DR11-W

DR11-W INTERPROC EXER
CZDRKB0

COPYRIGHT (c) 1978-83
AH-E783B-MC
FICHE 1 OF 1

APR 1984
digital
Made In USA



```
                        .REM    *
```

IDENTIFICATION

PRODUCT CODE: AC-E782B-MC

PRODUCT NAME: CZDRKB0 DR11-W INTRPROC EXER

DATE: DEC 1983

MAINTAINER: DIAGNOSTIC ENGINEERING

COPYRIGHT (C) 1978,1983
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

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

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

```
44
45
46
47                                  TABLE OF CONTENTS
48                                  -----------------
49
50                                  1.0     ABSTRACT
51
52                                  2.0     REQUIREMENTS
53                                  2.1     EQUIPMENT
54                                  2.2     STORAGE
55
56                                  3.0     LOADING AND START  PROCEDURE
57
58                                  4.0     ERRORS
59                                  4.1     ERROR COMMENT
60                                  4.2     ERROR DATA
61                                  4.3     ERROR RECOVERY
62                                  4.3.1   RECOVERABLE ERRORS
63                                  4.3.2   NON-RECOVERABLE ERRORS
64
65                                  5.0     SOFTWARE SWITCH REGISTER
66                                  5.1     OPTIONS
67                                  5.2     IMPLEMENTATION
68                                  5.3     CONTROL
69
70                                  6.0     MISCELLANEOUS
71                                  6.1     DRV11B AND/OR DR11W BUS & VECTOR ADDRESS MODIFICATION
72                                  6.2     POWER FAIL
73
74                                  7.0     EXECUTION TIME
75
76                                  8.0     PROGRAM DESCRIPTION
77                                  8.1     GENERAL
78                                  8.2     PROGRAM SEGMENTS
79
80                                  9.0     MODIFICATION HISTORY
81
```

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        0       1
END PASS        0       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.

```
311
312
313                         6.0     MISCELLANEOUS
314                                 -------------
315
316
317
318                         6.1     DRV11B AND/OR DR11W BUS & VECTOR ADDRESS MODIFICATION
319
320                                 MODIFY LOCATION 'INTADR' IF BASE BUS ADDRESS IS NOT 172410
321                                 MODIFY LOCATION 'DRVECT' IF VECTOR ADDRESS IS NOT 124
322
323                                 NOTE:  USE APPROPRIATE PATCH FACILITIES TO MODIFY THESE LOCATIONS
324                                 AFTER PROGRAM LOAD.
325
326                         6.2     POWER FAIL
327
328                                 THE PROGRAM OFFERS NO PROVISIONS FOR RESTART PROCEDURES
329                                 DUE TO POWER FAIL.  UPON POWER UP, AND ASSUMING NON
330                                 VOLATILE MEMORY, THE PROGRAM OF EACH SYSTEM MUST BE RESTARTED
331                                 ACCORDING TO SECT. 3.0.
332
333
334                         7.0     EXECUTION TIME
335                                 --------------
336
337                                 EXECUTION TIME IS ABOUT 3 SEC  FOR NO ITERATIONS
338                                 EXECUTION TIME W/ITERATIONS IS DETERMINED BY THE SYSTEM CONFIGURATION
339                                 ACTUAL RUN TIMES ARE AS FOLLOWS:
340                                         11/34 - 11/05: 3MIN
341                                         11/70 - 11/45: 45SEC.
342                                         11/70 - 11/03: 1MIN 15SEC.
343
344                         8.0     PROGRAM DESCRIPTION
345                                 -------------------
346
347
348
349                         8.1     GENERAL
350
351                                 THIS INTERPROCESSOR EXERCISER WAS DESIGNED TO TEST THE I/O
352                                 ABILITY OF THE DR11W GENERAL PURPOSE INTERFACE TO COMMUNICATE TO
353                                 ANOTHER DR11W OR DRV11B LOCATED IN ANOTHER SYSTEM.  THE TWO
354                                 COMPUTERS ARE STARTED AT DIFFERENT ADDRESSES TO ESTABLISH
355                                 INITIAL SYNCHRONIZATION.  THE SLAVE COMPUTER IS STARTED 1ST
356                                 (START 204) AND THE MASTER COMPUTER IS STARTED 2ND (START 200).
357                                 THE TERMS 'MASTER' AND 'SLAVE' SHOULD BE USED LOOSELY AS THE
358                                 MASTER WILL BECOME THE SLAVE AND THE SLAVE WILL BECOME THE
359                                 MASTER AS THE PROGRAM ADVANCES.  THE COMPUTER STARTED AT
360                                 ADDRESS 200 WILL ALWAYS REPORT THE 'END OF PASS' MESSAGE.
361
362
363
364                         8.2     PROGRAM SEGMENTS
365
366                                 1A.  MTST1 - MASTER SENDS OUT PROGRAM CONTROLLED SINGLE WORDS
367                                      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 5 WORDS.
                        TIMEOUT OCCURS AND THE BUS IS RELEASED
                        THEN CHECKS FOR PROPER INTERRUPT STATUS, WC, BA & DATA.
```

```
                                9.0     MODIFICATION HISTORY

                                        REV B.0 12-DEC-83       H.P. HOLSINGER (CWO)
                                          ALTERED TEST EXIT (MTST4&5, STST4&5) COMPANION STATUS
                                          CHECK TO COMPENSATE FOR ANY STATUS LINE SKEWING.
                                          ADDED SAVE PC TO RESYNC CODE FOR LINK DEBUG ASSISTANCE.
```

```
448
477                             .TITLE  CZDRKB0 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-C5), JAN, 1981.
                                ;*
000001                          $TN=1
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
478     122000                  $SWR=122000
479     000001                  $TN=1
480                             .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        10            BELL ON ERROR
481                             .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
482             106427                  MTPS=106427             ;INSTR EQUATE THAT MOVES BYTE TO PSW
483                                     .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
484 000204      000137  001514          JMP     @#START2        ;GO START AS SLAVE COMPUTER
485             000100                  .=100
486 000100      000104  000200  000002  .WORD   104,200,2       ;IF 'B EVENT' ON Q-BUS IS CONNECTED
487                                                             ;JUST DO A RTI (IGNORE IT)
```

```
488                                     .SBTTL  COMMON TAGS

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

        001100                                  .=1100
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
                                        ;;**********************************************************************
```

