DR11W

DR11-W INTRPROC EXER

CZDRKA0

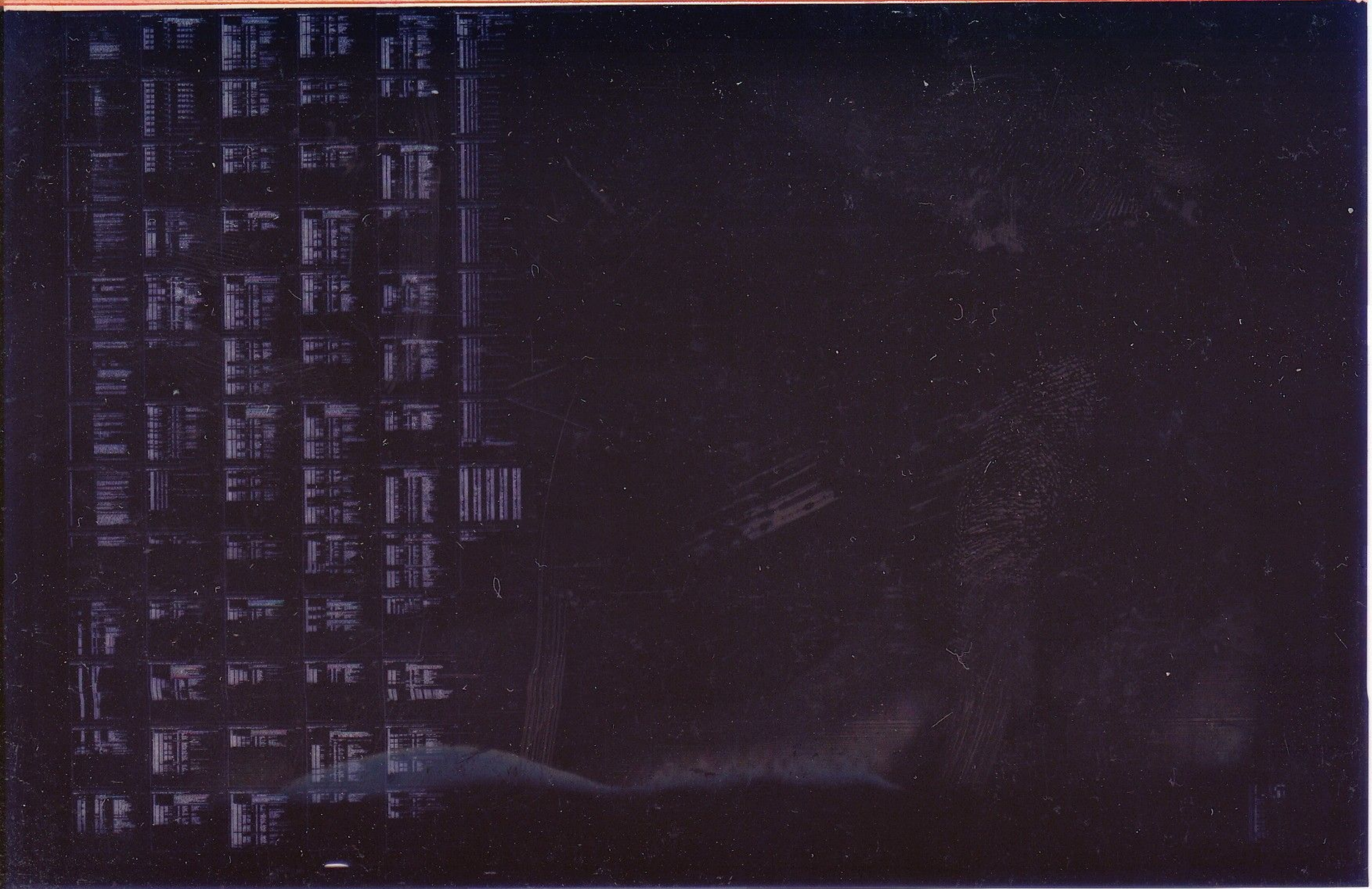
AH-E783A-MC

FICHE 1 OF 1

JAN 1980

digital

MADE IN USA



```
                        .REM    %
```

IDENTIFICATION

PRODUCT CODE:	AC-F782A-MC
PRODUCT NAME:	CZDRKA0 DR11-W INTRPROC EXER
DATE:	AUG 1979
MAINTAINER:	DIAGNOSTIC ENGINEERING

TABLE OF CONTENTS

1.0 ABSTRACT

THE DR11W INTERPROCESSOR EXERCISER WAS DESIGNED TO PASS DATA BETWEEN
TWO COMPUTERS. ONE COMPUTER CONTAINS A DR11W DMA INTERFACE
AND THE OTHER CONTAINS EITHER A DR11W OR A DRV11B DMA INTERFACE.
THE FOLLOWING CONFIGURATIONS ARE VALID

A.

LSI-11 DRV11-B	----------	PDP-11 DR11-W

B.

PDP-11 DR11-W	----------	PDP-11 DR11-W

EACH CONFIGURATION USES THE DR11W EXERCISER LOADED INTO EACH
OF THE COMPUTERS INVOLVED.

2.0 REQUIREMENTS

2.1 EQUIPMENT

1. ONE PDP11 COMPUTER EQUIPPED WITH A DR11-W.
2. ONE LSI-11 EQUIPPED WITH A DRV11-B OR ANOTHER PDP11 WITH A DR11-W.
3. EACH SYSTEM EQUIPPED WITH AN I/O TERMINAL
4. CABLES (TWO) TO INTER CONNECT THE TWO DMA INTERFACES

2.2 STORAGE

THE PROGRAM USES THE LOWER 3400 WORDS OF MEMORY

3.0 LOADING AND START PROCEDURE

1. CHOOSE ONE OF THE INTERPROCESSOR CONFIGURATIONS SPECIFIED
   AND LOAD THE DR11-W INTERPROCESSOR EXERCISER INTO EACH SYSTEM.

1. MAKE SURE EACH DMA INTERFACE OF BOTH COMPUTER SYSTEMS
   ARE CABLED TOGETHER THRU THE I/O CONNECTIONS

2. MAKE SURE THE DEVICE & VECTOR ADDRESSES AGREE WITH THE DEFAULT VALUES DEFINED IN SECTION 6.1. IF NOT, CHANGE LOCATION(S) AS DESIRED VIA A 'PATCH' METHOD CONSISTENT WITH WITH THE SYSTEM USED.

3. INSURE THAT THE HALT SWITCH IS DISABLED (IF ANY).

4. DEFINE WHAT COMPUTER IS TO BE THE INITIAL SLAVE - THE OTHER WILL BE THE INITIAL MASTER (THE ONE STARTED AS MASTER WILL REPORT THE 'END OF PASS' MESSAGE).

5. START THE SLAVE **FIRST** AT ADDRESS 204. AFTER THE SLAVE HAS BEEN STARTED THE FOLLOWING PRINTOUT WILL OCCUR:

```
DR11W INTERPROCESSOR EXERCISER

IS THIS CPU TESTING A DRV11B OR DR11W?
TYPE V OR W:
```

THE OPERATOR TYPES V FOR DRV11B OR W FOR DR11W. IF THE USER WERE TO TYPE 'V' THEN THE FOLLOWING WOULD BE PRINTED:

```
DRV11B BOARD
INTERPROCESSOR LINK NOW IN PROGRESS....
```

AT THIS POINT THE SLAVE IS WAITING FOR THE MASTER TO BE STARTED.

6. NOW START THE OTHER COMPUTER(MASTER) AT SA 200. SIMILARLY THE PROGRAM WILL REQUEST THE BOARD TYPE. WHEN USER ANSWERS (LET'S SAY WITH A W) THE PRINTOUT WILL SHOW:

```
DR11W BOARD
INTERPROCESSOR LINK NOW IN PROGRESS....
```

THE TWO SYSTEMS SHOULD NOW BE COMMUNICATING WITH THE TESTS BEING PERFORMED.

THE END OF PASS MESSAGE WILL SUSEQUENTLY OCCUR AS FOLLOWS:

```
END PASS        #       1
END PASS        #       2
        .
        .
      ETC.
```

THE FIRST END PASS IS WITH NO ITERATIONS; ALL OTHERS ARE WITH ITERATIONS.

4.0 ERRORS

4.1 ERROR COMMENT

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

4.2 ERROR DATA

ERRPC	LISTING ADDRESS WHERE THE ERROR WAS DETECTED
BUSADR	DRV11B BUS REG ADDRESS OF CONCERNED OPERATION
EXPCT	DATA THAT WAS EXPECTED
RCVD	DATA THAT WAS RECEIVED
MEMADR	MEMORY ADDRESS OF DATA ERROR

4.3 ERROR RECOVERY

4.3.1 RECOVERABLE ERRORS

BECAUSE OF THE SYNCHRONIZATION ESTABLISHED WITH THE OTHER COMPUTER,
MOST ERROR CONDITIONS WILL CAUSE BOTH COMPUTERS TO RE-SYNCHRONIZE
AT THE START OF THE PROGRAM. WHEN AN ERROR OCCURS, AN ERROR MESSAGE
WILL BE PRINTED. FOLLOWING THIS , A DELAY OF 5-10 SEC WILL OCCUR
AND THE COMPUTER STARTED AS SLAVE WILL PRINT:

*RESYNC...

THIS INDICATES THAT THIS CPU HAS SUCCESSFULLY STARTED ITS PROGRAM
AGAIN AND IS WAITING FOR MASTER CPU.
APPROX. 5-10 SEC LATER THE CPU STARTED AS MASTER WILL TYPE:

*RESYNC...

AT THIS POINT BOTH CPU'S SHOULD BE AGAIN COMMUNICATING AND TESTING
WILL HAVE BEEN RESUMED FROM THE START OF THE PROGRAM.

4.3.2 NON-RECOVERABLE ERRORS

SOME ERRORS CAN OCCUR DUE TO DMA INTERFACE MALFUNCTIONS,
INTERMITTENT CABLE CONNECTIONS, ETC.

1. AN ERROR OF THIS TYPE MAY RESULT IN ONE OR BOTH SYSTEMS
   STUCK IN ONE OF THE 'WAIT LOOPS' FOUND IN THE PROGRAM.
   THE WAIT LOOP COULD BE A SOFTWARE LOOP INVOKED BY A SYSTEM WHILE
   WAITING FOR ITS COMPANION TO SET A BIT IN IT'S REGISTER.
   IF THIS NEVER COMES THEN THE SYSTEM WAITS INDEFINITELY AND
   SYNCHRONIZATION IS LOST.

2. A TEST SUCH AS 'BURST DATA LATE 'MAY FAIL AND HANG
   THE BUS OF ONE SYSTEM.
   IN THIS CASE, THE CPU FAILS TO EXECUTE ANY FURTHER INSTRUCTIONS .

AS STATED IN 4.3.1 MOST ERROR CONDITIONS CAN BE HANDLED BY
RESYNCHRONIZATION. HOWEVER,THE PROGRAMS WILL NOT BE ABLE TO RECOVER

THEMSELVES FROM THE ABOVE ERROR TYPES. THEREFORE IT BEHOOVES THE OPERATOR TO
PERIODICALLY CHECK FOR 'END PASS' PRINTOUTS WHICH WILL OCCUR
APPROX. EVERY 2 MINUTES. THIS WILL VERIFY THAT BOTH SYSTEMS
ARE STILL IN SYNC.
IF SYNCHRONIZATION IS LOST,BOTH SYSTEMS MUST BE HALTED WHEREBY THE ADDRESS
LOCATION OF EACH PROGRAM CAN BE DETECTED THRU 'ADDRESS' REGISTER
OR 'ODT' PRINTOUT. BOTH PROGRAMS MUST BE RE-STARTED AS SPECIFIED
IN SECT. 3.0.

5.0 SOFTWARE SWITCH REGISTER

5.1 OPTIONS

SWITCH	OCTAL	FUNCTION
BIT15=1	100000	HALT ON ERROR
BIT13=1	020000	INHIBIT ERROR TYPEOUTS
BIT10=1	002000	BELL ON ERROR

5.2 IMPLEMENTATION

IF THE DIAGNOSTIC IS RUN ON A CPU WITHOUT A SWITCH
REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED WHICH ALLOWS
THE USER THE SAME SWITCH OPTIONS AS THE HARDWARE SWITCH REGISTER.
IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES
AND IT CONTAINS ALL ONES (177777) THEN THE SOFTWARE SWITCH
REGISTER (LOC. 176) IS USED. UNNDER THESE CONDITIONS
THE PROGRAM WILL REQUEST THAT THE SOFTWARE SWITCH REGISTER
BE LOADED UPON 1ST PASS OF THE PROGRAM. IT WILL PRINT:

SWR:=XXXXXX NEW= (USER ENTERS IN FOLLOWING NEW=)

5.3 CONTROL

1. THE SOFTWARE SWITCH REGISTER 'SWREG' (LOC. 176) CAN BE
   CHANGED BY USING ODT FACILITIES.

2. THE SOFTWARE SWITCH REGISTER CAN BE CHANGED UNDER PROGRAM
   CONTROL BY TYPING THE 'CONTROL & G' KEYS. THIS KEYBOARD
   OPERATION WILL PRINT OUT THE CURRENT CONTENTS AND ACCEPT
   NEW OCTAL SWITCH REGISTER DATA TERMINATED WITH A CARRIAGE
   RETURN.

   WHEN USING THE ODT MODE AND ONCE IT HAS BEEN ENTERED DUE TO AN ERROR
   CONDITION WITH BIT15 SET (HALT ON ERROR), STEP #2 ABOVE
   IS OF NO VALUE, SO RESORT TO STEP #1 TO ALTER THE CONTENTS.

6.0 MISCELLANEOUS

6.1 DRV11B AND/OR DR11W BUS & VECTOR ADDRESS MODIFICATION

MODIFY LOCATION 'INTADR' IF BASE BUS ADDRESS IS NOT 172410
MODIFY LOCATION 'DRVECT' IF VECTOR ADDRESS IS NOT 124

NOTE: USE APPROPRIATE PATCH FACILITIES TO MODIFY THESE LOCATIONS AFTER PROGRAM LOAD.

6.2 POWER FAIL

THE PROGRAM OFFERS NO PROVISIONS FOR RESTART PROCEDURES DUE TO POWER FAIL. UPON POWER UP, AND ASSUMING NON VOLATILE MEMORY, THE PROGRAM OF EACH SYSTEM MUST BE RESTARTED ACCORDING TO SECT. 3.0.

7.0 EXECUTION TIME

EXECUTION TIME IS ABOUT 3 SEC FOR NO ITERATIONS
EXECUTION TIME W/ITERATIONS IS DETERMINED BY THE SYSTEM CONFIGURATION
ACTUAL RUN TIMES ARE AS FOLLOWS:

- 11/34 - 11/05: 3MIN
- 11/70 - 11/45: 45SEC.
- 11/70 - 11/03: 1MIN 15SEC.

8.0 PROGRAM DESCRIPTION

8.1 GENERAL

THIS INTERPROCESSOR EXERCISER WAS DESIGNED TO TEST THE I/O ABILITY OF THE DR11W GENERAL PURPOSE INTERFACE TO COMMUNICATE TO ANOTHER DR11W OR DRV11B LOCATED IN ANOTHER SYSTEM. THE TWO COMPUTERS ARE STARTED AT DIFFERENT ADDRESSES TO ESTABLISH INITIAL SYNCHRONIZATION. THE SLAVE COMPUTER IS STARTED 1ST (START 204) AND THE MASTER COMPUTER IS STARTED 2ND (START 200). THE TERMS 'MASTER' AND 'SLAVE' SHOULD BE USED LOOSELY AS THE MASTER WILL BECOME THE SLAVE AND THE SLAVE WILL BECOME THE MASTER AS THE PROGRAM ADVANCES. THE COMPUTER STARTED AT ADDRESS 200 WILL ALWAYS REPORT THE 'END OF PASS' MESSAGE.

