATYC:
030210  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
030212  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
030214  105767  000206                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
030220  001450                          BEQ     5$              ;;IF NOT:  BR
030222  122767  000001  150776          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
030230  001031                          BNE     3$              ;;IF NOT:  BR
030232  132767  000100  150767          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
030240  001425                          BEQ     3$              ;;IF NOT:  BR
030242  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
030246  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
030254  005767  150726          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
030260  001375                          BNE     1$              ;;IF NOT:  WAIT
030262  010067  150734                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
030266  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
030270  001376                          BNE     2$
030272  166700  150724                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
030276  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
030300  010067  150720                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
030304  012767  000004  150674          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
030312  000413                          BR      5$
030314  017667  000004  000016  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
030322  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
030330  016746  147442                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
030334  004767  177342                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
030340  000000                  4$:     .WORD   0
030342                          5$:
030342  105767  000062          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
030346  001416                          BEQ     12$             ;;IF NOT:  BR
030350  005767  150652                  TST     $ENV            ;;RUNNING UNDER APT?
030354  001413                          BEQ     12$             ;;IF NOT:  BR
030356  005767  150624          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
030362  001375                          BNE     11$             ;;IF NOT:  WAIT
```

```
030364  017667  000004  150616          MOV     @4(SP),$FATAL   ;;GET ERROR #
030372  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
030400  005267  150602                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
030404  105067  000020          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
030410  105067  000013                  CLRB    $LFLG           ;;CLEAR LOG FLAG
030414  105067  000006                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
030420  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
030422  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
030424  000207                          RTS     PC              ;;RETURN
030426     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
030427     000          $LFLG:  .BYTE   0               ;;LOG FLAG
030430     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
                                .EVEN
             000200             APTSIZE=200
             000001             APTENV=001
             000100             APTSPOOL=100
             000040             APTCSUP=040
2559                            .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY.
                                ;*CALL:
                                ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
030432  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
030434  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
030442  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
030444  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
030446  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
030450  104411                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
030452  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
030454  005001                          CLR     R1              ;;CLEAR DATA WORD
030456  005002                          CLR     R2
030460  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
030462  001412                          BEQ     3$              ;;IF ZERO GET OUT
030464  006301                          ASL     R1              ;;*2
030466  006102                          ROL     R2
030470  006301                          ASL     R1              ;;*4
030472  006102                          ROL     R2
030474  006301                          ASL     R1              ;;*8
030476  006102                          ROL     R2
030500  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
030504  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
030506  000764                          BR      2$              ;;LOOP
030510  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
030512  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
030516  010267  000010                  MOV     R2,$HIOCT
030522  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
030524  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
030526  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
030530  000002                          RTI                     ;;RETURN
030532  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
2560                            .SBTTL  TTY INPUT ROUTINE
                                ;;**********************************************************************
                                .ENABL  LSB
                                ;;**********************************************************************
```

SEQ 0080

```
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                ;*WHEN OPERATING IN TTY FLAG MODE.
030534  022767  000176  150376  $CKSWR: CMP     @SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
030542  001074                          BNE     15$             ;;BRANCH IF NO
030544  105777  150374                  TSTB    @$TKS           ;;CHAR THERE?
030550  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
030552  117746  150370                  MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
030556  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
030562  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
030566  001062                          BNE     15$             ;;NO, RETURN TO USER
030570  126727  150340  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
030576  001456                          BEQ     15$             ;;BRANCH IF YES
030600  104401  031261                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
030604  104401  031266          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
030610  016746  147362                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
030614  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
030616  104401  031277                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
030622  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
030624  005046                          CLR     -(SP)           ;;THE NEW SWR
030626  105777  150312          7$:     TSTB    @$TKS           ;;CHAR THERE?
030632  100375                          BPL     7$              ;;IF NOT TRY AGAIN
030634  117746  150306                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
030640  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
030644  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
030650  001005                          BNE     10$             ;;BRANCH IF NOT
030652  104401  031254                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
030656  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
030662  000757                          BR      19$             ;;LET'S TRY IT AGAIN
030664  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
030670  001022                          BNE     16$             ;;BRANCH IF NO
030672  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
030676  001403                          BEQ     11$             ;;BRANCH IF YES
030700  016677  000002  150232          MOV     2(SP),@SWR      ;;SAVE NEW SWR
030706  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
030712  104401  001203          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
030716  126727  150213  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
030724  001003                          BNE     15$             ;;BRANCH IF NOT
030726  012777  000100  150210          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
030734  000002                  15$:    RTI                     ;;RETURN
030736  004767  177152          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
030742  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
030746  002420                          BLT     18$             ;;BRANCH IF YES
030750  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
030754  003015                          BGT     18$             ;;BRANCH IF YES
030756  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
030762  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
030766  001403                          BEQ     17$             ;;BRANCH IF YES
030770  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
030772  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
030774  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
030776  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
031002  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
031006  000707                          BR      7$              ;;GET THE NEXT ONE
031010  104401  001202          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
031014  000720                          BR      20$             ;;SIMULATE CONTROL-U
```