ERROR POINTER TABLE

```
                                .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                      $ERRTB:
489                             ;ERROR  1
490 001170  014234                      EM1             ;SLAVE  FAILED TO ECHO DBR CONTENTS
491 001172  016113                      DH1             ;ERRPC  BUSADR  EXPCT   RCVD
492 001174  016214                      DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
493 001176  000000                      0
494
495                             ;ERROR  2
496 001200  014327                      EM2             ;SLAVE  FAILED TO ECHO 'STAT' BITS
497 001202  016113                      DH1             ;ERRPC  BUSADR  EXPCT   RCVD
498 001204  016214                      DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
499 001206  000000                      0
500
501                             ;ERROR  3
502 001210  014410                      EM3             ;FAILED TO INTR ON A 'DATI'
503 001212  016113                      DH1             ;ERRPC  BUSADR  EXPCT   RCVD
504 001214  016214                      DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
505 001216  000000                      0
506
507                             ;ERROR  4
508 001220  014463                      EM4             ;STATUS ER ON 'DATI'
509 001222  016113                      DH1             ;ERRPC  BUSADR  EXPCT   RCVD
510 001224  016214                      DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
511 001226  000000                      0
```

ERROR POINTER TABLE

```
                                ;ERROR  5
001230  014527                          EM5             ;WORD COUNT ER ON 'DATI'
001232  016113                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001234  016214                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001236  000000                          0

                                ;ERROR  6
001240  014577                          EM6             ;BUFFER ADRS ER ON 'DATI'
001242  016113                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001244  016214                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001246  000000                          0

                                ;ERROR  7
001250  014650                          EM7             ;FAILED TO INTR ON A 'DATO'
001252  016113                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001254  016214                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001256  000000                          0
                                ;ERROR  10
001260  014722                          EM10            ;STATUS ER ON 'DATO'
001262  016113                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001264  016214                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001266  000000                          0

                                ;ERROR  11
001270  014765                          EM11            ;WORD COUNT ER ON 'DATO'
001272  016113                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001274  016214                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001276  000000                          0

                                ;ERROR  12
001300  015034                          EM12            ;BUFFER ADRS ER ON 'DATO'
001302  016113                          DH1             ;ERRPC  BUSADR  EXPCT   RCVD
001304  016214                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001306  000000                          0

                                ;ERROR  13
001310  015104                          EM13            ;DATA ER ON 'DATO'
001312  016150                          DH2             ;ERRPC  MEMADR  EXPCT   RCVD
001314  016214                          DT1             ;$ERRPC $BDADR  $GDDAT  $BDDAT
001316  000000                          0

                                ;ERROR  14
001320  015145                          EM14            ;CSR READY BIT WAS SET - SHOULD BE CLEAR
001322  016113                          DH1
001324  016214                          DT1
001326  000000                          0

                                ;ERROR 15
001330  015233                          EM15    ;PROTOCOL INITIALIZATION ERROR -MASTER AND SLAVE
001332  016205                          DH3
001334  016226                          DT2
001336  000000                          0

                                ;ERROR 16
001340  015355                          EM16    ;STUCK FOREVER WAITING FOR COMPANION
001342  016205                          DH3
```