8.2 PROGRAM SEGMENTS

1A. MTST1 - MASTER SENDS OUT PROGRAM CONTROLLED SINGLE WORDS
THRU THE DATA BUFFER REGISTER AND EXPECTS THE SLAVE TO ECHO

```
                        EACH WORD BACK TO THE MASTER VIA THE DATA BUFFER REGISTER.

                    1B.  STST1 - SLAVE ECHOS ALL CHANGES IN THE DATA BUFFER REGISTER
                    ----------

                    2A.  TEST THAT SLAVE CAN INTERRUPT MASTER: F2(SLAVE) TO  ATTN(MASTER)

                    2B.  TEST THAT SLAVE INTERRUPTS MASTER:F2 TO ATTN

                    ----------
                    3A.  MTST2 - MASTER SENDS OUT A 'FNCT' BIT CODE IN THE COMMAND
                        STATUS REGISTER AND EXPECTS THE SLAVE TO ECHO EACH CODE IN
                        ITS 'FNCT' BITS. THE MASTER WILL READ THE 'STAT' BIT CODE
                        FROM THE COMMAND/STATUS REGISTER AND COMPARE IT TO THE
                        CODE WRITTEN.

                    3B.  STST2 - SLAVE READS THE 'STAT' BIT CODE FROM IT'S COMMAND/
                        STATUS REGISTER, CONVERTS THIS CODE AND WRITES IT INTO
                        IT'S 'FNCT' BITS IN THE COMMAND/STATUS REGISTER.

                    ----------
                    4A.  MTST3-BLOCK MODE XFER-MASTER SENDS OUT A 32 WORD
                         DATA BLOCK TO THE SLAVE
                        AND CHECKS FOR PROPER INTERRUPT STATUS, WORD COUNT,
                        AND BUFFER ADDRESS AT THE COMPLETION OF THE TRANSFER.

                    4B.  STST3-BLOCK MODE XFER-SLAVE RECEIVES A 32 WORD DATA BLOCK FROM THE
                        MASTER AND CHECKS FOR PROPER INTERRUPT STATUS, WORD COUNT,
                        BUFFER ADDRESS, AND DATA CONTENT.

                    ----------

                    5A.   MTST4 BURST MODE WORD XFER-MASTER XMITS 32 WORDS TO SLAVE
                          (DR11-W TO (DR11-W ONLY)
                        SAME AS 4A EXCEPT BURST MODE:BOARD DOES NOT RELEASE BUS
                        UNTIL WORD COUNT OVERFLOW

                    5B.  STST4 BURST MODE WORD XFER TEST-SLAVE RECEIVES 32 WORDS
                        FROM MASTER(DR11W TO DR11W ONLY)
                        SAME AS 4B EXCEPT BURST MODE

                    ----------
                    6A. MTST5 BURST DATA LATE TEST-MASTER XMITS 5 WORDS TO SLAVE
                        WHILE SLAVE RECEIVES 5 AND TIMES OUT (DR11-W TO DR11-W ONLY)

                        PROCEDURE: MASTER XMITS 5 WORDS TO SLAVE IN BURST MODE
                        SLAVE IS SETUP TO DO 10 XFERS. CHECK THAT MASTER INTERRUPTS
                        BY WCOF. CHECK CSR,WC,AND BA. AFTER 5 XFERS ARE DONE IN
                        SLAVE,TIMEOUT OCCURS(50US), SINCE SLAVE WAITS FOR GO-AHEAD
                        FROM MASTER TO CONTINUE,BUT MASTER NEVER RESPONDS. IN
                        SLAVE: CHECK THAT 5 XFERS HAVE BEEN COMPLETED.
                        NOTE: 50US IS MAX TIMEOUT THAT REPRESENTS LENGTH OF TIME
                        BOARD HOLDS THE BUS WHILE WAITING FOR COMPANION . IT IS BEYOND
                        THE CAPABILITY OF THIS DIAGNOSTIC TO VERIFY A 50US TIMEOUT.
                        IN FACT, IF THE TIMEOUT FEATURE MALFUNCTIONS SO AS TO HOLD
                        THE BUS INDEFINITELY,THE PROGRAM WILL 'HANG UP' THE CPU
                        WITH NO RECOVERY PROVIDED FOR.
```

```
                                        THEREFORE,OPERATOR SCRUTINY IS ADVISABLE  TO VERIFY
                                        THAT THE PROGRAM IS STILL RUNNING AND THAT THE DR11-W HAS
                                        AT LEAST RELEASED THE BUS IN TEST 5.

                                    6B.   STST5 BURST DATA LATE TEST-SLAVE RECEIVES 5 WORDS
                                          FROM MASTER AND TIMES OUT WHILE EXPECTING MORE
                                         (DR11-W TO DR11-W ONLY)
                                        SETS UP TO RECEIVE 10 WORDS IN BURST MODE FROM MASTER,BUT MASTER WILL ONLY SEND
                                         TIMEOUT OCCURS AND THE BUS IS RELEASED
                                        THEN CHECKS FOR PROPER INTERRUPT STATUS, WC, BA & DATA
                                %
```

```
                                .TITLE  CZDRKA0 DR11-W INTRPROC EXER
                                ;*COPYRIGHT (C) 1979
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY JOHN W. CIUKAJ
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
000001                          $TN=1
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
122000                          $SWR=122000
000001                          $TN=1
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        10            BELL ON ERROR
                                .SBTTL  BASIC DEFINITIONS
                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
104000                          ERROR=EMT
000004                          SCOPE=IOT
                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
177776                          PSW=PS
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3=     %3              ;;GENERAL REGISTER
000004                          R4=     %4              ;;GENERAL REGISTER
000005                          R5=     %5              ;;GENERAL REGISTER
000006                          R6=     %6              ;;GENERAL REGISTER
000007                          R7=     %7              ;;GENERAL REGISTER
000006                          SP=     %6              ;;STACK POINTER
000007                          PC=     %7              ;;PROGRAM COUNTER
                                ;*PRIORITY LEVEL DEFINITIONS
000000                          PR0=    0               ;;PRIORITY LEVEL 0
000040                          PR1=    40              ;;PRIORITY LEVEL 1
000100                          PR2=    100             ;;PRIORITY LEVEL 2
000140                          PR3=    140             ;;PRIORITY LEVEL 3
000200                          PR4=    200             ;;PRIORITY LEVEL 4
000240                          PR5=    240             ;;PRIORITY LEVEL 5
000300                          PR6=    300             ;;PRIORITY LEVEL 6
```

```
                        ;
000340                  PR7=    340             ;;PRIORITY LEVEL 7
                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
001000                  SW9=SW09
000400                  SW8=SW08
000200                  SW7=SW07
000100                  SW6=SW06
000040                  SW5=SW05
000020                  SW4=SW04
000010                  SW3=SW03
000004                  SW2=SW02
000002                  SW1=SW01
000001                  SW0=SW00
                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
001000                  BIT9=BIT09
000400                  BIT8=BIT08
000200                  BIT7=BIT07
000100                  BIT6=BIT06
000040                  BIT5=BIT05
000020                  BIT4=BIT04
000010                  BIT3=BIT03
000004                  BIT2=BIT02
000002                  BIT1=BIT01
000001                  BIT0=BIT00
                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
```

```
                    000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                    000014                  TBITVEC=14              ;;"T" BIT
                    000014                  TRTVEC= 14              ;;TRACE TRAP
                    000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                    000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                    000024                  PWRVEC= 24              ;;POWER FAIL
                    000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                    000034                  TRAPVEC=34              ;;"TRAP" TRAP
                    000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
                    000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
                    000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
    475             106427                  MTPS=106427             ;INSTR EQUATE THAT MOVES BYTE TO PSW
    476                                     .SBTTL  TRAP CATCHER
                    000000                          .=0
                                            ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                            ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                            ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                    000174                          .=174
    000174  000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
    000176  000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                    .SBTTL  STARTING ADDRESS(ES)
    000200  000137  001506                  JMP     @#START1        ;;JUMP TO STARTING ADDRESS OF PROGRAM
    477 000204  000137  001514              JMP     @#START2        ;GO START AS SLAVE COMPUTER
    478         000100                      .=100
    479 000100  000104  000200  000002      .WORD   104,200,2       ;IF 'B EVENT' ON Q-BUS IS CONNECTED
    480                                                             ;JUST DO A RTI (IGNORE IT)
```

```
                                .SBTTL  COMMON TAGS
                                ;;**********************************************************************
                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;*USED IN THE PROGRAM.
                        001100          .=1100
001100                  001100  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102     000                  $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103     000                  $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114     000                  $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115     001                  $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134     000                  $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135     000                  $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                          .WORD   0
001140  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560                  $TKS:   177560                  ;;TTY KBD STATUS
001146  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154     000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155     002                  $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156     012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157     000                  $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160     207     377     377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001163     000
001164     077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
001165     015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001166     012     000          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**********************************************************************
```

```
                                .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
    001170                      $ERRTB:
 482
 483 001170  013754             ;ERROR  1
                                        EM1             ;SLAVE  FAILED TO ECHO DBR CONTENTS
 484 001172  015633                     DH1             ;ERRPC  BUSADR  EXPCT   RCVD
 485 001174  015734                     DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
 486 001176  000000                     0
 487
 488                            ;ERROR  2
 489 001200  014047                     EM2             ;SLAVE  FAILED TO ECHO 'STAT' BITS
 490 001202  015633                     DH1             ;ERRPC  BUSADR  EXPCT   RCVD
 491 001204  015734                     DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
 492 001206  000000                     0
 493
 494                            ;ERROR  3
 495 001210  014130                     EM3             ;FAILED TO INTR ON A 'DATI'
 496 001212  015633                     DH1             ;ERRPC  BUSADR  EXPCT   RCVD
 497 001214  015734                     DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
 498 001216  000000                     0
 499
 500                            ;ERROR  4
 501 001220  014203                     EM4             ;STATUS ER ON 'DATI'
 502 001222  015633                     DH1             ;ERRPC  BUSADR  EXPCT   RCVD
 503 001224  015734                     DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
 504 001226  000000                     0
```

```
                                ;ERROR  5
001230  014247                          EM5             ;WORD COUNT ER ON 'DATI'
001232  015633                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001234  015734                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001236  000000                          0

                                ;ERROR  6
001240  014317                          EM6             ;BUFFER ADRS ER ON 'DATI'
001242  015633                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001244  015734                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001246  000000                          0

                                ;ERROR  7
001250  014370                          EM7             ;FAILED TO INTR ON A 'DATO'
001252  015633                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001254  015734                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001256  000000                          0
                                ;ERROR  10
001260  014442                          EM10            ;STATUS ER ON 'DATO'
001262  015633                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001264  015734                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001266  000000                          0

                                ;ERROR  11
001270  014505                          EM11            ;WORD COUNT ER ON 'DATO'
001272  015633                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001274  015734                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001276  000000                          0

                                ;ERROR  12
001300  014554                          EM12            ;BUFFER ADRS ER ON 'DATO'
001302  015633                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001304  015734                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001306  000000                          0

                                ;ERROR  13
001310  014624                          EM13            ;DATA ER ON 'DATO'
001312  015670                          DH2             ;ERRPC  MEMADR  EXPCT   RCVD
001314  015734                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001316  000000                          0

                                ;ERROR  14
001320  014665                          EM14            ;CSR READY BIT WAS SET - SHOULD BE CLEAR
001322  015633                          DH1
001324  015734                          DT1
001326  000000                          0


                                ;ERROR 15
001330  014753                          EM15    ;PROTOCOL INITIALIZATION ERROR -MASTER AND SLAVE
001332  015725                          DH3
001334  015746                          DT2
001336  000000                          0

                                ;ERROR 16
001340  015075                          EM16    ;STUCK FOREVER WAITING FOR COMPANION
001342  015725                          DH3
```

```
    563 001344  015752                        DT3
    564 001346  000000                        0
    565
    566                                ;ERROR 17
    567 001350  015147                        EM17    ;TIMEOUT ERROR BURST MODE XFER-INCURRED INTERRUPT
    568 001352  015725                        DH3
    569 001354  015746                        DT2
    570 001356  000000                        0
    571
    572                                ;ERROR 20       ;SLAVE DID NOT INTERRUPT MASTER
    573 001360  015241                        EM20
    574 001362  015725                        DH3
    575 001364  015746                        DT2
    576 001366  000000                        0
    577
    578                                ;ERROR 21
    579 001370  015354                        EM21            ;TEST 2-ERROR BIT SET IN MASTER
    580 001372  015725                        DH3
    581 001374  015752                        DT3
    582 001376  000000                        0
    583
    584                                ;ERROR  22
    585 001400  015446                        EM22            ;BURST DATA LATE BIT IN EIR NOT SET
    586 001402  015633                        DH1
    587 001404  015734                        DT1
    588 001406  000000                        0
    589
```

```
                                ; BASE REGISTER ADDRESS ASSIGNMENT
                                
001410  172410                  INTADR: 172410          ;MODIFY THIS LOC IF DIFFERENT
                                
                                ; VECTOR ADDRESS ASSIGNMENT
                                
001412  000124                  DRVECT: 124             ;MODIFY THIS LOC IF DIFFERENT
                                
                                ; BUS REGISTER ADDRESS POINTERS
                                
001414  172410                  WCR:    172410          ;WORD COUNT
001416  172412                  BAR:    172412          ;BUFFER ADDRESS
001420  172414                  CSR:    172414          ;COMMAND/STATUS
001422  172416                  BDR:    172416          ;BUFFER DATA REGISTER
                                
                                ; VECTOR ADDRESS POINTERS
                                
001424  000124                  DRVCT0: 124             ;READY & NEX VECTOR
001426  000126                  DRVCT2: 126             ;NEW PSW ON INTR
                                
                                ;COMMON PROGRAM LOCATION(S)
                                
001430  000000                  CNT:    0
001432  000000                  NUM:    0
001434  000000                  BOARD:  0               ;DESCRIBES BOARD TYPE:DR11W OR DRV11B
001436  000000                  ABORT:  0               ;TIMER FOR ABORTING PROGRAM
001440  000000                  TESTPC: 0
001442  000000                  TOTAL:  0               ;USED TO DETERMINE BOARD TYPE
001444  000000                  TIME:   0               ;GENERAL PURPOSE COUNTER
001446  000000                  TEMP:   0
001450  000000                  SAVE:   0               ;REG DATA SAVED HERE
001452  000000                  MSTER:  0               ;0=MASTER START - NON-ZERO=SLAVE START
001454  000001                  ICOUNT: 1               ;# OF TIMES TO REPEAT ALL TESTS BEFORE END PASS MSG
001456  177740                  XPRAM:  -32.            ;XMIT WORD COUNT
001460  015756                          DBUF            ;XMIT BUFFER ADRS
001462  000511                          511             ;XMIT STATUS/CONTROL
001464  177740                  RPRAM:  -32.            ;RCV WORD COUNT
001466  015756                          DBUF            ;RCV BUFFER ADRS
001470  000113                          113             ;RCV STATUS/CONTROL
                                
001472  177773                  XMITMO:-5               ;XMIT BURST DATA LATE TEST.
001474  015756                           DBUF
001476  000501                           501
                                
001500  177766                  RCVTMO:-10.             ;RCV BURST DATA LATE TEST
001502  015756                         DBUF
001504  000103                         103
```

```
                                        .SBTTL  PROGRAM START
001506  005037  001452          START1: CLR     MSTER           ;THIS IT MASTER START
001512  000403                          BR      START           ;SKIP NEXT
001514  012737  177777  001452  START2: MOV     #-1,MSTER       ;SLAVE START
001522                          START:
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001522  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001526  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001530  022706  001140                  CMP     #SWR,R6 ;;DONE?
001534  001374                          BNE     .-6             ;;LOOP BACK IF NO
001536  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001542  012737  012452  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001550  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001556  012737  012172  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001564  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001572  012737  013320  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001600  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001606  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001612  012737  001646  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001620  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001626  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001634  022777  177777  177276          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001642  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
001644  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
001646  012716  001654          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
001652  000002                          RTI
001654  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001662  012737  000174  001142          MOV     #DISPREG,DISPLAY
001670  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
001674  012700  001414                  MOV     #WCR,R0 ;SET UP REG ADRS POINTERS
001700  013701  001410                  MOV     INTADR,R1       ;GET BASE ADRS
001704  010120                  SETUP2: MOV     R1,(R0)+        ;LOAD EM
001706  062701  000002                  ADD     #2,R1           ;
001712  022700  001424                  CMP     #BDR+2,R0       ;ALL DONE?
001716  001372                          BNE     SETUP2          ;BR IF NOT
001720  012700  001424                  MOV     #DRVCT0,R0      ;SET UP  VECTOR ADRS POINTER
001724  013701  001412                  MOV     DRVECT,R1       ;GET BASE VECTOR ADRS
001730  010120                  SETUP3: MOV     R1,(R0)+        ;
001732  062701  000002                  ADD     #2,R1           ;
001736  022700  001430                  CMP     #DRVCT2+2,R0    ;ALL DONE?
001742  001372                          BNE     SETUP3          ;BR IF NOT
                                .SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
001744  005227  177777                  INC     #-1             ;;FIRST TIME?
001750  001041                          BNE     64$             ;;BRANCH IF NO
001752  104401  002010                  TYPE    ,65$            ;;TYPE ASCIZ STRING
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
001756  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
001762  001006                          BNE     66$             ;;BRANCH IF YES
001764  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
001772  001005                          BNE     67$             ;;BRANCH IF NO
001774  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
```

```
        001776  000403                          BR      67$
        002000  112737  000001  001134  66$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
        002006                                  67$:
        002006  000422                          BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  <CRLF>#DR11W INTERPROCESSOR EXERCISER  #<CRLF>
        002054                                  64$:
    662 002054  000240                          NOP
```

```