```
                                .DSABL  LSB
                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;
031016  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
031020  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
031026  105777  150112          1$:     TSTB    @$TKS           ;;WAIT FOR
031032  100375                          BPL     1$              ;;A CHARACTER
031034  117766  150106  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
031042  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
031050  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
031056  001013                          BNE     3$              ;;BRANCH IF NO
031060  105777  150060          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
031064  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
031066  117746  150054                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
031072  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
031076  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
031102  001366                          BNE     2$              ;;IF NOT DISCARD IT
031104  000750                          BR      1$              ;;YES, RESUME
031106  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
031114  002407                          BLT     4$              ;;BRANCH IF YES
031116  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
031124  003003                          BGT     4$              ;;BRANCH IF YES
031126  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
031134  000002                  4$:     RTI                     ;;GO BACK TO USER
                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
031136  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
031140  012703  031244          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
031144  022703  031254          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
031150  101405                          BLOS    4$              ;;BR IF YES
031152  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
031154  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
031156  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
031162  001003                          BNE     3$              ;;SKIP IF NOT
031164  104401  001202          4$:     TYPE    ,$QUES          ;;TYPE A '?'
031170  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
031172  111367  000044          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
031176  104401  031242                  TYPE    ,9$
031202  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
031206  001356                          BNE     2$              ;;LOOP IF NOT RETURN
031210  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
031214  104401  001204                  TYPE    ,$LF            ;;TYPE A LINE FEED
031220  012603                          MOV     (SP)+,R3        ;;RESTORE R3
031222  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
031224  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
031232  012766  031244  000004          MOV     #$TTYIN,4(SP)
031240  000002                          RTI                     ;;RETURN
031242     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
```

```
031243  000                             .BYTE   0               ;;TERMINATOR
031244                          $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
031254  136  125  015           $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
031257  012  000
031261  136  107  015           $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
031264  012  000
031266  015  012  123           $MSWR:  .ASCIZ  <15><12>/SWR = /
031271  127  122  040
031274  075  040  000
031277  040  040  116           $MNEW:  .ASCIZ  /  NEW = /
031302  105  127  040
031305  075  040  000
2561                            .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
                                ;;**************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                ;*      .BYTE   M               ;;M=1 OR 0
                                ;*                                      ;;1=TYPE LEADING ZEROS
                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                ;*
                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                ;*$TYPOS OR $TYPOC
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPON                   ;;CALL FOR TYPEOUT
                                ;*
                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOC                   ;;CALL FOR TYPEOUT
031310  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
031314  116667  000001  000211          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
031322  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
031326  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
031332  000406                          BR      $TYPON
031334  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
031342  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
031350  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
031356  010346                          MOV     R3,-(SP)        ;;SAVE R3
031360  010446                          MOV     R4,-(SP)        ;;SAVE R4
031362  010546                          MOV     R5,-(SP)        ;;SAVE R5
031364  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
031370  005404                          NEG     R4
031372  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
031376  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
031402  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
031406  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
031412  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
031414  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
031416  000404                          BR      3$              ;;GO DO MSB
031420  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
031422  006105                          ROL     R5
```

```
031424  006105                          ROL     R5
031426  010503                          MOV     R5,R3
031430  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
031432  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
031436  100016                          BPL     7$              ;;BR IF NO
031440  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
031444  001002                          BNE     4$              ;;TEST FOR 0
031446  005704                          TST     R4              ;;SUPPRESS THIS 0?
031450  001403                          BEQ     5$              ;;BR IF YES
031452  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
031454  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
031460  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
031464  110367  000040                  MOVB    R3,8$           ;;SAVE FOR TYPING
031470  104401  031530                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
031474  105367  000032          7$:     DECB    $OCNT           ;;COUNT BY 1
031500  003347                          BGT     2$              ;;BR IF MORE TO DO
031502  002402                          BLT     6$              ;;BR IF DONE
031504  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
031506  000744                          BR      2$              ;;GO DO THE LAST DIGIT
031510  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
031512  012604                          MOV     (SP)+,R4        ;;RESTORE R4
031514  012603                          MOV     (SP)+,R3        ;;RESTORE R3
031516  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
031524  012616                          MOV     (SP)+,(SP)
031526  000002                          RTI                     ;;RETURN
031530     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
031531     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
031532     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
031533     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
031534  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
  2562                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
                              ;;******************************************************************
                              ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                              ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                              ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                              ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                              ;*REPLACED WITH SPACES.
                              ;*CALL:
                              ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                              ;*      TYPDS                   ;;GO TO THE ROUTINE
031536                        $TYPDS:
031536  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
031540  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
031542  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
031544  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
031546  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
031550  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
031554  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
031560  100004                          BPL     1$              ;;BR IF INPUT IS POS.
031562  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
031564  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
031572  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
031574  012703  031752                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
031600  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
031604  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
031606  016001  031742                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
031612  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
```

```
031614  002402                          BLT     4$              ;;BR IF DONE
031616  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
031620  000774                          BR      3$
031622  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
031624  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
031626  001002                          BNE     5$              ;;FALL THROUGH IF 0
031630  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
031632  100407                          BMI     7$              ;;BR IF YES
031634  106316                  5$:     ASLB    (SP)            ;;MSD?
031636  103003                          BCC     6$              ;;BR IF NO
031640  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
031646  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
031652  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
031656  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
031660  005720                          TST     (R0)+           ;;JUST INCREMENTING
031662  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
031666  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
031670  003002                          BGT     8$              ;;GO TO EXIT
031672  010502                          MOV     R5,R2           ;;GET THE LSD
031674  000764                          BR      6$              ;;GO CHANGE TO ASCII
031676  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
031700  100003                          BPL     9$              ;;BR IF NO
031702  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
031710  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
031712  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
031714  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
031716  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
031720  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
031722  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
031724  104401  031752                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
031730  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
031736  012616                          MOV     (SP)+,(SP)
031740  000002                          RTI                     ;;RETURN TO USER
031742  023420                  $DTBL:  10000.
031744  001750                          1000.
031746  000144                          100.
031750  000012                          10.
031752                          $DBLK:  .BLKW   4
2563                            .SBTTL  ERROR HANDLER ROUTINE
                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO $ERRTYP ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1         HALT ON ERROR
                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                ;*SW10=1         BELL ON ERROR
                                ;*SW09=1         LOOP ON ERROR
                                ;*CALL
                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
031762                          $ERROR:
031762  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
031764  116767  147112  147640          MOVB    $TSTNM,TSTNUM   ;SET UP TEST # ON ERROR
031772  105267  147105          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
031776  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
032000  016777  147076  147134          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
032006  032777  002000  147124          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
```