ERROR POINTER TABLE

```
    570 001344  016232                    DT3
    571 001346  000000                    0
    572
    573                             ;ERROR 17
    574 001350  015427                    EM17    ;TIMEOUT ERROR BURST MODE XFER-INCURRED INTERRUPT
    575 001352  016205                    DH3
    576 001354  016226                    DT2
    577 001356  000000                    0
    578
    579                             ;ERROR 20       ;SLAVE DID NOT INTERRUPT MASTER
    580 001360  015521                    EM20
    581 001362  016205                    DH3
    582 001364  016226                    DT2
    583 001366  000000                    0
    584
    585                             ;ERROR 21
    586 001370  015634                    EM21            ;TEST 2-ERROR BIT SET IN MASTER
    587 001372  016205                    DH3
    588 001374  016232                    DT3
    589 001376  000000                    0
    590
    591                             ;ERROR  22
    592 001400  015726                    EM22            ;BURST DATA LATE BIT IN EIR NOT SET
    593 001402  016113                    DH1
    594 001404  016214                    DT1
    595 001406  000000                    0
    596
```

F2


```
                                ; 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  016236                          DBUF            ;XMIT BUFFER ADRS
001462  000511                          511             ;XMIT STATUS/CONTROL
001464  177740                  RPRAM:  -32.            ;RCV WORD COUNT
001466  016236                          DBUF            ;RCV BUFFER ADRS
001470  000113                          113             ;RCV STATUS/CONTROL
                                
001472  177773                  XMITMO: -5              ;XMIT BURST DATA LATE TEST.
001474  016236                            DBUF
001476  000501                            501
                                
001500  177766                  RCVTMO: -10.            ;RCV BURST DATA LATE TEST
001502  016236                          DBUF
001504  000103                          103
                                
```

PROGRAM START

```
                                        .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  012714  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001550  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001556  012737  012434  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001564  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001572  012737  013572  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
```

H2

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
        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$:  .ASCIZ  <CRLF>#DR11W INTERPROCESSOR EXERCISER  #<CRLF>
        002054                          64$:
    669 002054  000240                          NOP
```

BOARD TYPE DIALOGUE

```
                                                .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$:     LOV     #127,BOARD
002412  104401  002420                          TYPE    ,75$            ;;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
```

J2


```
        002561  040     102     117
        002564  101     122     104
        002567  200
  704   002570  111     116     124             .ASCII  /INTERPROCESSOR 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
  705   002643  040     000                     .ASCIZ  / /
  706   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
  707   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
  708   002740  040     000                     .ASCIZ  / /
  709                                           .EVEN
```

```
                                        .SBTTL
                                        .SBTTL          MASTER TESTS
                                        .SBTTL
002742                          15$:
002742  004737  014050        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  006162                JMP     SINIT1          ;NO - GO ECHO MASTER'S DATA
```

```
                                        ;MASTER PROTOCOL INIT-TEST 1
                                    ;;**********************************************************************
                                    