                                                .SBTTL  BOARD TYPE DIALOGUE
002056                                  BRDTYP:
002056  012706  001100                          MOV     #STACK,SP       ;ALWAYS RESET STACK PTR
002062  000005                                  RESET                   ;INITIALIZE  BEFORE TESTING
002064  012737  000001  001454                  MOV     #1,ICOUNT       ;1ST TIME DO ALL TEST ONCE - THEN 100(10)
002072  005227  177777                          INC     #-1             ;DETERMINE BOARD TYPE ON FIRST PASS ONLY
002076  001402                                  BEQ     5$
002100  000137  002524                          JMP     MORS
002104                                  5$:
002104  104401  002112                          TYPE    ,65$            ;;TYPE ASCIZ STRING
002110  000425                                  BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>/IS THIS CPU TESTING A DRV11B OR DR11W?/<CRLF>
002164                                  64$:
002164  104401  002172                          TYPE    ,67$            ;;TYPE ASCIZ STRING
002170  000411                                  BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  /TYPE V OR W:    /
002214                                  66$:
002214  104410                                  RDCHR
002216  021627  000126                  2$:     CMP     (SP),#126
002222  001051                                  BNE     3$
002224  012737  000126  001434                  MOV     #126,BOARD
002232  104401  002240                          TYPE    ,69$            ;;TYPE ASCIZ STRING
002236  000411                                  BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  /V/<CRLF><CRLF>/DRV11B BOARD/<CRLF>
002262                                  68$:
002262  104401  002270                          TYPE    ,71$            ;;TYPE ASCIZ STRING
002266  000426                                  BR      70$             ;;GET OVER THE ASCIZ
                                        ;;71$:  .ASCIZ  /INTERPROCESSOR LINK NOW IN PROGRESS...../<CRLF><CRLF>
002344                                  70$:
002344  000576                                  BR      SYNCST
002346  021627  000127                  3$:     CMP     (SP),#127
002352  001414                                  BEQ     6$
002354  104401  002362                          TYPE    ,73$            ;;TYPE ASCIZ STRING
002360  000410                                  BR      72$             ;;GET OVER THE ASCIZ
                                        ;;73$:  .ASCIZ  /ERROR IN ENTRY/
002402                                  72$:
002402  000640                                  BR      5$
002404  012737  000127  001434          6$:     MOV     #127,BOARD
002412  104401  002420                          TYPE    ,75$            ;;TYPE ASCIZ STRING
002416  000410                                  BR      74$             ;;GET OVER THE ASCIZ
                                        ;;75$:  .ASCIZ  /W/<CRLF><CRLF>/DR11W BOARD/<CRLF>
002440                                  74$:
002440  104401  002446                          TYPE    ,77$            ;;TYPE ASCIZ STRING
002444  000426                                  BR      76$             ;;GET OVER THE ASCIZ
                                        ;;77$:  .ASCIZ  /INTERPROCESSOR LINK NOW IN PROGRESS...../<CRLF><CRLF>
002522                                  76$:
002522  000507                                  BR      SYNCST
002524                                  MORS:
002524  022737  000126  001434          4$:     CMP     #126,BOARD
002532  001403                                  BEQ     16$
002534  104401  002645                          TYPE    ,7$
002540  000500                                  BR      15$
002542  104401  002550                  16$:    TYPE    ,8$
002546  000475                                  BR      15$
002550      200       200       040   8$:     .ASCII  <CRLF><CRLF>/ DRV11B BOARD/<CRLF>
002553      104       122       126
002556      061       061       102
```

```
        002561  040     102     117
        002564  101     122     104
        002567  200
   697  002570  111     116     124             .ASCII  /INTERPROCESSOP LINK NOW IN PROGRESS...../<CRLF><CRLF><CRLF>
        002573  105     122     120
        002576  122     117     103
        002601  105     123     123
        002604  117     122     040
        002607  114     111     116
        002612  113     040     116
        002615  117     127     040
        002620  111     116     040
        002623  120     122     117
        002626  107     122     105
        002631  123     123     056
        002634  056     056     056
        002637  056     200     200
        002642  200
   698  002643  040     000                     .ASCIZ  / /
   699  002645  200     200     040     7$:     .ASCII  <CRLF><CRLF>/ DR11W BOARD /<CRLF>
        002650  104     122     061
        002653  061     127     040
        002656  102     117     101
        002661  122     104     040
        002664  200
   700  002665  111     116     124             .ASCII  /INTERPROCESSOR LINK NOW IN PROGRESS...../<CRLF><CRLF><CRLF>
        002670  105     122     120
        002673  122     117     103
        002676  105     123     123
        002701  117     122     040
        002704  114     111     116
        002707  113     040     116
        002712  117     127     040
        002715  111     116     040
        002720  120     122     117
        002723  107     122     105
        002726  123     123     056
        002731  056     056     056
        002734  056     200     200
        002737  200
   701  002740  040     000                     .ASCIZ  / /
   702                                          .EVEN
```

```
                                        .SBTTL
                                        .SBTTL          MASTER TESTS
                                        .SBTTL
002742                          15$:
002742  004737  013576        SYNCST: JSR     PC,CPUHI
002746  005737  001452                TST     MSTER           ;START AS MASTER?
002752  001402                        BEQ     MINIT1          ;YES - GO SEND TO SLAVE & CHECK ECHO
002754  000137  006072                JMP     SINIT1          ;NO - GO ECHO MASTER'S DATA
```

```
                                        ;MASTER PROTOCOL INIT-TEST 1
                                    ;;**************************************************************
                                    MINIT1: BIT     #BIT13,@CSR
                                            BEQ     1$
                                            TYPE    ,65$                ;;TYPE ASCIZ STRING
                                            BR      64$                 ;;GET OVER THE ASCIZ
                                    ;;65$:  .ASCIZ  <CRLF>/CABLE IS NOT INSERTED - HALTING /
                                    64$:
                                            HALT
                                    1$:     JSR     PC,CPUHI
                                            MOV     #STACK,SP
                                            CLR     @CSR
                                            MOV     #1000,TIME
                                    2$:     BIT     #1000,@CSR          ;HAS SLAVE SET MASTER DSTATC
                                            BNE     MTST1
                                            DEC     TIME
                                            BNE     2$
                                    104000+15
                                            RSYNC
```

```
                                .SBTTL  * MTST1  TEST THAT SLAVE CAN ECHO THE DBR CORRECTLY
                                ;;**********************************************************************
                                ;;**********************************************************************
                                ;MASTER COMPUTER STARTS HERE FROM PROGRAM START
                                ;IT SENDS OUT SINGLE WORDS (FLOATING 1/0 PTRN) THRU THE
                                ;DBR TO THE SLAVE COMPUTER
                                ;IT LOOKS FOR THE SLAVE TO ECHO THE DBR WORD CORRECTLY
                                ;WITHIN A CERTIAN AMOUNT OF TIME
                                ;IF IT FAILS TO RETURN THE WORD AN ERROR IS REPORTED
                                ;THIS TEST IS NOT EXITED UNTIL ALL DATA PATTERNS HAVE
                                ;BEEN SENT AND RECEIVED CORRECTLY
                                ;;**********************************************************************
                                ;;**********************************************************************
003106                          MTST1:
003106  005001                  1$:     CLR     R1              ;R1 SAYS FLOAT 0 LEFT WHEN ZERO
003110  012700  177776                  MOV     #-2,R0          ;START WITH #177776 IN R0
003114  010077  176302          2$:     MOV     R0,@BDR         ;SEND PATTERN OUT
003120  004737  013502                  JSR     PC,GOCMPN       ;TELL SLAVE TO PROCEED
003124  010037  001124                  MOV     R0,$GDDAT       ;SAVE IN EXPECTED
003130  013737  001422  001122          MOV     BDR,$BDADR      ;SET UP DBR ADRS
003136  004737  013414                  JSR     PC,WTCMPN       ;WAIT FOR SLAVE TO ECHO DATA
003142  017737  176254  001126          MOV     @BDR,$BDDAT     ;READ IT BACK
003150  012737  001000  001444          MOV     #1000,TIME      ;DELAY
003156  005337  001444          3$:     DEC     TIME
003162  001375                          BNE     3$
003164  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
003172  001402                          BEQ     4$              ;BR IF SO
003174  104001                  104000+1
003176  104412                          RSYNC                   ;RSYNC ON ERROR
003200  005100                  4$:     COM     R0              ;CONVERT PTRN TO FLOATING 1
003202  005101                          COM     R1              ;TIME TO FLOAT LEFT?
003204  001343                          BNE     2$              ;BR IF NOT
003206  006300                          ASL     R0              ;YES - FLOAT LEFT
003210  005200                          INC     R0              ;KEEP LSB SET
003212  103740                          BCS     2$              ;BR IF ZERO NOT TO CARRY YET
003214  012777  177001  176200          MOV     #177001,@BDR    ;TELL SLAVE TO GO TO NEXT TEST
003222  004737  013502                  JSR     PC,GOCMPN       ;TELL SLAVE TO READ TEST TERMINATOR
                                
```

```
                                                ;MASTER PROTOCOL INIT-TEST2
                                                .SBTTL  *  TEST THAT SLAVE CAN INTERRUPT MASTER: F2(SLAVE) TO  ATTN(MASTER)
                                                ;;*****************************************************************
003226  012737  001000  001444  MINIT2: MOV     #1000,TIME      ;WAIT FOR SLAVE TO CLEAR
003234  005337  001444          1$:     DEC     TIME
003240  001375                          BNE     1$
003242  017737  176152  001446          MOV     @CSR,TEMP
003250  042737  177775  001446          BIC     #177775,TEMP    ;CLEAR ALL BUT F1
003256  013777  001446  176134          MOV     TEMP,@CSR
003264  032777  100000  176126          BIT     #100000,@CSR    ;IS ERROR BIT CLEAR
003272  001402                          BEQ     .+6
003274  104021                  104000+21
003276  104412                          RSYNC                   ;RSYNC ON ERROR
003300  012777  003462  176116          MOV     #RTRN,@DRVCTO   ;DEFINE RETURN
003306  012777  000340  176112          MOV     #340,@DRVCT2
003314  004737  013624                  JSR     PC,CPULO        ;ALLOW CPU INTERRUPT
003320  017737  176074  001446          MOV     @CSR,TEMP
003326  042737  100000  001446          BIC     #BIT15,TEMP
003334  052737  000101  001446          BIS     #101,TEMP       ;IE,GO
003342  013777  001446  176050          MOV     TEMP,@CSR
003350  032777  000002  176042          BIT     #2,@CSR         ;TELL SLAVE TO INTERRUPT MASTER
003356  001415                          BEQ     4$
003360  017737  176034  001446          MOV     @CSR,TEMP
003366  042737  000002  001446          BIC     #2,TEMP
003374  042737  100000  001446          BIC     #BIT15,TEMP
003402  013777  001446  176010          MOV     TEMP,@CSR
003410  000414                          BR      5$
003412  017737  176002  001446  4$:     MOV     @CSR,TEMP
003420  052737  000002  001446          BIS     #2,TEMP
003426  042737  100000  001446          BIC     #BIT15,TEMP
003434  013777  001446  175756          MOV     TEMP,@CSR
003442  012737  010000  001444  5$:     MOV     #10000,TIME     ;DELAY;WAIT FOR SLAVE TO INTERRUPT
003450  005337  001444          3$:     DEC     TIME
003454  001375                          BNE     3$
003456  104020                  104000+20
003460  104412                          RSYNC                   ;RESYNC ON ERROR
003462                          RTRN:
003462  004737  011200          1$:     JSR     PC,RSTVEC
003466  032777  002000  175724          BIT     #2000,@CSR      ;IS ATTN CLR?
003474  001372                          BNE     1$
003476  012777  000000  175714          MOV     #0,@CSR         ;MAKE SURE DSTATC OF SLAVE IS CLR
003504  022626                          CMP     (SP)+,(SP)+     ;READJUST STACK
003506  012737  001000  001444          MOV     #1000,TIME      ;DELAY
003514  005337  001444          2$:     DEC     TIME
003520  001375                          BNE     2$

003522  005077  175674                  CLR     @BDR
```

```
                        .SBTTL  * MTST2  TEST THAT SLAVE CAN ECHO THE 'STAT' BITS CORRECTLY
                        ;;**********************************************************************
                        ;;**********************************************************************
                        ;MASTER SENDS OUT A 'FNCT' CODE (1-7) TO THE SLAVE COMPUTER
                        ;THE MASTER THEN LOOKS FOR THE SLAVE TO ECHO THE CODE VIA THE
                        ;'STAT' BITS WITHIN A CERTIAN AMOUNT OF TIME
                        ;IF IT FAILS TO RETURN THE CORRECT CODE AN ERROR IS REPORTED
                        ;THIS TEST IS NOT EXITED UNTIL ALL 'FNCT' CODES HAVE BEEN
                        ;SENT AND RECEIVED CORRECTLY
                        ;;**********************************************************************
                        ;;**********************************************************************
003526                  MTST2:
003526  013737  001420  001122          MOV     CSR,$BDADR      ;SET UP CSR ADRS
003534  012700  000002                  MOV     #2,R0           ;SET UP INITIAL FNCT BIT COUNT
003540  012737  001202  001124          MOV     #1202,$GDDAT    ;LD EXPECETD
003546  010077  175646          1$:     MOV     R0,@CSR         ;LOAD FNCT BITS
003552  004737  013414                  JSR     PC,WTCMPN       ;LOOK FOR SLAVE TO ECHO VIA 'STAT' BITS
003556  012737  001000  001444          MOV     #1000,TIME      ;DELAY
003564  005337  001444          2$:     DEC     TIME
003570  001375                  BNE     2$
003572  017737  175622  001126          MOV     @CSR,$BDDAT     ;READ CSR
003600  033737  001124  001126          BIT     $GDDAT,$BDDAT   ;CORRECT?
003606  001002                  BNE     3$              ;BR IF SO
003610  104002                  104000+2
003612  104412                  RSYNC                   ;RSYNC ON ERROR
003614  062737  001002  001124  3$:     ADD     #1002,$GDDAT    ;ADVANCE EXPECTED
003622  062700  000002                  ADD     #2,R0           ;ADVANCE PTRN
003626  032700  000020                  BIT     #20,R0          ;ALL BEEN DONE?
003632  001745                  BEQ     1$              ;BR IF NOT
003634  012777  177002  175560          MOV     #177002,@BDR    ;TELL SLAVE TO GO TO NEXT TEST
003642  004737  013502                  JSR     PC,GOCMPN       ;GO TO COMPANION
003646  004737  013414                  JSR     PC,WTCMPN       ;WAIT FOR COMPANION
```

```
                                        .SBTTL  * MTST3 BLOCK MODE WORD XFER-MASTER XMITS 32 WORDS TO SLAVE
                                        ;;**********************************************************************
                                        ;;**********************************************************************
                                        ;MASTER XMITS A 32 WORD BLOCK OF DATA TO SLAVE
                                        ;THEN CHECKS FOR PROPER INTERRUPT STATUS, WC & BA
                                        ;THE SLAVE CHECKS THE SAME PLUS THE DATA RECEIVED
                                        ;;**********************************************************************
                                        ;;**********************************************************************
003652                                  MTST3:
003652  004737  013576          1$:     JSR     PC,CPUHI        ;NO INTR ALLOWED YET
003656  005077  175536          41$:    CLR     @CSR
003662  022777  000200  175530          CMP     #200,@CSR       ;IS ONLY READY SET
003670  001372                          BNE     41$
003672  017737  175522  001446  38$:    MOV     @CSR,TEMP
003700  042737  100000  001446          BIC     #BIT15,TEMP
003706  052737  000002  001446          BIS     #2,TEMP         ;SET DSTATC IN SLAVE
                                                                ;TELLS SLAVE TO SETUP FOR XFERS
003714  013777  001446  175476          MOV     TEMP,@CSR
003722  032777  001000  175470  40$:    BIT     #1000,@CSR      ;DSTATC HERE SAYS SLAVE
                                                                ;IS READY TO DO XFERS
003730  001774                          BEQ     40$
003732  004537  011160          37$:    JSR     R5,SETVEC       ;SET UP INTR RETURN ADRS
003736  004172                          3$                      ;RETURN TO 3$ ON INTR
003740  004537  011236                  JSR     R5,LDBUF        ;GO LOAD 'DBUF' WITH DATA PTRN
003744  177740                          -32.                    ;DO 32. LOCATIONS
003746  012737  010000  001444          MOV     #10000,TIME     ;SET UP A TIMER VALUE
003754  013777  001456  175432          MOV     XPRAM,@WCR      ;SET UP WORD COUNT
003762  013777  001460  175426          MOV     XPRAM+2,@BAR    ;SET UP BUFFER ADRS
003770  004737  013624                  JSR     PC,CPULO        ;ALLOW THE RDY INTR
003774  017737  175420  001446          MOV     @CSR,TEMP
004002  042737  100000  001446          BIC     #BIT15,TEMP
004010  052737  000010  001446          BIS     #10,TEMP        ;SINGLE CYCLE
004016  013777  001446  175374          MOV     TEMP,@CSR
004024  017737  175370  001446          MOV     @CSR,TEMP
004032  042737  100000  001446          BIC     #BIT15,TEMP
004040  042737  000002  001446          BIC     #2,TEMP
004046  013777  001446  175344          MOV     TEMP,@CSR
004054  013777  001462  175336          MOV     XPRAM+4,@CSR    ;SET UP CONTROL - IE, FNCT 3 & GO,AND CYCLE
004062  005337  001444          2$:     DEC     TIME            ;WAIT FOR INTR
004066  001375                          BNE     2$              ;UNTIL 0
004070  017737  175324  001126          MOV     @CSR,$BDDAT     ;READ CSR - SHOULD NEVER GET HERE
004076  017737  175316  001446          MOV     @CSR,TEMP
004104  042737  100000  001446          BIC     #BIT15,TEMP
004112  042737  000500  001446          BIC     #500,TEMP       ;DISABLE IE & CYCLE(IF THIS IS
                                                                ;A DRV11B: CYCLE HIGH WILL
                                                                ;PREVENT BDR FROM BEING READ)
004120  013777  001446  175272          MOV     TEMP,@CSR
004126  022737  000127  001434          CMP     #127,BOARD      ;IF DR11W:CYCLE WILL BE CLEAR
004134  001404                          BEQ     45$
004136  012737  005710  001124          MOV     #5710,$GDDAT    ;LD EXPECTED - STAT A & C, CYCLE, RDY, IE, FNCT 3
                                                                ;MUST BE DRV11B
004144  000403                          BR      46$
004146  012737  005310  001124  45$:    MOV     #5310,$GDDAT    ;STAT A&C,RDY,IE,FNCT 3
004154  013737  001420  001122  46$:    MOV     CSR,$BDADR      ;SET UP CSR ADRS
004162  104401  015516                  TYPE    ,MSG1
004166  104003                  104000+3
004170  000514                          BR      6$              ;GO RESYNC ON ERROR
```

```
                                3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
004172  022626
004174  017737  175220  001126        MOV     @CSR,$BDDAT     ;READ STATUS
004202  017737  175212  001446        MOV     @CSR,TEMP
004210  042737  100000  001446        BIC     #BIT15,TEMP
004216  042737  000500  001446        BIC     #500,TEMP       ;DISABLE IE & CYCLE(CLR CYCLE IF THIS IS A DRV11B)
004224  013777  001446  175166        MOV     TEMP,@CSR
004232  022737  000127  001434        CMP     #127,BOARD
004240  001404                        BEQ     31$
004242  012737  005710  001124        MOV     #5710,$GDDAT    ;LD EXPECTED - STAT A & C, CYCLE, RDY, IE, FNCT 3
                                                              ;MUST BE DRV11B
004250  000403                        BR      30$
004252  012737  005310  001124  31$:   MOV     #5310,$GDDAT
004260                          30$:
004260  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;CORRECT?
004266  001407                        BEQ     4$              ;BR IF SO
004270  013737  001420  001122        MOV     CSR,$BDADR      ;SET UP CSR ADRS
004276  104401  015516                TYPE    ,MSG1
004302  104004                  104000+4
004304  000446                        BR      6$              ;GO RESYNC ON ER
004306  005037  001124          4$:   CLR     $GDDAT          ;LD EXPECTED WC
004312  017737  175076  001126        MOV     @WCR,$BDDAT     ;READ WORD COUNT
004320  001407                        BEQ     5$              ;BR IF ZERO
004322  013737  001414  001122        MOV     WCR,$BDADR      ;SET UP WCR ADRS
004330  104401  015516                TYPE    ,MSG1
004334  104005                  104000+5
004336  000431                        BR      6$              ;GO RESYNC ON ER
004340  013737  001460  001124  5$:   MOV     XPRAM+2,$GDDAT  ;GET STARTING ADRS OF XFER
004346  013700  001456                MOV     XPRAM,R0        ;GET WC
004352  005400                        NEG     R0              ;GET ACTUAL #
004354  006300                        ASL     R0              ;CONVERT TO WORD
004356  060037  001124                ADD     R0,$GDDAT       ;ADD TO BASE ADRS
004362  017737  175030  001126        MOV     @BAR,$BDDAT     ;READ BAR ADRS
004370  042737  000001  001126        BIC     #BIT00,$BDDAT   ;ELIMINATE LSB
004376  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;CORRECT?
004404  001431                        BEQ     8$              ;BR IF SO
004406  013737  001416  001122        MOV     BAR,$BDADR      ;SET UP BAR ADRS
004414  104401  015516                TYPE    ,MSG1
004420  104006                  104000+6
```

```
                                                ;ERROR IN MASTER-WAIT ON SLAVE FOR ITS STATUS
                                                
004422  012737  010000  001444  6$:     MOV     #10000,TIME     ;WAIT FOR SLAVE TO COMPLETE DATA CK
004430  005337  001444          7$:     DEC     TIME            ;COUNT AWAY
004434  001375                          BNE     7$              ;UNTIL DONE
004436  000005                          RESET                   ;TELL SLAVE WE HAVE AN ERROR
004440  017737  174754  001450  21$:    MOV     @CSR,SAVE
004446  042737  170777  001450          BIC     #170777,SAVE