```
032014  001402                          BEQ     1$              ;;NO - SKIP
032016  104401  001176                  TYPE    ,$BELL          ;;RING BELL
032022  005267  147064          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
032026  011667  147064                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
032032  162767  000002  147056          SUB     #2,$ERRPC
032040  117767  147052  147046          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
032046  032777  020000  147064          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
032054  001055                          BNE     20$             ;;SKIP TYPEOUTS
032056  021627  001002                  CMP     (SP),#1002      ;;IF RETURN PC LESS THAN 1002
032062  101046                          BHI     12$             ;;ERROR IS ILLEGAL TRAP
                        ;;PROCESS UNEXPECTED TRAP OR INTERRUPT
032064  016667  000004  147024          MOV     4(SP),$ERRPC    ;;GET PC AT TIME OF FALSE TRAP
032072  162767  000002  147016          SUB     #2,$ERRPC       ;;ADJUST PC
032100  104401  032144                  TYPE    ,10$            ;;TYPE HEADER
032104  016746  147006                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
032110  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
032112  104401  032152                  TYPE    ,11$
032116  162716  000004                  SUB     #4,(SP)         ;;GET FALSE TRAP VECTOR ADDR
032122  011667  146770                  MOV     (SP),$ERRPC
032126  016746  146764                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
032132  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
032134  104401  001203                  TYPE    ,$CRLF
032140  022626                          CMP     (SP)+,(SP)+     ;;POP FALSE TRAP VECTOR PC&ADDR
032142  000422                          BR      20$
032144     200     120     103  10$:    .ASCIZ  <200>'PC= '
032147     075     040     000
032152     040     040     125  11$:    .ASCIZ  '  UNEXPECTED TRAP TO '
032155     116     105     130
032160     120     105     103
032163     124     105     104
032166     040     124     122
032171     101     120     040
032174     124     117     040
032177     000
                                .EVEN
032200                      12$:
032200  004767  000106                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
032204  104401  001203                  TYPE    ,$CRLF
032210                      20$:
032210  122767  000001  147010          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
032216  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
032220  116767  146670  000004          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
032226  004767  175750                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
032232     000                  21$:    .BYTE   0
032233     000                          .BYTE   0
032234  000777                  22$:    BR      22$             ;;APT ERROR LOOP
032236  005777  146676          2$:     TST     @SWR            ;;HALT ON ERROR
032242  100002                          BPL     3$              ;;SKIP IF CONTINUE
032244  000000                          HALT                    ;;HALT ON ERROR!
032246  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
032250  032777  001000  146662  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
032256  001402                          BEQ     4$              ;;BR IF NO
032260  016716  146624                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
032264  005767  146704          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
032270  001402                          BEQ     5$              ;;BR IF NONE
032272  016716  146676                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
032276                      5$:
```

```
032276  022737  021416  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
032304  001001                          BNE     6$              ;;BRANCH IF NO
032306  000000                          HALT                    ;;YES
032310                          6$:
032310  000002                          RTI                     ;;RETURN
2564                            .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
                                ;;**********************************************************************
                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
032312                          $ERRTYP:
032312  104401  001203                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
032316  010046                          MOV     R0,-(SP)        ;;SAVE R0
032320  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
032322  153700  001114                  BISB    @#$ITEMB,R0
032326  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                ;;TYPE THE PC OF THE ERROR
032330  016746  146562                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                ;;ERROR ADDRESS
032334  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
032336  000426                          BR      6$              ;;GET OUT
032340  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
032342  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
032344  006300                          ASL     R0
032346  006300                          ASL     R0
032350  062700  001332                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
032354  012067  000004                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
032360  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
032362  104401                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
032364  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
032366  104401  001203                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
032372  012067  000004          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
032376  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
032400  104401                          TYPE                    ;;TYPE THE "DATA HEADER"
032402  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
032404  104401  001203                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
032410  011000                  5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
032412  001004                          BNE     7$              ;;GO TYPE THE DATA
032414  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
032416  104401  001203                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
032422  000207                          RTS     PC              ;;RETURN
032424                          7$:
032424  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
032426  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
032430  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
032432  001770                          BEQ     6$              ;;BR IF NO
032434  104401  032442                  TYPE    ,8$             ;;TYPE TWO(2) SPACES
032440  000771                          BR      7$              ;;LOOP
032442    040     040     000   8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
                                        .EVEN
2565                            .SBTTL  SCOPE HANDLER ROUTINE
                                ;;**********************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
```