004454  001404                          BEQ     24$
                                                
004456  022737  001000  001450  23$:    CMP     #1000,SAVE
004464  001365                          BNE     21$
004466  104412                  24$:    RSYNC                   ;RSYNC ON ERROR
                                                
                                                ;NO ERROR IN MASTER-WAIT ON SLAVE FOR ITS STATUS
                                                
004470  012737  010000  001444  8$:     MOV     #10000,TIME     ;WAIT FOR SLAVE TO COMPLETE CKS
004476  005337  001444          20$:    DEC     TIME            ;COUNT AWAY
004502  001375                          BNE     20$             ;UNTIL DONE
004504  112777  000002  174706          MOVB    #2,@CSR         ;TELL SLAVE ALL OK
004512  017737  174702  001450  9$:     MOV     @CSR,SAVE       ;READ CSR
004520  042737  170777  001450          BIC     #170777,SAVE    ;SAVE 'STAT' BITS
004526  001405                          BEQ     11$             ;WAS THERE AN ERROR?
004530  022737  001000  001450          CMP     #1000,SAVE      ;NO ERROR?
004536  001402                          BEQ     10$
004540  000764                          BR      9$
004542  104412                  11$:    RSYNC                   ;RSYNC ON ERROR
004544  004737  011200          10$:    JSR     PC,RSTVEC       ;GO RESTORE VECTOR
```

```
                                        ;BURST MODE TESTING DETERMINATION
                                ;;**************************************************************
                                        ;DETERMINE IF BURST XFERS AND BURST DATA LATE ARE
                                        ;BE DONE.  TEST WILL BE DONE ONLY IF BOTH MASTER AND SLAVE
                                        ;ARE BOTH DR11W'S.  OTHERWISE ,SLAVE WILL BECOME MASTER
                                        ;AND MASTER SLAVE AND TESTS 1 THRU 3 WILL BE DONE AGAIN.
                                ;;**************************************************************

004550  012737  001000  001444  MSBRD:  MOV     #1000,TIME      ;WAIT FOR SLAVE
004556  005337  001444          1$:     DEC     TIME
004562  001375                          BNE     1$
004564  013777  001434  174630          MOV     BOARD,@BDR      ;TELL SLAVE WHAT BOARD
                                                                ;TYPE MASTER IS
004572  004737  013502                  JSR     PC,GOCMPN       ;TELL SLAVE TO RESPOND
004576  004737  013414                  JSR     PC,WTCMPN       ;WAIT FOR SLAVE TO EVALUATE
004602  005777  174614                  TST     @BDR            ;SLAVE RESPONDS AND MASTER CAN
                                                                ;FIND OUT IF TEST 4 SHOULD BE DONE
004606  001402                          BEQ     2$              ;0 SAYS NO
004610  000137  004620                  JMP     MTST4           ;YES;DO TEST 4
004614  000137  006072          2$:     JMP     SINIT1          ;
```

```
                                        .SBTTL  * MTST4 BURST MODE WORD XFER-MASTER XMITS 32 WORDS TO SLAVE
                                        .SBTTL  (DR11-W TO DR11-W ONLY)
                                        ;;**********************************************************************
                                        ;;**********************************************************************
                                        ;SAME AS TEST 3 EXCEPT BURST MODE:BOARD DOES NOT RELEASE BUS
                                        ;UNTIL WORD COUNT OVERFLOW
                                        ;;**********************************************************************
                                        ;;**********************************************************************
004620                                  MTST4:
004620  004737  013576          1$:     JSR     PC,CPUHI        ;NO INTR ALLOWED YET
004624  005077  174570          41$:    CLR     @CSR
004630  022777  000200  174562          CMP     #200,@CSR       ;IS ONLY READY SET
004636  001372                          BNE     41$
004640  017737  174554  001446  38$:    MOV     @CSR,TEMP
004646  042737  100000  001446          BIC     #BIT15,TEMP
004654  052737  000002  001446          BIS     #2,TEMP         ;SET DSTATC IN SLAVE
                                                                ;TELLS SLAVE TO SETUP FOR XFERS
004662  013777  001446  174530          MOV     TEMP,@CSR
004670  032777  001000  174522  40$:    BIT     #1000,@CSR      ;DSTATC HERE SAYS SLAVE
                                                                ;IS READY TO DO XFERS
004676  001774                          BEQ     40$
004700  004537  011160          37$:    JSR     R5,SETVEC       ;SET UP INTR RETURN ADRS
004704  005070                          3$                      ;RETURN TO 3$ ON INTR
004706  004537  011236                  JSR     R5,LDBUF        ;GO LOAD 'DBUF' WITH DATA PTRN
004712  177740                          -32.                    ;DO 32. LOCATIONS
004714  012737  010000  001444          MOV     #10000,TIME     ;SET UP A TIMER VALUE
004722  013777  001456  174464          MOV     XPRAM,@WCR      ;SET UP WORD COUNT
004730  013777  001460  174460          MOV     XPRAM+2,@BAR    ;SET UP BUFFER ADRS
004736  004737  013624                  JSR     PC,CPULO        ;ALLOW THE RDY INTR
004742  017737  174452  001446          MOV     @CSR,TEMP
004750  042737  100000  001446          BIC     #BIT15,TEMP
004756  042737  000002  001446          BIC     #2,TEMP
004764  013777  001446  174426          MOV     TEMP,@CSR
004772  012777  000501  174420          MOV     #501,@CSR       ;SET UP CONTROL - IE,  GO,AND CYCLE
005000  005337  001444          2$:     DEC     TIME            ;WAIT FOR INTR
005004  001375                          BNE     2$              ;UNTIL 0
005006  017737  174406  001126          MOV     @CSR,$BDDAT     ;READ CSR - SHOULD NEVER GET HERE
005014  017737  174400  001446          MOV     @CSR,TEMP
005022  042737  100000  001446          BIC     #BIT15,TEMP
005030  042737  000100  001446          BIC     #100,TEMP       ;DISABLE IE
005036  013777  001446  174354          MOV     TEMP,@CSR
005044  012737  001300  001124  45$:    MOV     #1300,$GDDAT    ;STAT C,RDY,IE
005052  013737  001420  001122  46$:    MOV     CSR,$BDADR      ;SET UP CSR ADRS
005060  104401  015550                  TYPE    ,MSG2
005064  104003                  104000+3
005066  000504                          BR      6$              ;GO RESYNC ON ERROR
005070  022626                  3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
005072  017737  174322  001126          MOV     @CSR,$BDDAT     ;READ STATUS
005100  017737  174314  001446          MOV     @CSR,TEMP
005106  042737  100000  001446          BIC     #BIT15,TEMP
005114  042737  000100  001446          BIC     #100,TEMP       ;DISABLE IE
005122  013777  001446  174270          MOV     TEMP,@CSR
005130  012737  001300  001124  31$:    MOV     #1300,$GDDAT
005136                          30$:
005136  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
005144  001407                          BEQ     4$              ;BR IF SO
005146  013737  001420  001122          MOV     CSR,$BDADR      ;SET UP CSR ADRS
```

```
1058 005154  104401  015550                    TYPE    ,MSG2
1059 005160  104004                        104000+4
1060 005162  000446                            BR      6$                ;GO RESYNC ON FR
1061 005164  005037  001124                4$: CLR     $GDDAT            ;LD EXPECTED WC
1062 005170  017737  174220  001126            MOV     @WCR,$BDDAT       ;READ WORD COUNT
1063 005176  001407                            BEQ     5$                ;BR IF ZERO
1064 005200  013737  001414  001122            MOV     WCR,$BDADR        ;SET UP WCR ADRS
1065 005206  104401  015550                    TYPE    ,MSG2
1066 005212  104005                        104000+5
1067 005214  000431                            BR      6$                ;GO RESYNC ON ER
1068 005216  013737  001460  001124  5$:       MOV     XPRAM+2,$GDDAT    ;GET STARTING ADRS OF XFER
1069 005224  013700  001456                    MOV     XPRAM,R0          ;GET WC
1070 005230  005400                            NEG     R0                ;GET ACTUAL #
1071 005232  006300                            ASL     R0                ;CONVERT TO WORD
1072 005234  060037  001124                    ADD     R0,$GDDAT         ;ADD TO BASE ADRS
1073 005240  017737  174152  001126            MOV     @BAR,$BDDAT       ;READ BAR ADRS
1074 005246  042737  000001  001126            BIC     #BIT00,$BDDAT     ;ELIMINATE LSB
1075 005254  023737  001124  001126            CMP     $GDDAT,$BDDAT     ;CORRECT?
1076 005262  001437                            BEQ     8$                ;BR IF SO
1077 005264  013737  001416  001122            MOV     BAR,$BDADR        ;SET UP BAR ADRS
1078 005272  104401  015550                    TYPE    ,MSG2
1079 005276  104006                        104000+6
1080
1081
1082
1083                                    ;ERROR IN MASTER-WAIT ON SLAVE FOR ITS STATUS
1084
1085 005300  012737  010000  001444  6$:       MOV     #10000,TIME       ;WAIT FOR SLAVE TO COMPLETE DATA CK
1086 005306  005337  001444            7$:     DEC     TIME              ;COUNT AWAY
1087 005312  001375                            BNE     7$                ;UNTIL DONE
1088 005314  000005                            RESET                     ;IF ERROR THAN INIT
1089 005316  012777  000010  174074            MOV     #10,@CSR          ;TELL SLAVE WE HAVE AN ERROR
1090 005324  017737  174070  001450  21$:      MOV     @CSR,SAVE
1091 005332  042737  170777  001450            BIC     #170777,SAVE
1092 005340  022737  005000  001450            CMP     #5000,SAVE
1093 005346  001404                            BEQ     24$
1094
1095 005350  022737  004000  001450  23$:      CMP     #4000,SAVE
1096 005356  001362                            BNE     21$
1097 005360  104412                    24$:    RSYNC                     ;RSYNC ON ERROR
1098
1099
1100
1101                                    ;NO ERROR IN MASTER-WAIT ON SLAVE FOR ITS STATUS
1102
1103 005362  012737  010000  001444  8$:       MOV     #10000,TIME       ;WAIT FOR SLAVE TO COMPLETE CKS
1104 005370  005337  001444            20$:    DEC     TIME              ;COUNT AWAY
1105 005374  001375                            BNE     20$               ;UNTIL DONE
1106 005376  012777  000012  174014            MOV     #12,@CSR          ;TELL SLAVE ALL OK
1107 005404  017737  174010  001450  9$:       MOV     @CSR,SAVE         ;READ CSR
1108 005412  042737  170777  001450            BIC     #170777,SAVE      ;SAVE 'STAT' BITS
1109 005420  022737  005000  001450            CMP     #5000,SAVE
1110 005426  001405                            BEQ     10$               ;IF NO ER,CONTINUE
1111 005430  022737  004000  001450            CMP     #4000,SAVE        ;ERROR?
1112 005436  001362                            BNE     9$
1113 005440  104412                    11$:    RSYNC                     ;RSYNC ON ERROR
1114 005442  004737  011200            10$:    JSR     PC,RSTVEC         ;GO RESTORE VECTOR
```

```
                                .SBTTL  * MTST5 BURST DATA LATE TEST-MASTER XMITS 5 WORDS TO SLAVE
                                .SBTTL    WHILE SLAVE RECEIVES 5 AND TIMES OUT (DR11-W TO DR11-W ONLY)

                                ;;**********************************************************************
                                ;;**********************************************************************
                                ;PROCEDURE: MASTER XMITS 5 WORDS TO SLAVE IN BURST MODE
                                ;SLAVE IS SETUP TO DO 10 XFERS. CHECK THAT MASTER INTERRUPTS
                                ;BY WCOF.  AFTER 5 XFERS ARE DONE IN
                                ;SLAVE,TIMEOUT OCCURS, SINCE SLAVE WAITS FOR GO-AHEAD
                                ;FROM MASTER TO CONTINUE,BUT MASTER NEVER RESPONDS. IN
                                ;SLAVE: CHECK THAT 5 XFERS HAVE BEEN COMPLETED.
                                ;;**********************************************************************
                                ;;**********************************************************************
005446                          MTST5:
005446  004737  013576        1$:     JSR     PC,CPUHI                ;NO INTR ALLOWED YET
005452  005077  173742        41$:    CLR     @CSR
005456  022777  000200  173734        CMP     #200,@CSR       ;IS ONLY READY SET
005464  001372                        BNE     41$
005466  017737  173726  001446  38$:  MOV     @CSR,TEMP
005474  042737  100000  001446        BIC     #BIT15,TEMP
005502  052737  000002  001446        BIS     #2,TEMP         ;SET DSTATC IN SLAVE
                                                              ;TELLS SLAVE TO SETUP FOR XFERS
005510  013777  001446  173702        MOV     TEMP,@CSR
005516  032777  001000  173674  40$:  BIT     #1000,@CSR      ;DSTATC HERE SAYS SLAVE
                                                              ;IS READY TO DO XFERS
005524  001774                        BEQ     40$
005526  004537  011160        37$:    JSR     R5,SETVEC       ;SET UP INTR RETURN ADRS
005532  005666                        3$                      ;RETURN TO 3$ ON INTR
005534  004537  011236                JSR     R5,LDBUF        ;GO LOAD 'DBUF' WITH DATA PTRN
005540  177740                        -32.                    ;DO 32. LOCATIONS
005542  012737  010000  001444        MOV     #10000,TIME     ;SET UP A TIMER VALUE
005550  013777  001472  173636        MOV     XMITMO,@WCR     ;SET UP WORD COUNT
005556  013777  001474  173632        MOV     XMITMO+2,@BAR   ;SET UP BUFFER ADRS
005564  004737  013624                JSR     PC,CPULO        ;ALLOW THE RDY INTR
005570  013777  001476  173622        MOV     XMITMO+4,@CSR   ;SET UP CONTROL - IE, GO,AND CYCLE

005576  005337  001444        2$:     DEC     TIME            ;WAIT FOR INTR
005602  001375                        BNE     2$              ;UNTIL 0
005604  017737  173610  001126        MOV     @CSR,$BDDAT     ;READ CSR - SHOULD NEVER GET HERE
005612  017737  173602  001446        MOV     @CSR,TEMP
005620  042737  100000  001446        BIC     #BIT15,TEMP
005626  042737  000100  001446        BIC     #100,TEMP       ;DISABLE IE
005634  013777  001446  173556        MOV     TEMP,@CSR
005642  012737  001300  001124        MOV     #1300,$GDDAT    ;LD EXPECTED  STAT C    , RDY, IE
005650  013737  001420  001122        MOV     CSR,$BDADR      ;SET UP CSR ADRS
005656  104401  015602                TYPE    ,MSG3
                                104000+3
005664  000430                        BR      6$              ;GO RESYNC ON ERROR
005666  022626                3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
005670  017737  173524  001446        MOV     @CSR,TEMP
005676  042737  100000  001446        BIC     #BIT15,TEMP
005704  042737  000100  001446        BIC     #100,TEMP       ;DISABLE IE
005712  013777  001446  173500        MOV     TEMP,@CSR
005720  017737  173470  001126        MOV     @WCR,$BDDAT     ;READ WORD COUNT
005726  001432                        BEQ     9$              ;BR IF ZERO
005730  013737  001414  001122        MOV     WCR,$BDADR      ;SET UP WCR ADRS
```

```
1173 005736  104401  015602                    TYPE    ,MSG3
1174 005742  104005                104000+5
1175 005744  000400                            BR      6$              ;GO RESYNC ON ER
1176
1177
1178
1179                                           ;ERROR IN MASTER-WAIT FOR SLAVE STATUS
1180
1181 005746  000005                6$:         RESET
1182 005750  012777  000010  173442            MOV     #10,@CSR        ;TELL SLAVE WE HAVE AN ERROR
1183 005756  017737  173436  001450  21$:      MOV     @CSR,SAVE
1184 005764  042737  170777  001450            BIC     #170777,SAVE
1185 005772  022737  005000  001450            CMP     #5000,SAVE
1186 006000  001404                            BEQ     24$
1187 006002  022737  004000  001450  23$:      CMP     #4000,SAVE
1188 006010  001362                            BNE     21$
1189 006012  104412                24$:        RSYNC                   ;RSYNC ON ERROR
1190
1191
1192
1193                                           ;NO ERROR IN MASTER-WAIT FOR SLAVE STATUS
1194
1195 006014  017737  173400  001450  9$:       MOV     @CSR,SAVE       ;READ CSR
1196 006022  042737  170777  001450            BIC     #170777,SAVE    ;SAVE 'STAT' BITS
1197 006030  022737  005000  001450            CMP     #5000,SAVE
1198 006036  001410                            BEQ     10$             ;BR IF SLAVE DONE WITH NO ER'S
1199 006040  022737  004000  001450            CMP     #4000,SAVE      ;WAS THERE AN ER?
1200 006046  001362                            BNE     9$              ;BR IF NOT
1201 006050  012777  000010  173342            MOV     #10,@CSR
1202 006056  104412                            RSYNC                   ;RSYNC ON ERROR
1203 006060  012777  000012  173332  10$:      MOV     #12,@CSR        ;TELL SLAVE ALL OK
1204 006066  004737  011200                    JSR     PC,RSTVEC       ;GO RESTORE VECTOR
```

```
1207                                    .SBTTL
1208                                    .SBTTL          SLAVE TESTS
1209                                    .SBTTL
1210
1211
1212                                            ;SLAVE PROTOCOL INIT-TEST1
1213                                    ;;******************************************************************
1214 006072  032777  020000  173320  SINIT1: BIT     #BIT13,@CSR
1215 006100  001425                          BEQ     1$
1216 006102  104401  006110                  TYPE    ,65$            ;;TYPE ASCIZ STRING
     006106  000421                          BR      64$             ;;GET OVER THE ASCIZ
                                      ;;65$:  .ASCIZ  <CRLF>/CABLE IS NOT INSERTED - HALTING /
     006152                           64$:
1217 006152  000000                          HALT
1218 006154  004737  013576          1$:     JSR     PC,CPUHI
1219 006160  012706  001100                  MOV     #STACK,SP
1220 006164  005077  173230                  CLR     @CSR
1221 006170  032777  001000  173222          BIT     #1000,@CSR      ;WHAT IS DSTATC
1222 006176  001404                          BEQ     3$              ;IF CLR-BRANCH AND WAIT FOR MASTER
1223 006200  032777  001000  173212  2$:     BIT     #1000,@CSR      ;IF NOT,WAIT FOR CLEAR
1224 006206  001374                          BNE     2$
1225 006210  004737  013502          3$:     JSR     PC,GOCMPN
1226 006214  032777  001000  173176  4$:     BIT     #1000,@CSR      ;HAS MASTER SET DSTATC IN SLAVE?
1227 006222  001774                          BEQ     4$              ;NOT YET
1228 006224  000137  006240                  JMP     SL1STR
```

```
                                        .SBTTL  * STST1  RECEIVE MASTER'S DBR DATA AND SEND IT BACK VIA DBR
                                        ;;**********************************************************************
                                        ;;**********************************************************************
                                        ;NOW THIS COMPUTER BECOMES THE SLAVE AND ECHOS MASTER'S DBR DATA
                                        ;SLAVE COMPUTER STARTS HERE FROM PROGRAM START 204
                                        ;;**********************************************************************
                                        ;;**********************************************************************

006230  004737  013502          STST1:  JSR     PC,GOCMPN       ;TELL MASTER WE ARE READY
006234  004737  013414                  JSR     PC,WTCMPN       ;DATA AVAILABLE?
                                                                ;WAIT ON IT
006240  017737  173156  001450  SL1STR: MOV     @BDR,SAVE       ;GET DATA
006246  022737  177001  001450          CMP     #177001,SAVE    ;TEST TERMINATOR?
006254  001412                        BEQ     4$              ;BR IF SO
006256  013777  001450  173136          MOV     SAVE,@BDR       ;SEND IT BACK
006264  012737  001000  001444          MOV     #1000,TIME      ;DELAY BEFORE CALLING MASTER
006272  005337  001444          5$:     DEC     TIME
006276  001375                        BNE     5$
006300  000753                        BR      STST1           ;GO LOOK FOR MORE DATA
006302  000240                  4$:     NOP                     ;
```

```
1252                                    .SBTTL  * TEST THAT SLAVE INTERRUPTS MASTER:F2 TO ATTN
1253                                            ;SLAVE PROTOCOL INIT-TEST2
1254                                    ;;**************************************************************
1255 006304                            $INIT2:
1256
1257 006304  005077  173110                       CLR     @CSR            ;CLR DSTATC OF MASTER
1258 006310  004737  013414                       JSR     PC,WTCMPN
1259 006314  017737  173100  001446              MOV     @CSR,TEMP
1260 006322  042737  100000  001446              BIC     #BIT15,TEMP
1261 006330  052737  000004  001446              BIS     #4,TEMP         ;SET INTERRUPT MASTER
1262 006336  013777  001446  173054              MOV     TEMP,@CSR
1263 006344  017737  173050  001446              MOV     @CSR,TEMP
1264 006352  042737  100000  001446              BIC     #BIT15,TEMP
1265 006360  042737  000004  001446              BIC     #4,TEMP         ;CLR F2
1266 006366  013777  001446  173024              MOV     TEMP,@CSR
1267 006374  012737  010000  001444              MOV     #10000,TIME
1268 006402  032777  001000  173010  2$:         BIT     #1000,@CSR      ;WAIT DSTATC SLAVE TO CLEAR
1269 006410  001404                             BEQ     STST2
1270 006412  005337  001444                     DEC     TIME
1271 006416  001371                             BNE     2$
1272 006420  104412                             RSYNC
```

```
                                        .SBTTL  * STST2  RECEIVE MASTER'S 'STAT' BITS AND SEND IT BACK VIA 'FNCT' BITS
                                        ;;**********************************************************************
                                        ;;**********************************************************************
                                        ;RECEIVE 'STAT' BITS AND CONVERT TO 'FNCT' BITS AND WRITE TO CSR
                                        ;;**********************************************************************
                                        ;;**********************************************************************
                                        ;;**********************************************************************
006422                                  STST2:
006422  004737  013414                  4$:     JSR     PC,WTCMPN       ;WAIT FOR MASTER
006426  012737  001000  001444                  MOV     #1000,TIME      ;DELAY
006434  005337  001444          5$:     DEC     TIME
006440  001375                          BNE     5$
006442  022777  177002  172752  2$:     CMP     #177002,@BDR    ;TEST TERMINATOR?
006450  001412                          BEQ     3$              ;BR IF SO
006452  017737  172742  001450          MOV     @CSR,SAVE       ;READ THE CSR
006460  042737  170777  001450          BIC     #170777,SAVE    ;SAVE ONLY THE STAT BITS
006466  113777  001451  172724          MOVB    SAVE+1,@CSR     ;ECHO WITH FNCT BITS
006474  000752                          BR      4$              ;LOOK FOR NEXT 'STAT' CODE
006476  004737  013502          3$:     JSR     PC,GOCMPN               ;TELL MASTER TO CONTINUE

```

```
                                        .SBTTL  * STST3 BLOCK MODE WORD XFER TEST-SLAVE RECEIVES 32
                                        .SBTTL    WORDS FROM MASTER
                                        ;;**********************************************************************
                                        ;;**********************************************************************
                                        ;THIS TEST SETS UP TO RECEIVE A 32 WORD BLOCK OF DATA FROM THE MASTER
                                        ;THEN CHECKS FOR PROPER INTERRUPT STATUS, WC, BA & DATA
                                        ;IF AN ERROR IS DETECTED THE SLAVE TELLS THE MASTER, REPORTS THE
                                        ;ERROR, AND RESYNCS ON PROGRAM
                                        ;;**********************************************************************
                                        ;;**********************************************************************
006502                                  STST3:
006502  004737  013576                1$:     JSR     PC,CPUHI        ;NO INTRS ALLOWED YET
006506  012737  001000  001444                MOV     #1000,TIME
006514  005337  001444                40$:    DEC     TIME
006520  001375                              BNE     40$
006522  005077  172672                41$:    CLR     @CSR
006526  022777  001200  172664              CMP     #1200,@CSR      ;ONLY DSTATC AND READY SET
006534  001372                              BNE     41$
006536  004537  011160                39$:    JSR     R5,SETVEC       ;SET UP THE INTR RETURN ADRS
006542  006726                              3$                      ;RETURN TO 3$ ON DATO INTR
006544  004537  011220                      JSR     R5,CLRBUF       ;GO CLR 'DBUF'
006550  177740                              -32.                    ;DO 32. LOCATIONS
006552  012737  010000  001444              MOV     #10000,TIME     ;SET UP A TIMER VALUE
006560  013777  001464  172626              MOV     RPRAM,@WCR      ;SET UP WORD COUNT
006566  013777  001466  172622              MOV     RPRAM+2,@BAR    ;SET UP BUFFER ADRS
006574  004737  013624                      JSR     PC,CPULO        ;ALLOW THE INTR
006600  017737  172614  001446              MOV     @CSR,TEMP
006606  042737  100000  001446              BIC     #BIT15,TEMP
006614  052737  000012  001446              BIS     #12,TEMP        ;FNCT 3&1
006622  013777  001446  172570              MOV     TEMP,@CSR
006630  013777  001470  172562              MOV     RPRAM+4,@CSR    ;SET UP CONTROL - FNCT 3 & 1, IE  & GO
006636  005337  001444                2$:     DEC     TIME            ;WAIT FOR INTR
006642  001375                              BNE     2$              ;UNTIL ZERO
006644  017737  172550  001126              MOV     @CSR,$BDDAT     ;READ CSR - SHOULD NEVER GET HERE
006652  017737  172542  001446              MOV     @CSR,TEMP
006660  042737  100000  001446              BIC     #BIT15,TEMP
006666  042737  000100  001446              BIC     #100,TEMP       ;DISABLE IE
006674  013777  001446  172516              MOV     TEMP,@CSR
006702  012737  004312  001124              MOV     #4312,$GDDAT    ;LD EXPECTED - STAT A, RDY, IE & FNCT 3 & 1
006710  013737  001420  001122              MOV     CSR,$BDADR      ;SET UP CSR ADRS
                                        
006716  104401  015516                      TYPE    ,MSG1
006722  104007                        104000+7
006724  000540                              BR      10$             ;GO RESYNC ON ER
006726  022626                        3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RETURN
006730  017737  172464  001126              MOV     @CSR,$BDDAT     ;READ STATUS
006736  017737  172456  001446              MOV     @CSR,TEMP
006744  042737  000100  001446              BIC     #100,TEMP       ;DISABLE IE
006752  042737  100000  001446              BIC     #BIT15,TEMP
006760  013777  001446  172432              MOV     TEMP,@CSR
006766  012737  004312  001124              MOV     #4312,$GDDAT    ;LD EXPECTED - STAT A, RDY, IE & FNCT 3 & 1
006774  023737  001124  001126              CMP     $GDDAT,$BDDAT   ;CORRECT?
007002  001407                              BEQ     4$              ;BR IF SO
007004  013737  001420  001122              MOV     CSR,$BDADR      ;SET UP CSR ADRS
                                        
007012  104401  015516                      TYPE    ,MSG1
007016  104010                        104000+10
```

```
1351 007020  000502                          BR      10$             ;GO RESYNC ON ERROR
1352 007022  005037  001124          4$:     CLR     $GDDAT          ;LD EXPECTED WC
1353 007026  017737  172362  001126          MOV     @WCR,$BDDAT     ;READ WCR
1354 007034  001407                          BEQ     5$              ;BR IF SO
1355 007036  013737  001414  001122          MOV     WCR,$BDADR      ;SET UP WCR ADRS
1356
1357 007044  104401  015516                  TYPE    ,MSG1
1358 007050  104011                  104000+11
1359 007052  000465                          BR      10$             ;GO RESYNC ON ERROR
1360 007054  013737  001466  001124  5$:     MOV     RPRAM+2,$GDDAT  ;GET STARTING ADRS OF XFER
1361 007062  013700  001464                  MOV     RPRAM,R0        ;GET WC
1362 007066  005400                          NEG     R0              ;GET ACTUAL #
1363 007070  006300                          ASL     R0              ;CONVERT TO WORD
1364 007072  060037  001124                  ADD     R0,$GDDAT       ;ADD TO BASE ADRS
1365 007076  017737  172314  001126          MOV     @BAR,$BDDAT     ;READ BAR ADRS
1366 007104  042737  000001  001126          BIC     #BIT00,$BDDAT   ;ELIMINATE LSB
1367 007112  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1368 007120  001407                          BEQ     6$              ;BR IF SO
1369 007122  013737  001416  001122          MOV     BAR,$BDADR      ;SET UP BAR ADRS
1370
1371 007130  104401  015516                  TYPE    ,MSG1
1372 007134  104012                  104000+12
1373 007136  000433                          BR      10$             ;GO RESYNC ON ERROR
1374 007140  013700  001466          6$:     MOV     RPRAM+2,R0      ;GET BUFFER ADRS
1375 007144  013701  001464                  MOV     RPRAM,R1        ;GET WORD COUNT
1376 007150  012702  177776          7$:     MOV     #177776,R2      ;GET 1ST DATA PTRN (FLOATING 0)
1377 007154  005003                          CLR     R3              ;R3 SAYS WHEN TO SHIFT PTRN
1378 007156  020220                  8$:     CMP     R2,(R0)+        ;COMPARE DATA IN DBUF TO EXPECTED
1379 007160  001011                          BNE     9$              ;BR IF DATA ERROR
1380 007162  005201                          INC     R1              ;COUNT THE WORD COUNT
1381 007164  001435                          BEQ     13$             ;BR IF DATA CHECKS DONE
1382 007166  005102                          COM     R2              ;CONVERT PTRN TO FLOATING 1
1383 007170  005103                          COM     R3              ;TIME TO SHIFT?
1384 007172  001371                          BNE     8$              ;BR IF NOT - GO CK NEXT
1385 007174  006302                          ASL     R2              ;FLOAT PTRN LEFT
1386 007176  005202                          INC     R2              ;KEEP LSB SET
1387 007200  103363                          BCC     7$              ;GC RESET FLOATING PTRN
1388 007202  000765                          BR      8$              ;GO CHECK NEXT
1389 007204  014037  001126          9$:     MOV     -(R0),$BDDAT    ;GET BAD DATA
1390 007210  010037  001122                  MOV     R0,$BDADR       ;GET MEM ADRS OF DATA ER
1391 007214  010237  001124                  MOV     R2,$GDDAT       ;LD EXPECTED DATA
1392
1393 007220  104401  015516                  TYPE    ,MSG1
1394 007224  104013                  104000+13
```

```
1396                                            ;ERROR IN SLAVE-WAIT FOR MASTER STATUS
1397
1398 007226  000005                    10$:     RESET                   ;TELL MASTER WE HAVE ERROR
1399
1400 007230  017737  172164  001450    15$:     MOV     @CSR,SAVE       ;LOOP ON ERROR
1401 007236  042737  170777  001450             BIC     #170777,SAVE    ;BUT WAIT FOR MASTER
1402 007244  001404                             BEQ     24$
1403 007246  022737  001000  001450    16$:     CMP     #1000,SAVE
1404 007254  001365                             BNE     15$
1405 007256  104412                    24$:     RSYNC                   ;RSYNC ON ERROR
1406
1407
1408
1409                                            ;NO ERROR IN SLAVE-WAIT FOR MASTER STATUS
1410
1411 007260  112777  000002  172132    13$:     MOVB    #2,@CSR         ;TELL MASTER ALL OK
1412 007266  017737  172126  001450    14$:     MOV     @CSR,SAVE       ;READ CSR
1413 007274  042737  170777  001450             BIC     #170777,SAVE    ;SAVE 'STAT' BITS ONLY
1414 007302  001405                             BEQ     31$             ;IS ER IN MASTER?
1415 007304  022737  001000  001450             CMP     #1000,SAVE      ;MASTER OK?
1416 007312  001402                             BEQ     30$
1417 007314  000764                             BR      14$
1418 007316  104412                    31$:     RSYNC                   ;RSYNC ON ERROR
1419 007320  004737  011200            30$:     JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1420
1421
1422
```

```
1424                                            ;BURST MODE WORD XFER TEST DETERMINATION
1425                                    ;;**************************************************************
1426                                            ;DETERMINE IF BURST XFERS AND BURST DATA LATE ARE TO BE
1427                                            ;DONE
1428                                    ;;**************************************************************
1429
1430 007324  004737  013414                      JSR     PC,WTCMPN       ;WAIT FOR MASTER TO INDICATE
1431                                                                     ;WHAT BOARD TYPE IT IS
1432 007330  012737  001000  001444  SLBRD:  MOV     #1000,TIME      ;WAIT ON MASTER
1433 007336  005337  001444          1$:     DEC     TIME
1434 007342  001375                          BNE     1$
1435 007344  013737  001434  001442          MOV     BOARD,TOTAL     ;DETERMINE IF BOTH MASTER
1436                                                                     ;AND SLAVE ARE DR11W'S
1437 007352  067737  172044  001442          ADD     @BDR,TOTAL
1438 007360  023727  001442  000256          CMP     TOTAL,#256
1439 007366  001406                          BEQ     2$              ;DO BURST MODE
1440
1441
1442 007370  005077  172026                  CLR     @BDR
1443 007374  004737  013502                  JSR     PC,GOCMPN
1444 007400  000137  010776                  JMP     SLVFIN          ;NO BURST MODE
1445 007404  012777  000001  172010  2$:     MOV     #1,@BDR
1446 007412  004737  013502                  JSR     PC,GOCMPN
1447
```

```
                                        .SBTTL  * STST4 BURST MODE WORD XFER TEST-SLAVE RECEIVES 32
                                        .SBTTL    WORDS FROM MASTER(DR11-W TO DR11-W ONLY)
                                        ;;**********************************************************************
                                        ;;**********************************************************************
                                        ;THIS TEST SETS UP TO RECEIVE A 32 WORD BLOCK OF DATA FROM THE MASTER
                                        ;SAME AS TEST 3 EXCEPT BURST MODE
                                        ;;**********************************************************************
                                        ;;**********************************************************************
007416                                  STST4:
007416  004737  013576                  1$:     JSR     PC,CPUHI        ;NO INTRS ALLOWED YET
007422  012737  001000  001444                  MOV     #1000,TIME
007430  005337  001444                  40$:    DEC     TIME
007434  001375                                  BNE     40$
007436  005077  171756                  41$:    CLR     @CSR
007442  022777  001200  171750                  CMP     #1200,@CSR      ;ONLY DSTATC AND READY SET
007450  001372                                  BNE     41$
007452  004537  011160                  39$:    JSR     R5,SETVEC       ;SET UP THE INTR RETURN ADRS
007456  007642                                  3$                      ;RETURN TO 3$ ON DATO INTR
007460  004537  011220                          JSR     R5,CLRBUF       ;GO CLR 'DBUF'
007464  177740                                  -32.                    ;DO 32. LOCATIONS
007466  012737  010000  001444                  MOV     #10000,TIME     ;SET UP A TIMER VALUE
007474  013777  001464  171712                  MOV     RPRAM,@WCR      ;SET UP WORD COUNT
007502  013777  001466  171706                  MOV     RPRAM+2,@BAR    ;SET UP BUFFER ADRS
007510  004737  013624                          JSR     PC,CPULO                ;ALLOW THE INTR
007514  017737  171700  001446                  MOV     @CSR,TEMP
007522  042737  100000  001446                  BIC     #BIT15,TEMP
007530  052737  000002  001446                  BIS     #2,TEMP         ;FNCT 1
007536  013777  001446  171654                  MOV     TEMP,@CSR
007544  012777  000103  171646                  MOV     #103,@CSR       ;SET UP CONTROL - FNCT  1, IE  & GO
007552  005337  001444                  2$:     DEC     TIME            ;WAIT FOR INTR
007556  001375                                  BNE     2$              ;UNTIL ZERO
007560  017737  171634  001126                  MOV     @CSR,$BDDAT     ;READ CSR - SHOULD NEVER GET HERE
007566  017737  171626  001446                  MOV     @CSR,TEMP
007574  042737  100000  001446                  BIC     #BIT15,TEMP
007602  042737  000100  001446                  BIC     #100,TEMP       ;DISABLE IE
007610  013777  001446  171602                  MOV     TEMP,@CSR
007616  012737  000302  001124                  MOV     #302,$GDDAT     ;LD EXPECTED -  RDY, IE & FNCT  1
007624  013737  001420  001122                  MOV     CSR,$BDADR      ;SET UP CSR ADRS
                                        
007632  104401  015550                          TYPE    ,MSG2
007636  104007                          104000+7
007640  000540                                  BR      10$             ;GO RESYNC ON ER
007642  022626                          3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RETURN
007644  017737  171550  001126                  MOV     @CSR,$BDDAT     ;READ STATUS
007652  017737  171542  001446                  MOV     @CSR,TEMP
007660  042737  100000  001446                  BIC     #BIT15,TEMP
007666  042737  000100  001446                  BIC     #100,TEMP       ;DISABLE IE
007674  013777  001446  171516                  MOV     TEMP,@CSR
007702  012737  000302  001124                  MOV     #302,$GDDAT     ;LD EXPECTED -  RDY, IE & FNCT  1
007710  023737  001124  001126                  CMP     $GDDAT,$BDDAT   ;CORRECT?
007716  001407                                  BEQ     4$              ;BR IF SO
007720  013737  001420  001122                  MOV     CSR,$BDADR      ;SET UP CSR ADRS
                                        
007726  104401  015550                          TYPE    ,MSG2
007732  104010                          104000+10
007734  000502                                  BR      10$             ;GO RESYNC ON ERROR
007736  005037  001124                  4$:     CLR     $GDDAT          ;LD EXPECTED WC
```

```
1506 007742  017737  171446  001126          MOV     @WCR,$BDDAT     ;READ WCR
1507 007750  001407                          BEQ     5$              ;BR IF SO
1508 007752  013737  001414  001122          MOV     WCR,$BDADR      ;SET UP WCR ADRS
1509
1510 007760  104401  015550                  TYPE    ,MSG2
1511 007764  104011                  104000+11
1512 007766  000465                          BR      10$             ;GO RESYNC ON ERROR
1513 007770  013737  001466  001124  5$:     MOV     RPRAM+2,$GDDAT  ;GET STARTING ADRS OF XFER
1514 007776  013700  001464                  MOV     RPRAM,R0        ;GET WC
1515 010002  005400                          NEG     R0              ;GET ACTUAL #
1516 010004  006300                          ASL     R0              ;CONVERT TO WORD
1517 010006  060037  001124                  ADD     R0,$GDDAT       ;ADD TO BASE ADRS
1518 010012  017737  171400  001126          MOV     @BAR,$BDDAT     ;READ BAR ADRS
1519 010020  042737  000001  001126          BIC     #BIT00,$BDDAT   ;ELIMINATE LSB
1520 010026  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1521 010034  001407                          BEQ     6$              ;BR IF SO
1522 010036  013737  001416  001122          MOV     BAR,$BDADR      ;SET UP BAR ADRS
1523
1524 010044  104401  015550                  TYPE    ,MSG2
1525 010050  104012                  104000+12
1526 010052  000433                          BR      10$             ;GO RESYNC ON ERROR
1527 010054  013700  001466          6$:     MOV     RPRAM+2,R0      ;GET BUFFER ADRS
1528 010060  013701  001464                  MOV     RPRAM,R1        ;GET WORD COUNT
1529 010064  012702  177776          7$:     MOV     #177776,R2      ;GET 1ST DATA PTRN (FLOATING 0)
1530 010070  005003                          CLR     R3              ;R3 SAYS WHEN TO SHIFT PTRN
1531 010072  020220                  8$:     CMP     R2,(R0)+        ;COMPARE DATA IN DBUF TO EXPECTED
1532 010074  001011                          BNE     9$              ;BR IF DATA ERROR
1533 010076  005201                          INC     R1              ;COUNT THE WORD COUNT
1534 010100  001443                          BEQ     13$             ;BR IF DATA CHECKS DONE
1535 010102  005102                          COM     R2              ;CONVERT PTRN TO FLOATING 1
1536 010104  005103                          COM     R3              ;TIME TO SHIFT?
1537 010106  001371                          BNE     8$              ;BR IF NOT - GO CK NEXT
1538 010110  006302                          ASL     R2              ;FLOAT PTRN LEFT
1539 010112  005202                          INC     R2              ;KEEP LSB SET
1540 010114  103363                          BCC     7$              ;GO RESET FLOATING PTRN
1541 010116  000765                          BR      8$              ;GO CHECK NEXT
1542 010120  014037  001126          9$:     MOV     -(R0),$BDDAT    ;GET BAD DATA
1543 010124  010037  001122                  MOV     R0,$BDADR       ;GET MEM ADRS OF DATA ER
1544 010130  010237  001124                  MOV     R2,$GDDAT       ;LD EXPECTED DATA
1545
1546 010134  104401  015550                  TYPE    ,MSG2
1547 010140  104013                  104000+13
```

```
1549                                                    ;ERROR IN SLAVE-WAIT FOR MASTER STATUS
1550
1551 010142  000005                              10$:   RESET
1552 010144  012777  000010  171246                     MOV     #10,@CSR        ;TELL MASTER WE HAVE AN ERROR
1553 010152  017737  171242  001450              15$:   MOV     @CSR,SAVE       ;LOOP ON ERROR
1554 010160  042737  170777  001450                     BIC     #170777,SAVE    ;BUT WAIT FOR MASTER
1555 010166  022737  005000  001450                     CMP     #5000,SAVE
1556 010174  001404                                     BEQ     24$
1557 010176  022737  004000  001450              16$:   CMP     #4000,SAVE
1558 010204  001362                                     BNE     15$
1559 010206  104412                              24$:   RSYNC                   ;RSYNC ON ERROR
1560
1561
1562
1563                                                    ;NO ERROR IN SLAVE-WAIT FOR MASTER STATUS
1564
1565 010210  012777  000012  171202              13$:   MOV     #12,@CSR        ;TELL MASTER ALL OK
1566 010216  017737  171176  001450              14$:   MOV     @CSR,SAVE       ;READ CSR
1567 010224  042737  170777  001450                     BIC     #170777,SAVE    ;SAVE 'STAT' BITS ONLY
1568 010232  022737  005000  001450                     CMP     #5000,SAVE
1569 010240  001405                                     BEQ     30$             ;IF NO ER,CONTINUE
1570 010242  022737  004000  001450                     CMP     #4000,SAVE      ;IF ERROR,RESYNC
1571 010250  001362                                     BNE     14$
1572 010252  104412                              31$:   RSYNC                   ;RSYNC ON ERROR
1573 010254  004737  011200                      30$:   JSR     PC,RSTVEC       ;GO RESTORE VECTOR
```

```
                                        .SBTTL  * STST5 BURST DATA LATE TEST-SLAVE RECEIVES 5 WORDS
                                        .SBTTL    FROM MASTER AND TIMES OUT WHILE EXPECTING MORE
                                        .SBTTL     (DR11-W TO DR11-W ONLY)
                                        ;;**************************************************************
                                        ;;**************************************************************
                                        ;SETS UP TO RECEIVE 10 WORDS IN BURST MODE FROM MASTER,BUT MASTER WILL ONLY SEND 5 WORDS.
                                        ; TIMEOUT OCCURS AND THE BUS IS RELEASED
                                        ;THEN CHECKS FOR PROPER INTERRUPT STATUS, WC, BA & DATA
                                        ;IF AN ERROR IS DETECTED THE SLAVE TELLS THE MASTER, REPORTS THE
                                        ;ERROR, AND RESYNCS ON PROGRAM
                                        ;IF ALL OK THEN THIS COMPUTER BECOMES MASTER AND GOES TO MTST1
                                        ;;**************************************************************
                                        ;;**************************************************************
010260                                  STST5:
010260  004737  013576                  1$:     JSR     PC,CPUHI        ;NO INTRS ALLOWED YET
010264  012737  001000  001444                  MOV     #1000,TIME      ;DELAY ON MASTER
010272  005337  001444                  40$:    DEC     TIME
010276  001375                          41$:    BNE     40$
010300  005077  171114                          CLR     @CSR
010304  022777  001200  171106                  CMP     #1200,@CSR      ;ONLY DSTATC AND READY
010312  001372                                  BNE     41$
010314  004537  011160                  39$:    JSR     R5,SETVEC       ;SET UP THE INTR RETURN ADRS
010320  010424                                  2$                      ;RETURN TO 2$ ON DATO INTR
010322  004537  011220                          JSR     R5,CLRBUF       ;GO CLR 'DBUF'
010326  177740                                  -32.                    ;DO 32. LOCATIONS
010330  012737  040000  001444                  MOV     #40000,TIME     ;SET UP A TIMER VALUE
010336  013777  001500  171050                  MOV     RCVTMO,@WCR     ;SET UP WORD COUNT