```
                                        ;*SW11=1        INHIBIT ITERATIONS
                                        ;*SW09=1        LOOP ON ERROR
                                        ;*SW08=1        LOOP ON TEST IN SWR<7:0>
                                        ;*CALL
                                        ;*      SCOPE           ;;SCOPE=IOT
032446                                  $SCOPE:
032446  104407                                CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
                                        ;;GO TO ERROR ROUTINE IF RETURN PC LESS THAN 1002
                                        ;;OTHERWISE CONTINUE
032450  021627  001002                        CMP     (SP),#1002      ;;UNEXPECTED TRAP OR INTERRRUPT
032454  101002                                BHI     1$              ;;ARE TRAPPED HERE VIA IOT
032456  000167  177300                        JMP     $ERROR          ;;GO PROCESS UNEXPECTED TRAP
032462  032777  040000  146450  1$:           BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
032470  001114                                BNE     $OVER           ;;YES IF SW14=1
                                        ;#####START OF CODE FOR THE XOR TESTER#####
032472  000416                          $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
032474  013746  000004                        MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
032500  012737  032520  000004                MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
032506  005737  177060                        TST     @#177060        ;;TIME OUT ON XOR?
032512  012637  000004                        MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
032516  000463                                BR      $SVLAD          ;;GO TO THE NEXT TEST
032520  022626                          5$:   CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
032522  012637  000004                        MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
032526  000423                                BR      7$              ;;LOOP ON THE PRESENT TEST
032530                                  6$:;#####END OF CODE FOR THE XOR TESTER#####
032530  032777  000400  146402                BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
032536  001404                                BEQ     2$              ;;BR IF NO
032540  127767  146374  146334                CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
032546  001465                                BEQ     $OVER           ;;BR IF YES
032550  105767  146327                  2$:   TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
032554  001421                                BEQ     3$              ;;BR IF NO
032556  126767  146333  146317                CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
032564  101015                                BHI     3$              ;;BR IF NO
032566  032777  001000  146344                BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
032574  001404                                BEQ     4$              ;;BR IF NO
032576  016767  146306  146302  7$:           MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
032604  000446                                BR      $OVER
032606  105067  146271                  4$:   CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
032612  005067  146354                        CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
032616  000415                                BR      1$              ;;ESCAPE TO THE NEXT TEST
032620  032777  004000  146312  3$:           BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
032626  001011                                BNE     1$              ;;BR IF YES
032630  005767  146360                        TST     $PASS           ;;IF FIRST PASS OF PROGRAM
032634  001406                                BEQ     1$              ;;      INHIBIT ITERATIONS
032636  005267  146242                        INC     $ICNT           ;;INCREMENT ITERATION COUNT
032642  026767  146324  146234                CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
032650  002024                                BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
032652  012767  000001  146224  1$:           MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
032660  016767  000052  146304                MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
032666  105267  146210                $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
032672  116767  146204  146312                MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
032700  011667  146202                        MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
032704  011667  146200                        MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
032710  005067  146260                        CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
032714  112767  000001  146173                MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
032722  016777  146154  146212  $OVER:        MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
```

```
        032730  016716  146152                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
        032734  000002                          RTI                     ;;FIXES PS
        032736  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
  2566                                  .SBTTL  POWER DOWN AND UP ROUTINES
                                        ;;**********************************************************************
                                        ;POWER DOWN ROUTINE
        032740  012737  033104  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
        032746  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
        032754  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
        032756  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
        032760  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
        032762  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
        032764  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
        032766  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
        032770  017746  146144                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
        032774  010667  000110                  MOV     SP,$SAVR6       ;;SAVE SP
        033000  012737  033012  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
        033006  000000                          HALT
        033010  000776                          BR      .-2             ;;HANG UP
                                        ;;**********************************************************************
                                        ;POWER UP ROUTINE
        033012  012737  033104  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
        033020  016706  000064                  MOV     $SAVR6,SP       ;;GET SP
        033024  005067  000060                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
        033030  005267  000054          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
        033034  001375                          BNE     1$              ;;OF  WORD
        033036  012677  146076                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
        033042  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
        033044  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
        033046  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
        033050  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
        033052  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
        033054  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
        033056  012737  032740  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
        033064  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
        033072  104401                          TYPE                    ;;REPORT THE POWER FAILURE
        033074  033112                  $PWRMG: .WORD   PWRMSG           ;;POWER FAIL MESSAGE POINTER
        033076  012716                          MOV     (PC)+,(SP)      ;;RESTART AT RESTRT
        033100  002526                  $PWRAD: .WORD   RESTRT           ;;RESTART ADDRESS
        033102  000002                          RTI
        033104  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
        033106  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
        033110  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
  2567  033112     015     012     122  PWRMSG: .ASCIZ  <15><12>/RESTARTED FROM POWER FAIL/
        033115     105     123     124
        033120     101     122     124
        033123     105     104     040
        033126     106     122     117
        033131     115     040     120
        033134     117     127     105
        033137     122     040     106
        033142     101     111     114
        033145     000
  2568                                  .EVEN
  2569                                  .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.
033146  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
033150  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
033154  005740                          TST     -(R0)           ;;BACKUP BY 2
033156  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
033160  006300                          ASL     R0              ;;POSITION FOR INDEXING
033162  016000  033202                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
033166  000200                          RTS     R0              ;;GO TO ROUTINE
                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
033170  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
033172  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
033200  000002                          RTI                     ;;RESTORE THE PSW
                                .SBTTL  TRAP TABLE
                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE "TRAP" INSTRUCTION.
                                ;       ROUTINE
                                ;       -------
033202  033170                  $TRPAD: .WORD   $TRAP2
033204  027702                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
033206  031334                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
033210  031310                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
033212  031350                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
033214  031536                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
033216  030604                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
033220  030534                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
033222  031016                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
033224  031136                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
033226  030432                          $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
2570                            .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
                                ;;*****************************************************************
                                ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
                                ;*WITH A RANGE OF 0 TO 2(+33)-1.
                                ;*CALL:
                                ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
                                ;*      RETURN                  ;;RETURN HERE THE RANDOM
                                ;*                              ;;NUMBER WILL BE IN
                                ;*                              ;;$HINUM,$LONUM
033230                          $RAND:
033230  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
033232  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
033234  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
033236  016700  000066                  MOV     $LONUM,R0       ;;SET R0 WITH LOW
033242  016701  000060                  MOV     $HINUM,R1       ;;SET R1 WITH HIGH
033246  012702  177771                  MOV     #-7,R2          ;;SET SHIFT COUNT
033252  006300                  1$:     ASL     R0              ;;SHIFT R0 LEFT AND
033254  006101                          ROL     R1              ;;ROTATE CARRY INTO R1 AND
033256  005202                          INC     R2              ;;CHECK FOR DONE
033260  001374                          BNE     1$              ;;CONTINUE SHIFT LOOP
033262  066700  000042                  ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
033266  005501                          ADC     R1              ;;PROPOGATE CARRY
033270  066701  000032                  ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
033274  062700  001057                  ADD     #1057,R0        ;;ADD LOW CONSTANT
033300  005501                          ADC     R1              ;;PROPOGATE CARRY
033302  062701  047401                  ADD     #47401,R1       ;;ADD HIGH CONSTANT
033306  010067  000016                  MOV     R0,$LONUM       ;;SAVE R0
```

RANDOM NUMBER GENERATOR ROUTINE

SEQ 0090

```
033312  010167  000010                  MOV     R1,$HINUM       ;;SAVE R1
033316  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
033320  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
033322  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
033324  000207                          RTS     PC              ;;RETURN
033326  176543                  $HINUM: .WORD   176543
033330  123456                  $LONUM: .WORD   123456
```

```
                                                .SBTTL  ASCII MESSAGES

033332  120     117     123     EM1:    .ASCIZ  /POST-RESET CONDITIONS NOT MET IN CSR./
033335  124     055     122
033340  105     123     105
033343  124     040     103
033346  117     116     104
033351  111     124     111
033354  117     116     123
033357  040     116     117
033362  124     040     115
033365  105     124     040
033370  111     116     040
033373  103     123     122
033376  056     000
033400  122     105     101     EM2:    .ASCIZ  %READ/WRITE BIT FAILURE IN CSR.%
033403  104     057     127
033406  122     111     124
033411  105     040     102
033414  111     124     040
033417  106     101     111
033422  114     125     122
033425  105     040     111
033430  116     040     103
033433  123     122     056
033436  000
033437  102     131     124     EM3:    .ASCIZ  /BYTE ACCESS FAILURE IN CSR./
033442  105     040     101
033445  103     103     105
033450  123     123     040
033453  106     101     111
033456  114     125     122
033461  105     040     111
033464  116     040     103
033467  123     122     056
033472  000
033473  105     122     122     EM4:    .ASCIZ  /ERROR IN CONNECT PROCESS./
033476  117     122     040
033501  111     116     040
033504  103     117     116
033507  116     105     103
033512  124     040     120
033515  122     117     103
033520  105     123     123
033523  056     000
033525  111     116     124     EM5:    .ASCIZ  /INTERUPT DETECTED WITH INTERUPT ENABLE CLEARED./
033530  105     122     125
033533  120     124     040
033536  104     105     124
033541  105     103     124
033544  105     104     040
033547  127     111     124
033552  110     040     111
033555  116     124     105
033560  122     125     120
033563  124     040     105
033566  116     101     102
```

```
                033571  114     105     040
                033574  103     114     105
                033577  101     122     105
                033602  104     056     000
     2579       033605  103     117     125     EM6:    .ASCIZ  /COULD NOT CONNECT./
                033610  114     104     040
                033613  116     117     124
                033616  040     103     117
                033621  116     116     105
                033624  103     124     056
                033627  000
     2580       033630  105     122     122     EM7:    .ASCIZ  /ERROR IN RELEASE PROCESS./
                033633  117     122     040
                033636  111     116     040
                033641  122     105     114
                033644  105     101     123
                033647  105     040     120
                033652  122     117     103
                033655  105     123     123
                033660  056     000
     2581       033662  116     117     040     EM10:   .ASCIZ  /NO INTERUPT DETECTED./
                033665  111     116     124
                033670  105     122     125
                033673  120     124     040
                033676  104     105     124
                033701  105     103     124
                033704  105     104     056
                033707  000
     2582       033710  111     116     124     EM11:   .ASCIZ  /INTERUPT AT WRONG PRIORITY./
                033713  105     122     125
                033716  120     124     040
                033721  101     124     040
                033724  127     122     117
                033727  116     107     040
                033732  120     122     111
                033735  117     122     111
                033740  124     131     056
                033743  000
     2583       033744  103     117     116     EM12:   .ASCIZ  /CONNECT CONDITION FAILURE./
                033747  116     105     103
                033752  124     040     103
                033755  117     116     104
                033760  111     124     111
                033763  117     116     040
                033766  106     101     111
                033771  114     125     122
                033774  105     056     000
     2584       033777  105     122     122     EM13:   .ASCIZ  /ERROR IN CONNECT FAILURE PROCESS./
                034002  117     122     040
                034005  111     116     040
                034010  103     117     116
                034013  116     105     103
                034016  124     040     106
                034021  101     111     114
                034024  125     122     105
                034027  040     120     122
                034032  117     103     105
```

```
                034035  123     123     056
                034040  000
        2585    034041  105     122     122     EM14:   .ASCIZ  /ERROR IN RELEASE TEST WITHOUT BUSS MASTERSHIP./
                034044  117     122     040
                034047  111     116     040
                034052  122     105     114
                034055  105     101     123
                034060  105     040     124
                034063  105     123     124
                034066  040     127     111
                034071  124     110     117
                034074  125     124     040
                034077  102     125     123
                034102  123     040     115
                034105  101     123     124
                034110  105     122     123
                034113  110     111     120
                034116  056     000
        2586    034120  122     105     114     EM15:   .ASCIZ  /RELEASE CONDITIONS NOT MET./
                034123  105     101     123
                034126  105     040     103
                034131  117     116     104
                034134  111     124     111
                034137  117     116     123
                034142  040     116     117
                034145  124     040     115
                034150  105     124     056
                034153  000
        2587    034154  124     115     117     EM16:   .ASCIZ  /TMO BIT NOT SET./
                034157  040     102     111
                034162  124     040     116
                034165  117     124     040
                034170  123     105     124
                034173  056     000
        2588    034175  103     117     125     EM17:   .ASCIZ  %COULDN'T READ/WRITE EXTERNAL DEVICE.%
                034200  114     104     116
                034203  047     124     040
                034206  122     105     101
                034211  104     057     127
                034214  122     111     124
                034217  105     040     105
                034222  130     124     105
                034225  122     116     101
                034230  114     040     104
                034233  105     126     111
                034236  103     105     056
                034241  000
        2589    034242  122     105     123     EM20:   .ASCIZ  /RESET HOLD (HLD) DIDN'T DISABLE RESET./
                034245  105     124     040
                034250  110     117     114
                034253  104     040     050
                034256  110     114     104
                034261  051     040     104
                034264  111     104     116
                034267  047     124     040
                034272  104     111     123
                034275  101     102     114
```