010344  013777  001502  171044                  MOV     RCVTMO+2,@BAR   ;SET UP BUFFER ADRS
010352  004737  013624                          JSR     PC,CPULO        ;ALLOW THE INTR
010356  017737  171036  001446                  MOV     @CSR,TEMP
010364  042737  100000  001446                  BIC     #BIT15,TEMP
010372  052737  000002  001446                  BIS     #2,TEMP         ;FNCT1
010400  013777  001446  171012                  MOV     TEMP,@CSR
010406  013777  001504  171004                  MOV     RCVTMO+4,@CSR   ;SET UP CONTROL:IE,FNCT1 & GO
                                                ;TIMEOUT
010414  005337  001444          25$:    DEC     TIME            ;WAIT FOR MASTER TO SETUP;
010420  001375                          BNE     25$             ;THEN SLAVE BUS WILL BE HELD FOR DURATION
                                                                ;OF BURST XFERS;TIMEOUT WILL OCCUR WITH SUBSEQUENT
                                                                ;RELEASE OF BUS AND PROGRAM CONTINUATION
010422  000416                          BR      3$              ;GO TO CHECKS
010424  017737  170770  001446  2$:     MOV     @CSR,TEMP
010432  042737  100000  001446          BIC     #BIT15,TEMP
010440  042737  000100  001446          BIC     #100,TEMP       ;DISABLE IE
010446  013777  001446  170744          MOV     TEMP,@CSR
010454  104017                  104000+17
010456  000471                          BR      10$
010460  017737  170734  001126  3$:     MOV     @CSR,$BDDAT     ;READ STATUS
010466  032737  000200  001126          BIT     #BIT7,$BDDAT    ;TEST READY BIT
010474  001412                          BEQ     5$
010476  013737  001446  001124          MOV     TEMP,$GDDAT
010504  013737  001420  001122          MOV     CSR,$BDADR
010512  104401  015602                  TYPE    ,MSG3
010516  104014                  104000+14
010520  000450                          BR      10$
010522  012777  100000  170670  5$:     MOV     #BIT15,@CSR     ;GO TO EIR
010530  017737  170664  001126          MOV     @CSR,$BDDAT     ;SAVE EIR
010536  032737  001000  001126          BIT     #BIT9,$BDDAT    ;TEST BURST DATA LATE BIT
```

```
1632 010544  001015                          BNE     4$
1633 010546  042737  000400  001126          BIC     #BIT8,$BDDAT
1634 010554  012737  101000  001124          MOV     #101000,$GDDAT
1635 010562  013737  001420  001122          MOV     CSR,$BDADR
1636 010570  104401  015602                  TYPE    ,MSG3
1637 010574  104022                  104000+22
1638 010576  000421                          BR      10$             ;GO RESYNC ON ERROR
1639 010600  012737  177773  001124  4$:     MOV     #177773,$GDDAT          ;LD EXPECTED WC
1640 010606  017737  170602  001126          MOV     @WCR,$BDDAT     ;READ WCR
1641 010614  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1642 010622  001432                          BEQ     13$             ;BR IF SO
1643 010624  013737  001414  001122          MOV     WCR,$BDADR      ;SET UP WCR ADRS
1644 010632  104401  015602                  TYPE    ,MSG3
1645 010636  104011                  104000+11
1646 010640  000400                          BR      10$             ;GO RESYNC ON ERROR
1647
1648
1649
1650                                         ;ERROR IN SLAVE-WAIT FOR MASTER STATUS
1651
1652 010642  000005                  10$:    RESET                   ;IF ERROR,INIT
1653 010644  012777  000010  170546          MOV     #10,@CSR
1654 010652  017737  170542  001450  15$:    MOV     @CSR,SAVE       ;LOOP ON ERROR
1655 010660  042737  170777  001450          BIC     #170777,SAVE    ;BUT WAIT FOR MASTER
1656 010666  022737  005000  001450          CMP     #5000,SAVE
1657 010674  001404                          BEQ     24$
1658 010676  022737  004000  001450  16$:    CMP     #4000,SAVE
1659 010704  001362                          BNE     15$
1660 010706  104412                  24$:    RSYNC                   ;RSYNC ON ERROR
1661
1662
1663
1664                                         ;NO ERROR IN SLAVE-WAIT FOR MASTER STATUS
1665
1666 010710  000005                  13$:    RESET                   ;SLAVE HAS TIMED OUT LEAVING
1667                                                                 ;READY CLEAR. FNCT BITS TO DSTAT
1668                                                                 ;BITS COMMUNICATION CAN BE
1669                                                                 ;RESUMED BY SETTING READY.RESET
1670                                                                 ;WILL SET READY.
1671 010712  012737  010000  001444          MOV #10000,TIME
1672 010720  005337  001444          11$:    DEC TIME
1673 010724  001375                          BNE 11$
1674 010726  012777  000012  170464          MOV     #12,@CSR        ;TELL MASTER ALL OK
1675 010734  017737  170460  001450  14$:    MOV     @CSR,SAVE       ;READ CSR
1676 010742  042737  170777  001450          BIC     #170777,SAVE    ;SAVE 'STAT' BITS ONLY
1677 010750  022737  005000  001450          CMP     #5000,SAVE
1678 010756  001405                          BEQ     EOPT            ;BR IF MASTER DONE WITH NO ER'S
1679 010760  022737  004000  001450          CMP     #4000,SAVE      ;DOES MASTER HAVE AN ERROR?
1680 010766  001362                          BNE     14$             ;BR IF NOT
1681 010770  104412                          RSYNC                   ;RSYNC ON ERROR
1682 010772  004737  011200          EOPT:   JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1683 010776  104407          SLVFIN: CKSWR                   ;GO CHECK SWR
1684 011000  005737  001452                  TST     MSTER           ;WERE WE STARTED AS MASTER?
1685 011004  001055                          BNE     EOPTA           ;BR IF NOT
1686 011006  005337  001454                  DEC     ICOUNT          ;COUNT TEST PASS
1687 011012  001052                          BNE     EOPTA           ;BR IF NOT DUE FOR 'END PASS' MSG
1688 011014  012737  000144  001454          MOV     #144,ICOUNT     ;RESET PASS COUNT TO  100 ITERATIONS
```

```
                                .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 EOPTA
011022                          $EOP:
011022  000240                          NOP
011024  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
011030  005237  001100                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
011034  042737  100000  001100          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
011042  005327                          DEC     (PC)+           ;;LOOP?
011044  000001                  $EOPCT: .WORD   1
011046  003022                          BGT     $DOAGN          ;;YES
011050  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
011052  000001                  $ENDCT: .WORD   1
011054  011044                          $EOPCT
011056  104401  011123                  TYPE    ,$ENDMG         ;;TYPE "END PASS #"
011062  013746  001100                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
011066  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
011070  104401  011120                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
011074  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
011100  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
011102  000005                          RESET                   ;;CLEAR THE WORLD
011104  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
011106  000240                          NOP                     ;;SAVE ROOM
011110  000240                          NOP                     ;;FOR
011112  000240                          NOP                     ;;ACT11
011114                          $DOAGN:
011114  000137                          JMP     @(PC)+          ;;RETURN
011116  011140                  $RTNAD: .WORD   EOPTA
011120     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
011123     015     012     105  $ENDMG: .ASCIZ  <15><12>/END PASS #/
011126     116     104     040
011131     120     101     123
011134     123     040     043
011137     000
1691 011140  012737  001000  001444  EOPTA:  MOV     #1000,TIME      ;SET UP A COUNT
1692 011146  005337  001444          EOPTB:  DEC     TIME            ;STALL FOR OTHER CPU TO BECOME SLAVE
1693 011152  001375                          BNE     EOPTB           ;UNTIL TIME=0
1694 011154  000137  002760                  JMP     MINIT1          ;NOW BECOME MASTER
```

```
1696                                    .SBTTL
1697                                    .SBTTL  PROGRAM SUBROUTINES
1698
1699                                    ;;**********************************************************************
1700                                    ;THIS ROUTINE SETS THE PRIORITY LEVEL FOR NO INTERRUPT -
1701                                    ;SETS UP THE  INTERRUPT TO RETURN ON INTERRUPT
1702                                    ;TO THE ADDRESS INDICATED ((R5)) BY THE CALL +2
1703                                    ;;**********************************************************************
1704 011160  004737  013576            SETVEC: JSR     PC,CPUHI        ;SET UP FOR NO INTERRUPT
1705 011164  012577  170234                    MOV     (R5)+,@DRVCT0   ;SET UP INTR RETURN ADRS
1706 011170  012777  000200  170230            MOV     #200,@DRVCT2    ;KEEP PRIORITY LEVEL AT TOP ON INTR
1707 011176  000205                            RTS     R5              ;EXIT
1708
1709                                    ;;**********************************************************************
1710                                    ;THIS ROUTINE RESTORES THE  INTERRUPT VECTOR TO A HALT
1711                                    ;AND RAISES THE PRIORITY LEVEL
1712                                    ;;**********************************************************************
1713 011200  013777  001426  170216    RSTVEC: MOV     DRVCT2,@DRVCT0  ;POINT VECTOR TO HALT
1714 011206  005077  170214                    CLR     @DRVCT2         ;SET UP HALT
1715 011212  004737  013576                    JSR     PC,CPUHI        ;RAISE PRIORITY LEVEL
1716 011216  000207                            RTS     PC              ;EXIT
1717
1718                                    ;;**********************************************************************
1719                                    ;THIS ROUTINE CLEARS THE 'DBUF' BEFORE A 'DATI' XFER
1720                                    ;THE # OF LOCATIONS IN 'DBUF' TO BE CLEARED IS SPECIFIED BY
1721                                    ;THE VALUE IN THE CALL +2 - WHEN ALL CLEARED THE RETURN IS TO
1722                                    ;THE CALL +4
1723                                    ;;**********************************************************************
1724 011220  012500                    CLRBUF: MOV     (R5)+,R0        ;GET THE LOC COUNT
1725 011222  012701  015756                    MOV     #DBUF,R1        ;GET 1ST ADRS
1726 011226  005021                    1$:     CLR     (R1)+           ;CLR MEM LOC
1727 011230  005200                            INC     R0              ;COUNT LOC
1728 011232  001375                            BNE     1$              ;UNTIL ALL DONE
1729 011234  000205                            RTS     R5              ;EXIT
1730
```

```
 1732                                   ;;*****************************************************************
 1733                                   ;THIS ROUTINE LOADS 'DBUF' WITH A FLOATING ZERO/ONE PATTERN
 1734                                   ;THE # OF LOCATIONS IN 'DBUF' TO BE LOADED IS SPECIFIED BY
 1735                                   ;THE VALUE IN THE CALL +2 - WHEN ALL LOADED THE RETURN IS TO CALL +4
 1736                                   ;;*****************************************************************
 1737 011236  012500                   LDBUF:  MOV     (R5)+,R0        ;GET LOC COUNT
 1738 011240  012701  015756                   MOV     #DBUF,R1        ;GET 1ST ADRS
 1739 011244  012702  177776           1$:     MOV     #177776,R2      ;SET UP FLOATING ZERO PTRN
 1740 011250  005003                           CLR     R3              ;R3 SAYS WHEN TO SHIFT PTRN
 1741 011252  010221                   2$:     MOV     R2,(R1)+        ;LOAD MEM WITH PTRN
 1742 011254  005200                           INC     R0              ;COUNT LOC
 1743 011256  001001                           BNE     3$              ;BR IF MORE
 1744 011260  000205                           RTS     R5              ;EXIT
 1745 011262  005102                   3$:     COM     R2              ;CONVERT PTRN TO FLOATING 1
 1746 011264  005103                           COM     R3              ;SHOULD WE SHIFT?
 1747 011266  001371                           BNE     2$              ;BR IF NOT - THIS WILL BE A FLOATING 1
 1748 011270  006302                           ASL     R2              ;FLOAT ZERO LEFT
 1749 011272  005202                           INC     R2              ;KEEP LSB SET
 1750 011274  103363                           BCC     1$              ;GO RESET FLOATING PATRN
 1751 011276  000765                           BR      2$              ;GO LOAD NEXT PATRN
```

```
                                        .SBTTL  SYSMAC ROUTINES


                                .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
                                ;*
011300  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
011304  100002                          BPL     1$              ;;BR IF YES
011306  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
011310  000407                          BR      3$              ;;LEAVE
011312  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
011314  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
011320  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
011322  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
011324  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
011326  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
011330  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
011334  000002                          RTI                     ;;RETURN
011336  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
011342  001430                          BEQ     8$
011344  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
011350  001006                          BNE     5$
011352  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
011354  104401                          TYPE                    ;;TYPE A CR AND LF
011356  001165                          $CRLF
011360  105037  011514                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
011364  000755                          BR      2$              ;;GET NEXT CHARACTER
011366  004737  011450          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
011372  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
011376  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
011400  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
011404  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
011410  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
011412  004737  011450                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
011416  105337  011514                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
011422  000770                          BR      7$              ;;LOOP
                                ;HORIZONTAL TAB PROCESSOR
011424  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
011430  004737  011450          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
011434  132737  000007  011514          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
011442  001372                          BNE     9$              ;;TAB STOP
011444  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
011446  000724                          BR      2$              ;;GET NEXT CHARACTER
011450  105777  167474          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
011454  100375                          BPL     $TYPEC
```

```
011456  116677  000002  167466          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
011464  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
011472  001003                          BNE     1$              ;;BRANCH IF NO
011474  105037  011514                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
011500  000406                          BR      $TYPEX          ;;EXIT
011502  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
011510  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
011512  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
011514  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
011516  000207                  $TYPEX: RTS     PC
```

```
1757                                    .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
011520  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
011524  116637  000001  011743          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
011532  112637  011745                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
011536  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
011542  000406                          BR      $TYPON
011544  112737  000001  011743  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
011552  112737  000006  011745          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
011560  112737  000005  011742  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
011566  010346                          MOV     R3,-(SP)        ;;SAVE R3
011570  010446                          MOV     R4,-(SP)        ;;SAVE R4
011572  010546                          MOV     R5,-(SP)        ;;SAVE R5
011574  113704  011745                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
011600  005404                          NEG     R4
011602  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
011606  110437  011744                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
011612  113704  011743                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
011616  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
011622  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
011624  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
011626  000404                          BR      3$              ;;GO DO MSB
011630  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
011632  006105                          ROL     R5
011634  006105                          ROL     R5
011636  010503                          MOV     R5,R3
011640  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
011642  105337  011744                  DECB    $OMODE          ;;TYPE THIS DIGIT?
011646  100016                          BPL     7$              ;;BR IF NO
011650  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
011654  001002                          BNE     4$              ;;TEST FOR 0
011656  005704                          TST     R4              ;;SUPPRESS THIS 0?
011660  001403                          BEQ     5$              ;;BR IF YES
011662  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
011664  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
011670  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
```

```
011674  110337  011740                  MOVB    R3,8$           ;;SAVE FOR TYPING
011700  104401  011740                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
011704  105337  011742          7$:     DECB    $OCNT           ;;COUNT BY 1
011710  003347                          BGT     2$              ;;BR IF MORE TO DO
011712  002402                          BLT     6$              ;;BR IF DONE
011714  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
011716  000744                          BR      2$              ;;GO DO THE LAST DIGIT
011720  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
011722  012604                          MOV     (SP)+,R4        ;;RESTORE R4
011724  012603                          MOV     (SP)+,R3        ;;RESTORE R3
011726  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
011734  012616                          MOV     (SP)+,(SP)
011736  000002                          RTI                     ;;RETURN
011740     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
011741     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
011742     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
011743     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
011744  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
1759                                    .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
011746                                  $TYPDS:
011746  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011750  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
011752  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
011754  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
011756  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
011760  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
011764  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
011770  100004                          BPL     1$              ;;BR IF INPUT IS POS.
011772  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
011774  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
012002  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
012004  012703  012162                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
012010  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
012014  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
012016  016001  012152                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
012022  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
012024  002402                          BLT     4$              ;;BR IF DONE
012026  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
012030  000774                          BR      3$
012032  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
012034  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
012036  001002                          BNE     5$              ;;FALL THROUGH IF 0
012040  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
012042  100407                          BMI     7$              ;;BR IF YES
012044  106316                  5$:     ASLB    (SP)            ;;MSD?
012046  103003                          BCC     6$              ;;BR IF NO
012050  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
012056  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
012062  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
012066  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
012070  005720                          TST     (R0)+           ;;JUST INCREMENTING
012072  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
012076  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
012100  003002                          BGT     8$              ;;GO TO EXIT
012102  010502                          MOV     R5,R2           ;;GET THE LSD