```
        034300     105     040     122
        034303     105     123     105
        034306     124     056     000
 2590   034311     122     105     123  EM21:   .ASCIZ  /RESET (RST) DIDN'T WORK./
        034314     105     124     040
        034317     050     122     123
        034322     124     051     040
        034325     104     111     104
        034330     116     047     124
        034333     040     127     117
        034336     122     113     056
        034341     000
 2591   034342     124     110     105  EM22:   .ASCIZ  /THE OPERATION OF THE RESET-IN-NEUTRAL HAS CHANGED./
        034345     040     117     120
        034350     105     122     101
        034353     124     111     117
        034356     116     040     117
        034361     106     040     124
        034364     110     105     040
        034367     122     105     123
        034372     105     124     055
        034375     111     116     055
        034400     116     105     125
        034403     124     122     101
        034406     114     040     110
        034411     101     123     040
        034414     103     110     101
        034417     116     107     105
        034422     104     056     000
 2592   034425     040     000          SPACE:  .ASCIZ  / /
 2593   034427     200     060     075  SELMS:  .ASCII  <200>/0=THIS MESSAGE/
        034432     124     110     111
        034435     123     040     115
        034440     105     123     123
        034443     101     107     105
 2594   034446     200     061     075          .ASCII  <200>/1=MONOPORT TEST/
        034451     115     117     116
        034454     117     120     117
        034457     122     124     040
        034462     124     105     123
        034465     124
 2595   034466     200     062     075          .ASCII  <200>/2=MULTIPORT TEST/
        034471     115     125     114
        034474     124     111     120
        034477     117     122     124
        034502     040     124     105
        034505     123     124
 2596   034507     200     063     075          .ASCII  <200>/3=MANUAL INTERVENTION TEST/
        034512     115     101     116
        034515     125     101     114
        034520     040     111     116
        034523     124     105     122
        034526     126     105     116
        034531     124     111     117
        034534     116     040     124
        034537     105     123     124
 2597   034542     200     064     075          .ASCIZ  <200>/4=TEST NPR LINE/
```

```
        034545  124  105  123
        034550  124  040  116
        034553  120  122  040
        034556  114  111  116
        034561  105  000
 2598   034563  105  122  122  DH1:    .ASCII  /ERROR  TEST#   SHOULD  WAS     FROM/<200>
        034566  117  122  011
        034571  124  105  123
        034574  124  043  011
        034577  123  110  117
        034602  125  114  104
        034605  011  127  101
        034610  123  011  106
        034613  122  117  115
        034616  200
 2599   034617  120  103  011          .ASCIZ  /PC             BE/
        034622  011  102  105
        034625  000
 2600   034626  105  122  122  DH2:    .ASCII  /ERROR  TEST#   WROTE   READ    DEVICE ADDRESS/<200>
        034631  117  122  011
        034634  124  105  123
        034637  124  043  011
        034642  127  122  117
        034645  124  105  011
        034650  122  105  101
        034653  104  011  104
        034656  105  126  111
        034661  103  105  040
        034664  101  104  104
        034667  122  105  123
        034672  123  200
 2601   034674  120  103  000          .ASCIZ  /PC/
 2602   034677  105  122  122  DH3:    .ASCII  /ERROR  TEST#   BITS FAILING RESET      DEVICE ADDRESS/<200>
        034702  117  122  011
        034705  124  105  123
        034710  124  043  011
        034713  102  111  124
        034716  123  040  106
        034721  101  111  114
        034724  111  116  107
        034727  040  122  105
        034732  123  105  124
        034735  011  104  105
        034740  126  111  103
        034743  105  040  101
        034746  104  104  122
        034751  105  123  123
        034754  200
 2603   034755  120  103  000          .ASCIZ  /PC/
 2604   034760  105  122  122  DH4:    .ASCII  /ERROR  TEST#   RESET   READ    DEVICE ADDRESS/<200>
        034763  117  122  011
        034766  124  105  123
        034771  124  043  011
        034774  122  105  123
        034777  105  124  011
        035002  122  105  101
        035005  104  011  104
```

```
                035010  105     126     111
                035013  103     105     040
                035016  101     104     104
                035021  122     105     123
                035024  123     200
 2605           035026  120     103     040             .ASCIZ  /PC              BITS/
                035031  040     040     040
                035034  040     040     040
                035037  040     040     040
                035042  040     040     040
                035045  102     111     124
                035050  123     000
 2606           035052  105     122     122     DH5:    .ASCII  /ERROR/<200>
                035055  117     122     200
 2607           035060  040     120     103             .ASCIZ  / PC /
                035063  040     000
 2608           035065  105     122     122     DH6:    .ASCII  /ERROR          BR/<200>
                035070  117     122     040
                035073  040     040     040
                035076  040     040     040
                035101  040     040     040
                035104  102     122     200
 2609           035107  040     120     103             .ASCIZ  / PC            LEVEL/
                035112  040     040     040
                035115  040     040     040
                035120  040     040     040
                035123  040     040     114
                035126  105     126     105
                035131  114     000
 2610                                                   .EVEN
 2611           035134  001116  001632  001124  DT1:    $ERRPC,TSTNUM,GOOD,BAD,BADA,0
                035142  001126  001122  000000
 2612           035150  001116  001632  001640  DT2:    $ERRPC,TSTNUM,DEVRW,BAD,DEVAD,0
                035156  001126  001636  000000
 2613           035164  001116  001632  001126  DT3:    $ERRPC,TSTNUM,BAD,DEVAD,0
                035172  001636  000000
 2614           035176  001116  000000          DT4:    $ERRPC,0
 2615           035202  001116  027164  000000  DT5:    $ERRPC,BRCNT,0
 2616                   002612                          .END    ROUT3
```

SYMBOL TABLE

SEQ 0097

```
ABASE = 177420
ACDW1 = 000000
ACDW2 = 000000
ACPUOP= 000000
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000000
AENV  = 000000
AENVM = 000000
AFATAL= 0C0000
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2= 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
APASS = 000000
APRIOR= 000000
APTCSU= 000040
APTENV= 000001
APTSIZ= 000200
APTSPO= 000100
ASWREG= 000000
ATESTN= 000000
AUNIT = 000000
AUSWR = 000000
AVECT1= 160350
AVECT2= 000000
BAD   = 001126
BADA  = 001122
BEBA    027174
BEBD    027170
BECC    027172
BECR1   027176
BECR2   027200
BERE    027202
BIT0  = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1  = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4  = 000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BPTVEC= 000014
BRCNT   027164
BREND   027246
BRLV    027236
BUFF1   027212
CKADR   026340
CKSWR = 104407
CLRREG  027576
CNTLC   026376
CONID   021510
CONNEC  021452
COVID   001730
CR    = 000015
CRDY    027652
CRLF  = 000200
CSR     001622
CURFUN  001722
CURID   001724
CYCLE   001726
DDISP = 177570
DEVAD   001636
DEVCON  001754
DEVRC   001642
DEVRW   001640
DH1     034563
DH2     034626
DH3     034677
DH4     034760
DH5     035052
DH6     035065
DISPAT  007722
DISPLA  001142
DISPRE  000174
DMODE   022754
DSWR  = 177570
DTIME   023316
DT03    001634
DT1     035134
DT2     035150
DT3     035164
DT3SP   014760
DT4     035176
DT5     035202
ECHOB   026206
EMT   = 104000
EMTVEC= 000030
EM1     033332
EM10    033662
EM11    033710
EM12    033744
EM13    033777
EM14    034041
EM15    034120
EM16    034154
EM17    034175
EM2     033400
EM20    034242
EM21    034311
EM22    034342
EM23    027252
EM24    027302
EM25    027334
EM26    027412
EM27    027504
EM3     033437
EM4     033473
EM5     033525
EM6     033605
EM7     033630
ENDMG   027560
ERREN = 017510
ERROR = 104000
ERROUT  026310
ERRVEC= 000004
EXPEND  001654
EXPRT1  001646
EXPRT2  001650
EXPRT3  001652
EXPTCT  001656
FLAG    026336
FLUSHM  022702
FLUSHP  022724
FUNCT1  010164
FUNCT2  010660
FUNCT3  011706
FUNCT4  012076
FUNCT5  012374
GETADR  024010
GOOD  = 001124
GTSWR = 104406
HT    = 000011
INTVEC  027204
IOT   = 000004
IOTVEC= 000020
KIPAR0= 172340
KIPAR1= 172342
KIPAR2= 172344
KIPAR3= 172346
KIPAR4= 172350
KIPAR5= 172352
KIPAR6= 172354
KIPAR7= 172356
KIPDR0= 172300
KIPDR1= 172302
KIPDR2= 172304
KIPDR3= 172306
KIPDR4= 172310
KIPDR5= 172312
KIPDR6= 172314
KIPDR7= 172316
KONST   001752
LF    = 000012
MANEND  021256
MANIN   015056
MANMOD  015074
MASMEN  022206
MASTER  007740
MENU    001676
MMVEC = 000250
MONEND  007544
MONPRT  002676
MULDT3  012540
MULEND  015014
MULPRT  007556
NEXENT  001674
NPR     026544
NULL    027555
NUM     027166
ONOFF   001744
OOPS    022600
PARFLG  026204
PARIN   024534
PASS    027162
PFVSEV  017746
PIRQ  = 177772
PIRQVE= 000240
PORTNO  001644
POSTSV= 020564
PRI     001630
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSW   = 177776
PWFSER  020602
PWRMSG  033112
PWROK   016154
PWRVEC= 000024
RCATCH  027630
RDCHR = 104410
RDLIN = 104411
RDOCT = 104412
REPAR = 024552
REQ   = 000001
RESTRT  002526
RESVEC= 000010
ROUTE   002672
ROUT1   002566
ROUT2   002600
ROUT3   002612
ROUT4   002636
R6    =%000006
R7    =%000007
SALMEN  022274
SCOPE = 000004
SECT2   012770
SECT2A  013370
SECT2B  013434
SECT2C  013664
SECT2D  013402
SECT3   013674
SECT4   014150
SECT5   014374
SELMS   034427
SEQ     001660
SEQINT  022444
SEQMAK  022076
SEQPTR  001672
SEQSTP  022466
SETUP1  002232
SETUP2  002260
SETUP3  002302
SLAVE   010054
SPACE   034425
SR0   = 177572
SR1   = 177574
SR2   = 177576
SR3   = 172516
STACK = 001100
START   001762
STKLMT= 177774
STPNEX  022542
SWBPWF  016740
SWR     001140
```

SYMBOL TABLE

SEQ 0098