012104  000764                          BR      6$              ;;GO CHANGE TO ASCII
012106  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
012110  100003                          BPL     9$              ;;BR IF NO
012112  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
012120  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
012122  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
012124  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
012126  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
012130  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
012132  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
012134  104401  012162                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
```

```
012140  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
012146  012616                          MOV     (SP)+,(SP)
012150  000002                          RTI                     ;;RETURN TO USER
012152  023420                  $DTBL:  10000.
012154  001750                          1000.
012156  000144                          100.
012160  000012                          10.
012162                          $DBLK:  .BLKW   4
```

```
 1761                                   .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 SWRCK 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
                                        ;*CALL
                                        ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
     012172                             $ERROR:
     012172  104407                             CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
     012174  104407                             CKSWR                   ;GO LOOK FOR SWR CHANGE
     012176  105237  001103             7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
     012202  001775                             BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
     012204  013777  001102  166730             MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
     012212  032777  002000  166720             BIT     #BIT10,@SWR     ;;BELL ON ERROR?
     012220  001402                             BEQ     1$              ;;NO - SKIP
     012222  104401  001160                     TYPE    ,$BELL          ;;RING BELL
     012226  005237  001112             1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
     012232  011637  001116                     MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
     012236  162737  000002  001116             SUB     #2,$ERRPC
     012244  117737  166646  001114             MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
     012252  032777  020000  166660             BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
     012260  001004                             BNE     20$             ;;SKIP TYPEOUTS
     012262  004737  012306                     JSR     PC,SWRCK        ;;GO TO USER ERROR ROUTINE
     012266  104401  001165                     TYPE    ,$CRLF
     012272                             20$:
     012272  005777  166642             2$:     TST     @SWR            ;;HALT ON ERROR
     012276  100002                             BPL     3$              ;;SKIP IF CONTINUE
     012300  000000                             HALT                    ;;HALT ON ERROR!
     012302  104407                             CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
     012304                             3$:
     012304  000002                             RTI                     ;;RETURN
 1762                                   ;;*****************************************************************
 1763                                   ;GO TYPE ERROR
 1764                                   ;GO UPDATE SOFTWARE SWR IF 'CNTRL/G'
 1765                                   ;;*****************************************************************
 1766 012306  004737  012316  SWRCK:  JSR     PC,$ERRTYP      ;GO TYPE ERROR
 1767 012312  104407                   CKSWR                   ;GO LOOK FOR SWR CHANGE
 1768 012314  000207                   RTS     PC              ;RETURN TO ERROR HANDLER
```

```
 1770                                   .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.
012316                                  $ERRTYP:
012316  104401  001165                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012322  010046                                MOV     R0,-(SP)        ;;SAVE R0
012324  005000                                CLR     R0              ;;PICKUP THE ITEM INDEX
012326  153700  001114                        BISB    @#$ITEMB,R0
012332  001004                                BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                        ;;TYPE THE PC OF THE ERROR
012334  013746  001116                        MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;ERROR ADDRESS
012340  104402                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012342  000426                                BR      6$              ;;GET OUT
012344  005300                          1$:   DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
012346  006300                                ASL     R0              ;;      WORK FOR THE ERROR TABLE
012350  006300                                ASL     R0
012352  006300                                ASL     R0
012354  062700  001170                        ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
012360  012037  012370                        MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
012364  001404                                BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
012366  104401                                TYPE                    ;;TYPE THE "ERROR MESSAGE"
012370  000000                          2$:   .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
012372  104401  001165                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012376  012037  012406                  3$:   MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
012402  001404                                BEQ     5$              ;;SKIP TYPEOUT IF 0
012404  104401                                TYPE                    ;;TYPE THE "DATA HEADER"
012406  000000                          4$:   .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
012410  104401  001165                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012414  011000                          5$:   MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
012416  001004                                BNE     7$              ;;GO TYPE THE DATA
012420  012600                          6$:   MOV     (SP)+,R0        ;;RESTORE R0
012422  104401  001165                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012426  000207                                RTS     PC              ;;RETURN
012430                                  7$:
012430  013046                                MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
012432  104402                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012434  005710                                TST     (R0)            ;;IS THERE ANOTHER NUMBER?
012436  001770                                BEQ     6$              ;;BR IF NO
012440  104401  012446                        TYPE    ,8$             ;;TYPE TWO(2) SPACES
012444  000771                                BR      7$              ;;LOOP
012446     040     040     000          8$:   .ASCIZ  /  /            ;;TWO(2) SPACES
                                              .EVEN
```

```
 1772                                   .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>
                                        ;*CALL
                                        ;*      SCOPE           ;;SCOPE=IOT
 012452                                 $SCOPE:
 012452  104407                                CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
 012454  104407                                CKSWR                   ;GO LOOK FOR SWR CHANGE
                                        ;#####START OF  CODE FOR THE XOR TESTER#####
 012456  000416                         $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
 012460  013746  000004                         MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 012464  012737  012504  000004                 MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
 012472  005737  177060                         TST     @#177060        ;;TIME OUT ON XOR?
 012476  012637  000004                         MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 012502  000404                                 BR      $SVLAD          ;;GO TO THE NEXT TEST
 012504  022626                         5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
 012506  012637  000004                         MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 012512  000406                                 BR      $OVER           ;;LOOP ON THE PRESENT TEST
 012514                                 6$:;#####END OF  CODE FOR THE XOR TESTER#####
 012514  105237  001102                $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
 012520  011637  001106                         MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 012524  105037  001103                         CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
 012530  013777  001102  166404         $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 012536  013716  001106                         MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 012542  000002                                 RTI                     ;;FIXES PS
```

```
 1774                                   .SBTTL  TTY INPUT ROUTINE
                                        ;;*****************************************************************
                                        .ENABL  LSB
                                        ;;*****************************************************************
                                        ;*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.
012544  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
012552  001074                         BNE     15$             ;;BRANCH IF NO
012554  105777  166364                 TSTB    @$TKS           ;;CHAR THERE?
012560  100071                         BPL     15$             ;;IF NO, DON'T WAIT AROUND
012562  117746  166360                 MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
012566  042716  177600                 BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
012572  022726  000007                 CMP     #7,(SP)+        ;;IS IT A CONTROL G?
012576  001062                         BNE     15$             ;;NO, RETURN TO USER
012600  123727  001134  000001         CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
012606  001456                         BEQ     15$             ;;BRANCH IF YES
012610  104401  013271                 TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
012614  104401  013276         $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
012620  013746  000176                 MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
012624  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012626  104401  013307                 TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
012632  005046                 19$:    CLR     -(SP)           ;;CLEAR COUNTER
012634  005046                         CLR     -(SP)           ;;THE NEW SWR
012636  105777  166302         7$:     TSTB    @$TKS           ;;CHAR THERE?
012642  100375                         BPL     7$              ;;IF NOT TRY AGAIN
012644  117746  166276                 MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
012650  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
012654  021627  000025         9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
012660  001005                         BNE     10$             ;;BRANCH IF NOT
012662  104401  013264                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
012666  062706  000006         20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
012672  000757                         BR      19$             ;;LET'S TRY IT AGAIN
012674  021627  000015         10$:    CMP     (SP),#15        ;;IS IT A <CR>?
012700  001022                         BNE     16$             ;;BRANCH IF NO
012702  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
012706  001403                         BEQ     11$             ;;BRANCH IF YES
012710  016677  000002  166222         MOV     2(SP),@SWR      ;;SAVE NEW SWR
012716  062706  000006         11$:    ADD     #6,SP           ;;CLEAR UP STACK
012722  104401  001165         14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
012726  123727  001135  000001         CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
012734  001003                         BNE     15$             ;;BRANCH IF NOT
012736  012777  000100  166200         MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
012744  000002                 15$:    RTI                     ;;RETURN
012746  004737  011450         16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
012752  021627  000060                 CMP     (SP),#60        ;;CHAR < 0?
012756  002420                         BLT     18$             ;;BRANCH IF YES
012760  021627  000067                 CMP     (SP),#67        ;;CHAR > 7?
012764  003015                         BGT     18$             ;;BRANCH IF YES
012766  042726  000060                 BIC     #60,(SP)+       ;;STRIP-OFF ASCII
012772  005766  000002                 TST     2(SP)           ;;IS THIS THE FIRST CHAR
012776  001403                         BEQ     17$             ;;BRANCH IF YES
013000  006316                         ASL     (SP)            ;;NO, SHIFT PRESENT
013002  006316                         ASL     (SP)            ;;   CHAR OVER TO MAKE
013004  006316                         ASL     (SP)            ;;   ROOM FOR NEW ONE.
013006  005266  000002         17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
```

```
013012  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
013016  000707                          BR      7$              ;;GET THE NEXT ONE
013020  104401  001164          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
013024  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
                                ;
013026  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
013030  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
013036  105777  166102          1$:     TSTB    @$TKS           ;;WAIT FOR
013042  100375                          BPL     1$              ;;A CHARACTER
013044  117766  166076  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
013052  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
013060  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
013066  001013                          BNE     3$              ;;BRANCH IF NO
013070  105777  166050          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
013074  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
013076  117746  166044                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
013102  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
013106  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
013112  001366                          BNE     2$              ;;IF NOT DISCARD IT
013114  000750                          BR      1$              ;;YES, RESUME
013116  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
013124  002407                          BLT     4$              ;;BRANCH IF YES
013126  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
013134  003003                          BGT     4$              ;;BRANCH IF YES
013136  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
013144  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
013146  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
013150  012703  013254          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
013154  022703  013264          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
013160  101405                          BLOS    4$              ;;BR IF YES
013162  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
013164  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
013166  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
013172  001003                          BNE     3$              ;;SKIP IF NOT
013174  104401  001164          4$:     TYPE    ,$QUES          ;;TYPE A '?'
013200  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
013202  111337  013252          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
013206  104401  013252                  TYPE    ,9$
013212  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
013216  001356                          BNE     2$              ;;LOOP IF NOT RETURN
013220  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
013224  104401  001166                  TYPE    ,$LF            ;;TYPE A LINE FEED
013230  012603                          MOV     (SP)+,R3        ;;RESTORE R3
013232  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
```

```
013234  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
013242  012766  013254  000004          MOV     #$TTYIN,4(SP)
013250  000002                          RTI                     ;;RETURN
013252     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
013253     000                          .BYTE   0               ;;TERMINATOR
013254                          $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
013264     136     125     015  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
013267     012     000
013271     136     107     015  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
013274     012     000
013276     015     012     123  $MSWR:  .ASCIZ  <15><12>/SWR = /
013301     127     122     040
013304     075     040     000
013307     040     040     116  $MNEW:  .ASCIZ  /  NEW = /
013312     105     127     040
013315     075     040     000
```

```
                                .EVEN
                                .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.
013320  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
013322  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
013326  005740                          TST     -(R0)           ;;BACKUP BY 2
013330  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
013332  022700  000026                  CMP     #$TERM,R0       ;;CHECK FOR OUT OF BOUNDS
013336  003002                          BGT     .+6             ;;BR IF OK
013340  000000                          HALT                    ;;OUT OF BOUNDS
013342  000776                          BR      .-2             ;;HANGUP
013344  006300                          ASL     R0              ;;POSITION FOR INDEXING
013346  016000  013366                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
013352  000200                          RTS     R0              ;;GO TO ROUTINE
                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
013354  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
013356  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
013364  000002                          RTI                     ;;RESTORE THE PSW
                                .SBTTL  TRAP TABLE
                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE "TRAP" INSTRUCTION.
                                ;       ROUTINE
                                ;       -------
013366  013354                  $TRPAD: .WORD   $TRAP2
013370  011300                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
013372  011544                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
013374  011520                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
013376  011560                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
013400  011746                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
013402  012614                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
013404  012544                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
013406  013026                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
013410  013146                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
013412  013652                          RESYNC  ;;CALL=RSYNC    TRAP+12(104412) RESYNC PROGRAM
000026                  $TERM=.-$TRPAD
```

```
                        ;;*****************************************************************
                        .SBTTL  INTERBOARD COMMUNICATION SUBROUTINES
                        ;WTCMPN
                        ;;*****************************************************************

013414  012737  010000  001436  WTCMPN: MOV     #10000,ABORT    ;SETUP ABORT TIMER
013422  032777  001000  165770          BIT     #1000,@CSR
013430  001010                        BNE     2$              ;IF DSTATC IS ONE,WAIT FOR 0

013432  032777  001000  165760  1$:     BIT     #1000,@CSR      ;WAIT FOR DSTATC=1
013440  001017                        BNE     4$              ;GO AHEAD;IT HAS COME
013442  005337  001436                DEC     ABORT
013446  001371                        BNE     1$              ;IF NOT STUCK IN WAIT LOOP,
                                                              ;KEEP WAITING
013450  000407                        BR      3$              ;WAITED TOO LONG;ABORT PROGRAM
013452  032777  001000  165740  2$:     BIT     #1000,@CSR      ;WAIT DSTATC=0
013460  001407                        BEQ     4$              ;GO AHEAD;IT HAS COME
013462  005337  001436                DEC     ABORT           ;KEEP WAITING
013466  001371                        BNE     2$

013470  011637  001440          3$:     MOV     (SP),TESTPC     ;SAVE LOCATION WHERE PROGRAM
                                                              ;ORIGINATED
013474  104016                  104000+16
013476  104412                        RSYNC                   ;RESYNCHRONIZE BOTH PROGRAMS
                                                              ;FROM START
013500  000207                  4$:     RTS     PC

                        ;;*****************************************************************
                        ;GOCMPN
                        ;;*****************************************************************

013502                          GOCMPN:
013502  032777  000002  165710          BIT     #2,@CSR         ;IS F1 CLR?
013510  001415                        BEQ     1$              ;IF SO,SET F1
013512  017737  165702  001446          MOV     @CSR,TEMP
013520  042737  100000  001446          BIC     #BIT15,TEMP
013526  042737  000002  001446          BIC     #2,TEMP         ;F1 IS SET,CLR IT
013534  013777  001446  165656          MOV     TEMP,@CSR
013542  000207                        RTS     PC
013544  017737  165650  001446  1$:     MOV     @CSR,TEMP
013552  042737  100000  001446          BIC     #BIT15,TEMP