```
SWREG   000176
SW0   = 000001
SW00  = 000001
SW01  = 000002
SW02  = 000004
SW03  = 000010
SW04  = 000020
SW05  = 000040
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW1   = 000002
SW10  = 002000
SW11  = 004000
SW12  = 010000
SW13  = 020000
SW14  = 040000
SW15  = 100000
SW2   = 000004
SW3   = 000010
SW4   = 000020
SW5   = 000040
SW6   = 000100
SW7   = 000200
SW8   = 000400
SW9   = 001000
SYNCUP  021642
TABM    012526
TBITVE= 000014
TEMP1   001732
TEMP2   002674
TICK    021636
TIMA    001746
TIMB    001750
TKVEC = 000060
TOCK    021640
TPVEC = 000064
TRAPVE= 000034
TRAP4   026222
TRIG    021564
TRIN    001756
TRTVEC= 000014
TSTNPR  026430
TSTNUM  001632
TST1    003060
TST10   005032
TST11   005240
TST12   005530
TST13   005710
TST14   006154
TST15   006546
TST16   006702
TST17   007050
TST2    003152
TST3    003404
TST4    003524
TST5    003760
TST6    004242
TST7    004550
TX      026212
TYPDS = 104405
TYPE  = 104401
TYPOC = 104402
TYPON = 104404
TYPOS = 104403
USEPAR  001760
VEC0    001624
VEC2    001626
WAIT    021600
WAIT05  021566
WAIT10  021540
WAIT20  021552
$APTHD  001000
$ATYC   030210
$ATY1   030164
$ATY3   030172
$ATY4   030202
$AUTOB  001134
$BASE   001262
$BDADR  001122
$BDDAT  001126
$BELL   001176
$CDW1   001266
$CDW2   001270
$CHARC  030160
$CKSWR  030534
$CMTAG  001100
$CM3  = 000000
$CM4  = 000005
$CNTLG  031261
$CNTLU  031254
$CPUOP  001234
$CRLF   001203
$DBLK   031752
$DDW0   001272
$DDW1   001274
$DDW10  001316
$DDW11  001320
$DDW12  001322
$DDW13  001324
$DDW14  001326
$DDW15  001330
$DDW2   001276
$DDW3   001300
$DDW4   001302
$DDW5   001304
$DDW6   001306
$DDW7   001310
$DDW8   001312
$DDW9   001314
$DEVCT  001216
$DEVM   001264
$DOAGN  021426
$DTBL   031742
$ENDAD  C21416
$ENDCT  021364
$ENDMG  021435
$ENULL  021432
$ENV    001226
$ENVM   001227
$EOP    021330
$EOPCT  021356
$ERFLG  001103
$ERMAX  001115
$ERROR  031762
$ERRPC  001116
$ERRTB  001332
$ERRTY  032312
$ERTTL  001112
$ESCAP  001174
$ETABL  001226
$ETEND  001332
$FATAL  001210
$FFLG   030430
$FILLC  001156
$FILLS  001155
$GDADR  001120
$GDDAT  001124
$GET42  021406
$GTSWR  030604
$HIBTS  001000
$HINUM  033326
$HIOCT  030532
$ICNT   001104
$ILLUP  033104
$INTAG  001135
$ITEMB  001114
$LF     001204
$LFLG   030427
$LONUM  033330
$LPADR  001106
$LPERR  001110
$MADR1  001240
$MADR2  001244
$MADR3  001250
$MADR4  001254
$MAIL   001206
$MAMS1  001236
$MAMS2  001242
$MAMS3  001246
$MAMS4  001252
$MBADR  001002
$MFLG   030426
$MNEW   031277
$MSGAD  001222
$MSGLG  001224
$MSGTY  001206
$MSWR   031266
$MTYP1  001237
$MTYP2  001243
$MTYP3  001247
$MTYP4  001253
$MXCNT  032736
$NULL   001154
$NWTST= 000001
$OCNT   031532
$OMODE  031534
$OVER   032722
$PASS   001214
$PASTM  001006
$PWRAD  033100
$PWRDN  032740
$PWRMG  033074
$PWRUP  033012
$QUES   001202
$RAND   033230
$RDCHR  031016
$RDLIN  031136
$RDOCT  030432
$RDSZ = 000010
$RTNAD  021430
$SAVR6  033110
$SCOPE  032446
$SETUP= 000137
$STUP = 177777
$SVLAD  032666
$SVPC = 000204
$SWR  = 167400
$SWREG  001230
$SWRMK= 000000
$TESTN  001212
$TIMES  001172
$TKB    001146
$TKS    001144
$TMP0   001160
$TMP1   001162
$TMP2   001164
$TMP3   001166
$TMP4   001170
$TN   = 000020
$TPB    001152
$TPFLG  001157
$TPS    001150
$TRAP   033146
$TRAP2  033170
$TRP  = 000013
$TRPAD  033202
$TSTM   001004
$TSTNM  001102
$TTYIN  031244
$TYPDS  031536
$TYPE   027702
$TYPEC  030114
$TYPEX  030162
$TYPOC  031334
$TYPON  031350
$TYPOS  031310
$UNIT   001220
$UNITM  001010
$USWR   001232
$VECT1  001256
$VECT2  001260
$XTSTR  032472
$$GET4= 000000
$0FILL  031533
$40CAT  000000
.$X   = 001000

. ABS.  035210      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  51568 WORDS  ( 202 PAGES)
DYNAMIC MEMORY:  20060 WORDS  ( 77 PAGES)
ELAPSED TIME:  20.03:27
CRDTAC,CRDTAC/-SP=CRD.PC.MLB/ML,CRDTAC.P11
```