013560  052737  000002  001446          BIS     #2,TEMP
013566  013777  001446  165624          MOV     TEMP,@CSR
013574  000207                        RTS     PC
```

```
1832                                    .SBTTL  CPU PRIORITY SUBROUTINES
1833                                    ;;**********************************************************************
1834                                                    ;CPUHI
1835                                    ;;**********************************************************************
1836
1837 013576                             CPUHI:
1838 013576  023727  001434  000126              CMP     BOARD,#126      ;IS THIS CPU LSI?
1839 013604  001003                             BNE     1$              ;NO,IT IS UNIBUS
1840 013606  106427                             106427
     013610  000200                             200
1841 013612  000207                             RTS     PC
1842
1843 013614  012737  000340  177776  1$:       MOV     #340,PS         ;SET HIGH PRIOR. IN UNIBUS
1844 013622  000207                             RTS     PC
1845
1846                                    ;;**********************************************************************
1847                                                    ;CPULO
1848                                    ;;**********************************************************************
1849
1850 013624                             CPULO:
1851 013624  023727  001434  000126              CMP     BOARD,#126
1852 013632  001003                             BNE     1$
1853 013634  106427                             106427
     013636  000000                             0
1854 013640  000207                             RTS     PC
1855 013642  012737  000000  177776  1$:       MOV     #0,PS
1856 013650  000207                             RTS     PC
```

```
                                        .SBTTL  INTERPROCESSOR PROGRAM RESYNCHRONIZATION  ROUTINE
                                        ;;**********************************************************************
                                                        ;RSYNC
                                        ;;**********************************************************************

013652  004737  011200          RESYNC: JSR     PC,RSTVEC
013656  012706  001100                  MOV     #STACK,SP
013662  005737  001452                  TST     MSTER           ;WAS THIS CPU STARTED AS MASTER?
013666  001404                          BEQ     MRDELY          ;YES
013670  012737  000017  001444  SLDELY: MOV     #15.,TIME       ;SLAVE CPU;WAIT FOR MASTER TO REACH ITS
                                                                ;RESYNC ROUTINE
013676  000403                          BR      WTSYNC
013700  012737  000036  001444  MRDELY: MOV     #30.,TIME
013706                          WTSYNC:
013706  005001                  1$:     CLR     R1
013710  005301                  2$:     DEC     R1
013712  001376                          BNE     2$
013714  005337  001444                  DEC     TIME
013720  001372                          BNE     1$
013722  000005                          RESET
013724  104401  013732                  TYPE    ,65$            ;;TYPE ASCIZ STRING
013730  000407                          BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>/*RESYNC.../<CRLF>
013750                                  64$:
013750  000137  002742                  JMP     SYNCST

                                        .SBTTL  ASCII MESSAGES
013754      124     105     123  EM1:    .ASCII  /TEST 1    STATUS: MASTER/<CRLF>
013757      124     040     061
013762      040     040     040
013765      040     123     124
013770      101     124     125
013773      123     072     040
013776      115     101     123
014001      124     105     122
014004      200
014005      123     114     101          .ASCIZ  /SLAVE FAILED TO ECHO DBR CONTENTS/
014010      126     105     040
014013      106     101     111
014016      114     105     104
014021      040     124     117
014024      040     105     103
014027      110     117     040
014032      104     102     122
014035      040     103     117
014040      116     124     105
014043      116     124     123
014046      000
014047      040     123     124  EM2:    .ASCII  / STATUS: MASTER/<CRLF>
014052      101     124     125
014055      123     072     040
014060      115     101     123
014063      124     105     122
014066      200
014067      123     114     101          .ASCIZ  /SLAVE FAILED TO ECHO 'STAT' BITS/
014072      126     105     040
014075      106     101     111
```

```
                014100  114     105     104
                014103  040     124     117
                014106  040     105     103             -
                014111  110     117     040
                014114  047     123     124
                014117  101     124     047
                014122  040     102     111
                014125  124     123     000
1886            014130  040     123     124     EM3:    .ASCII  / STATUS: MASTER/<CRLF>
                014133  101     124     125
                014136  123     072     040
                014141  115     101     123
                014144  124     105     122
                014147  200
1887            014150  106     101     111             .ASCIZ  /FAILED TO INTR ON A 'DATI'/
                014153  114     105     104
                014156  040     124     117
                014161  040     111     116
                014164  124     122     040
                014167  117     116     040
                014172  101     040     047
                014175  104     101     124
                014200  111     047     000
1888            014203  040     123     124     EM4:    .ASCII  / STATUS: MASTER/<CRLF>
                014206  101     124     125
                014211  123     072     040
                014214  115     101     123
                014217  124     105     122
                014222  200
1889            014223  123     124     101             .ASCIZ  /STATUS ER ON 'DATI'/
                014226  124     125     123
                014231  040     105     122
                014234  040     117     116
                014237  040     047     104
                014242  101     124     111
                014245  047     000
1890            014247  040     123     124     EM5:    .ASCII  / STATUS: MASTER/<CRLF>
                014252  101     124     125
                014255  123     072     040
                014260  115     101     123
                014263  124     105     122
                014266  200
1891            014267  127     117     122             .ASCIZ  /WORD COUNT ER ON 'DATI'/
                014272  104     040     103
                014275  117     125     116
                014300  124     040     105
                014303  122     040     117
                014306  116     040     047
                014311  104     101     124
                014314  111     047     000
1892            014317  040     123     124     EM6:    .ASCII  / STATUS: MASTER/<CRLF>
                014322  101     124     125
                014325  123     072     040
                014330  115     101     123
                014333  124     105     122
                014336  200
1893            014337  102     125     106             .ASCIZ  /BUFFER ADRS ER ON 'DATI'/
```

```
        014342  106  105  122
        014345  040  101  104
        014350  122  123  040
        014353  105  122  040
        014356  117  116  040
        014361  047  104  101
        014364  124  111  047
        014367  000
  1894  014370  040  123  124  EM7:    .ASCII  / STATUS: SLAVE/<CRLF>
        014373  101  124  125
        014376  123  072  040
        014401  123  114  101
        014404  126  105  200
  1895  014407  106  101  111         .ASCIZ  /FAILED TO INTR ON A 'DATO'/
        014412  114  105  104
        014415  040  124  117
        014420  040  111  116
        014423  124  122  040
        014426  117  116  040
        014431  101  040  047
        014434  104  101  124
        014437  117  047  000
  1896  014442  040  123  124  EM10:   .ASCII  / STATUS: SLAVE/<CRLF>
        014445  101  124  125
        014450  123  072  040
        014453  123  114  101
        014456  126  105  200
  1897  014461  123  124  101         .ASCIZ  /STATUS ER ON 'DATO'/
        014464  124  125  123
        014467  040  105  122
        014472  040  117  116
        014475  040  047  104
        014500  101  124  117
        014503  047  000
  1898  014505  040  123  124  EM11:   .ASCII  / STATUS: SLAVE/<CRLF>
        014510  101  124  125
        014513  123  072  040
        014516  123  114  101
        014521  126  105  200
  1899  014524  127  117  122         .ASCIZ  /WORD COUNT ER ON 'DATO'/
        014527  104  040  103
        014532  117  125  116
        014535  124  040  105
        014540  122  040  117
        014543  116  040  047
        014546  104  101  124
        014551  117  047  000
  1900  014554  040  123  124  EM12:   .ASCII  / STATUS: SLAVE/<CRLF>
        014557  101  124  125
        014562  123  072  040
        014565  123  114  101
        014570  126  105  200
  1901  014573  102  125  106         .ASCIZ  /BUFFER ADRS ER ON 'DATO'/
        014576  106  105  122
        014601  040  101  104
        014604  122  123  040
        014607  105  122  040
```

```
        014612  117     116     040
        014615  047     104     101
        014620  124     117     047
        014623  000
  1902  014624  040     123     124     EM13:   .ASCII  / STATUS: SLAVE/<CRLF>
        014627  101     124     125
        014632  123     072     040
        014635  123     114     101
        014640  126     105     200
  1903  014643  104     101     124             .ASCIZ  /DATA ER ON 'DATO'/
        014646  101     040     105
        014651  122     040     117
        014654  116     040     047
        014657  104     101     124
        014662  117     047     000
  1904  014665  123     124     101     EM14:   .ASCII  /STATUS: SLAVE/<CRLF>
        014670  124     125     123
        014673  072     040     123
        014676  114     101     126
        014701  105     200
  1905  014703  122     105     101             .ASCIZ  /READY BIT WAS SET - IT SHOULD BE CLEAR/<CRLF>
        014706  104     131     040
        014711  102     111     124
        014714  040     127     101
        014717  123     040     123
        014722  105     124     040
        014725  055     040     111
        014730  124     040     123
        014733  110     117     125
        014736  114     104     040
        014741  102     105     040
        014744  103     114     105
        014747  101     122     200
        014752  000
  1906  014753  040     123     124     EM15:   .ASCII  / STATUS: MASTER/<CRLF>
        014756  101     124     125
        014761  123     072     040
        014764  115     101     123
        014767  124     105     122
        014772  200
  1907  014773  115     101     123             .ASCII  /MASTER CANNOT START COMMUNICATION WITH SLAVE-/
        014776  124     105     122
        015001  040     103     101
        015004  116     116     117
        015007  124     040     123
        015012  124     101     122
        015015  124     040     103
        015020  117     115     115
        015023  125     116     111
        015026  103     101     124
        015031  111     117     116
        015034  040     127     111
        015037  124     110     040
        015042  123     114     101
        015045  126     105     055
  1908  015050  127     101     111             .ASCIZ  /WAITING FOR DSTATC /<CRLF>
        015053  124     111     116
```

```
                015056  107  040  106
                015061  117  122  040
                015064  104  123  124
                015067  101  124  103
                015072  040  200  000
1909            015075  123  124  125  EM16:   .ASCIZ  /STUCK WAITING FOR COMPANION FOR GO-AHEAD/<CRLF>
                015100  103  113  040
                015103  127  101  111
                015106  124  111  116
                015111  107  040  106
                015114  117  122  040
                015117  103  117  115
                015122  120  101  116
                015125  111  117  116
                015130  040  106  117
                015133  122  040  107
                015136  117  055  101
                015141  110  105  101
                015144  104  200  000
1910            015147  124  105  123  EM17:   .ASCII  /TEST 5 TIME OUT ERROR-/
                015152  124  040  065
                015155  040  124  111
                015160  115  105  040
                015163  117  125  124
                015166  040  105  122
                015171  122  117  122
                015174  055
1911            015175  123  114  101        .ASCIZ  /SLAVE SHOULD NOT HAVE  INTERRUPTED/<CRLF>
                015200  126  105  040
                015203  123  110  117
                015206  125  114  104
                015211  040  116  117
                015214  124  040  110
                015217  101  126  105
                015222  040  040  111
                015225  116  124  105
                015230  122  122  125
                015233  120  124  105
                015236  104  200  000
1912
1913            015241  040  123  124  EM20:   .ASCII  / STATUS: MASTER/<CRLF>
                015244  101  124  125
                015247  123  072  040
                015252  115  101  123
                015255  124  105  122
                015260  200
1914            015261  115  101  123        .ASCII  /MASTER IS LEFT WAITING FOR INTERRUPT TO OCCUR/
                015264  124  105  122
                015267  040  111  123
                015272  040  114  105
                015275  106  124  040
                015300  127  101  111
                015303  124  111  116
                015306  107  040  106
                015311  117  122  040
                015314  111  116  124
                015317  105  122  122
```

```
                015322  125     120     124
                015325  040     124     117
                015330  040     117     103
                015333  103     125     122
1915            015336  126     111     101             .ASCIZ  /VIA ATTN SET/<CRLF>
                015341  040     101     124
                015344  124     116     040
                015347  123     105     124
                015352  200     000
1916            015354  124     105     123     EM21:   .ASCII  /TEST 2   STATUS: MASTER/<CRLF>
                015357  124     040     062
                015362  040     040     040
                015365  123     124     101
                015370  124     125     123
                015373  072     040     115
                015376  101     123     124
                015401  105     122     200
1917            015404  105     122     122             .ASCIZ  /ERROR BIT IN MASTER CSR NOT CLEAR/
                015407  117     122     040
                015412  102     111     124
                015415  040     111     116
                015420  040     115     101
                015423  123     124     105
                015426  122     040     103
                015431  123     122     040
                015434  116     117     124
                015437  040     103     114
                015442  105     101     122
                015445  000
1918            015446  102     125     122     EM22:   .ASCIZ  /BURST DATA LATE BIT OF EIR WAS NOT SET/<CRLF>
                015451  123     124     040
                015454  104     101     124
                015457  101     040     114
                015462  101     124     105
                015465  040     102     111
                015470  124     040     117
                015473  106     040     105
                015476  111     122     040
                015501  127     101     123
                015504  040     116     117
                015507  124     040     123
                015512  105     124     200
                015515  000
1919
1920            015516  200     124     105     MSG1:   .ASCIZ  <CRLF>  /TEST 3-BLOCK MODE XFERS/<CRLF>
                015521  123     124     040
                015524  063     055     102
                015527  114     117     103
                015532  113     040     115
                015535  117     104     105
                015540  040     130     106
                015543  105     122     123
                015546  200     000
1921            015550  200     124     105     MSG2:   .ASCIZ  <CRLF>  /TEST 4-BURST MODE XFERS/<CRLF>
                015553  123     124     040
                015556  064     055     102
                015561  125     122     123
```

```
                015564  124     040     115
                015567  117     104     105
                015572  040     130     106
                015575  105     122     123
                015600  200     000
1922            015602  200     124     105     MSG3:   .ASCIZ  <CRLF>/TEST 5-BURST DATA LATE/<CRLF>
                015605  123     124     040
                015610  065     055     102
                015613  125     122     123
                015616  124     040     104
                015621  101     124     101
                015624  040     114     101
                015627  124     105     200
                015632  000
1923            015633  105     122     122     DH1:    .ASCIZ  /ERRPC   BUSADR  EXPCT   RCVD/
                015636  120     103     040
                015641  040     040     102
                015644  125     123     101
                015647  104     122     040
                015652  040     105     130
                015655  120     103     124
                015660  040     040     040
                015663  122     103     126
                015666  104     000
1924            015670  105     122     122     DH2:    .ASCIZ  /ERRPC   MEMADR  EXPCT   RCVD/
                015673  120     103     040
                015676  040     040     115
                015701  105     115     101
                015704  104     122     040
                015707  040     105     130
                015712  120     103     124
                015715  040     040     040
                015720  122     103     126
                015723  104     000
1925            015725  105     122     122     DH3:    .ASCIZ  /ERRPC/
                015730  120     103     000
1926                                            .EVEN
1927            015734  001116  001122  001124  DT1:    $ERRPC,$BDADR,$GDDAT,$BDDAT,0
                015742  001126  000000
1928            015746  001116  000000          DT2:    $ERRPC,0
1929            015752  001440  000000          DT3:    TESTPC,0
1930                                            ;;**********************************************************************
1931                                            ;'DBUF' IS THE WORKING AREA IN MEM FOR ALL NPR OPERATIONS
1932                                            ;;**********************************************************************
1933            015756  000000                  DBUF:   0               ;1ST ADRS OF DATA BUFFER
1934                    000001                          .END
```

```
ABORT   001436
BAR     001416
BDR     001422
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
BOARD   001434
BPTVEC= 000014
BRDTYP  002056
CKSWR = 104407
CLRBUF  011220
CNT     001430
CPUHI   013576
CPULO   013624
CR    = 000015
CRLF  = 000200
CSR     001420
DBUF    015756
DDISP = 177570
DH1     015633
DH2     015670
DH3     015725
DISPLA  001142
DISPRE  000174
DRVCT0  001424
DRVCT2  001426
DRVECT  001412
DSWR  = 177570
DT1     015734
DT2     015746
DT3     015752
EMTVEC= 000030
EM1     013754
EM10    014442
EM11    014505
EM12    014554
EM13    014624
EM14    014665
EM15    014753
EM16    015075
EM17    015147
EM2     014047
EM20    015241
EM21    015354
EM22    015446
EM3     014130
EM4     014203
EM5     014247
EM6     014317
EM7     014370
EOPT    010772
EOPTA   011140
EOPTB   011146
ERROR = 104000
ERRVEC= 000004
GOCMPN  013502
GTSWR = 104406
HT    = 000011
ICOUNT  001454
INTADR  001410
IOTVEC= 000020
LDBUF   011236
LF    = 000012
MINIT1  002760
MINIT2  003226
MORS    002524
MRDELY  013700
MSBRD   004550
MSG1    015516
MSG2    015550
MSG3    015602
MSTER   001452
MTPS  = 106427
MTST1   003106
MTST2   003526
MTST3   003652
MTST4   004620
MTST5   005446
NUM     001432
PIRQ  = 177772
PIRQVE= 000240
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSW   = 177776
PWRVEC= 000024
RCVTMO  001500
RDCHR = 104410
RDLIN = 104411
RESVEC= 000010
RESYNC  013652
RPRAM   001464
RSTVEC  011200
RSYNC = 104412
RTRN    003462
R6    =%000006
R7    =%000007
SAVE    001450
SCOPE = 000004
SETUP2  001704
SETUP3  001730
SETVEC  011160
SINIT1  006072
SINIT2  006304
SLBRD   007330
SLDELY  013670
SLVFIN  010776
SL1STR  006240
STACK = 001100
START   001522
START1  001506
START2  001514
STKLMT= 177774
STST1   006230
STST2   006422
STST3   006502
STST4   007416
STST5   010260
SWR     001140
SWRCK   012306
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
SYNCST  002742
TBITVE= 000014
TEMP    001446
TESTPC  001440
TIME    001444
TKVEC = 000060
TOTAL   001442
TPVEC = 000064
TRAPVE= 000034
TRTVEC= 000014
TYPDS = 104405
TYPE  = 104401
TYPOC = 104402
TYPON = 104404
TYPOS = 104403
WCR     001414
WTCMPN  013414
WTSYNC  013706
XMITMO  001472
XPRAM   001456
$AUTOB  001134
$BDADR  001122
$BDDAT  001126
$BELL   001160
$CHARC  011514
$CKSWR  012544
$CMTAG  001100
$CM3  = 000000
$CNTLG  013271
$CNTLU  013264
$CRLF   001165
$DBLK   012162
$DOAGN  011114
$DTBL   012152
$ENDAD  011104
$ENDCT  011052
$ENDMG  011123
$ENULL  011120
$EOP    011022
$EOPCT  011044
$ERFLG  001103
$ERMAX  001115
$ERROR  012172
$ERRPC  001116
$ERRTB  001170
$ERRTY  012316
$ERTTL  001112
$FILLC  001156
$FILLS  001155
$GDADR  001120
$GDDAT  001124
$GET42  011074
$GTSWR  012614
$HD   = 000003
$ICNT   001104
$INTAG  001135
$ITEMB  001114
$LF     001166
$LPADR  001106
$LPERR  001110
$MNEW   013307
$MSWR   013276
$NULL   001154
$OCNT   011742
$OMODE  011744
$OVER   012530
$PASS   001100
$QUES   001164
$RDCHR  013026
$RDLIN  013146
$RDSZ = 000010
$RTNAD  011116
$SCOPE  012452
$SETUP= 000107
$STUP = 177777
$SVLAD  012514
$SWR  = 122000
$SWRMK= 000000
$TERM = 000026
$TKB    001146
$TKS    001144
$TN   = 000001
$TPB    001152
$TPFLG  001157
$TPS    001150
$TRAP   013320
$TRAP2  013354
$TRP  = 000013
$TRPAD  013366
$TSTNM  001102
$TTYIN  013254
$TYPDS  011746
$TYPE   011300
$TYPEC  011450
$TYPEX  011516
$TYPOC  011544
$TYPON  011560
$TYPOS  011520
$XTSTR  012456
$$GET4= 000000
$OFILL  011743
```

```
. ABS.  015760     000
        000000     001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  35848 WORDS  ( 141 PAGES)
DYNAMIC MEMORY:  20746 WORDS  ( 79 PAGES)
ELAPSED TIME:  00:05:21
LINK,LINK=SYSMAC.MLB/ML,LINK.P11
```