003760  032777  020000  176432  MINIT1: BIT     @#BIT13,@CSR
003766  001425                          BEQ     1$
003770  104401  002776                  TYPE    ,65$            ;;TYPE ASCIZ STRING
002774  000421                          BR      64$             ;;GET OVER THE ASCIZ
                                    ;;65$:  .ASCIZ  <CRLF>/CABLE IS NOT INSERTED - HALTING /
003040                              64$:
003040  000000                          HALT
003042  004737  014050          1$:     JSR     PC,CPUHI
003046  012706  001100                  MOV     #STACK,SP
003052  005077  176342                  CLR     @CSR
003056  012737  001000  001444          MOV     #1000,TIME
003064  032777  001000  176326  2$:     BIT     #1000,@CSR      ;HAS SLAVE SET MASTER DSTATC
003072  001005                          BNE     MTST1
003074  005337  001444                  DEC     TIME
003100  001371                          BNE     2$
003102  104015                  104000+15
003104  104412                          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  013754                  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  013666                  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  013754                  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  014076                  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  011360          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$:     MOV     R0,@CSR         ;LOAD FNCT BITS
003552  004737  013666                  JSR     PC,WTCMPN       ;LOOK FOR SLAVE TO ECHO VIA 'STAT' BITS
003556  012737  001000  001444          MOV     #1000.,TIME     ;DELAY
003564  005337  001444          2$:     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  013754                  JSR     PC,GOCMPN       ;GO TO COMPANION
003646  004737  013666                  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  014050                  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  011340          37$:    JSR     R5,SETVEC       ;SET UP INTR RETURN ADRS
003736  004172                                  3$                      ;RETURN TO 3$ ON INTR
003740  004537  011416                          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  014076                          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  015776                  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,$BDADR      ;SET UP CSR ADRS
004276  104401  015776                TYPE    ,MSG1
004302  104004                    104000+4
004304  000446                        BR      6$              ;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,$BDADR      ;SET UP WCR ADRS
004330  104401  015776                TYPE    ,MSG1
004334  104005                    104000+5
004336  000431                        BR      6$              ;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  015776                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$:    RSYNC                   ;RSYNC ON ERROR
004544  004737  011360          10$:    JSR     PC,RSTVEC       ;GO RESTORE VECTOR
                                        
```

F3


```
                                        ;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  013754                  JSR     PC,GOCMPN       ;TELL SLAVE TO RESPOND
 004576  004737  013666                  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  006162          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  014050          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  011340          37$:    JSR     R5,SETVEC       ;SET UP INTR RETURN ADRS
004704  005070                          3$                      ;RETURN TO 3$ ON INTR
004706  004537  011416                  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  014076                  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,$BDADR      ;SET UP CSR ADRS
005060  104401  016030                  TYPE    ,MSG2
005064  104003                  104000+3
005066  000504                          BR      6$              ;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
```

```
1065 005154  104401  016030                    TYPE    ,MSG2
1066 005160  104004                      104000+4
1067 005162  000446                            BR      6$              ;GO RESYNC ON ER
1068 005164  005037  001124              4$:   CLR     $GDDAT          ;LD EXPECTED WC
1069 005170  017737  174220  001126            MOV     @WCR,$BDDAT     ;READ WORD COUNT
1070 005176  001407                            BEQ     5$              ;BR IF ZERO
1071 005200  013737  001414  001122            MOV     WCR,$BDADR      ;SET UP WCR ADRS
1072 005206  104401  016030                    TYPE    ,MSG2
1073 005212  104005                      104000+5
1074 005214  000431                            BR      6$              ;GO RESYNC ON ER
1075 005216  013737  001460  001124  5$:       MOV     XPRAM+2,$GDDAT  ;GET STARTING ADRS OF XFER
1076 005224  013700  001456                    MOV     XPRAM,R0        ;GET WC
1077 005230  005400                            NEG     R0              ;GET ACTUAL #
1078 005232  006300                            ASL     R0              ;CONVERT TO WORD
1079 005234  060037  001124                    ADD     R0,$GDDAT       ;ADD TO BASE ADRS
1080 005240  017737  174152  001126            MOV     @BAR,$BDDAT     ;READ BAR ADRS
1081 005246  042737  000001  001126            BIC     #BIT00,$BDDAT   ;ELIMINATE LSB
1082 005254  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
1083 005262  001446                            BEQ     8$              ;BR IF SO
1084 005264  013737  001416  001122            MOV     BAR,$BDADR      ;SET UP BAR ADRS
1085 005272  104401  016030                    TYPE    ,MSG2
1086 005276  104006                      104000+6
1087
1088
1089
1090                                    ;ERROR IN MASTER-WAIT ON SLAVE FOR ITS STATUS
1091
1092 005300  012737  010000  001444  6$:       MOV     #10000,TIME     ;WAIT FOR SLAVE TO COMPLETE DATA CK
1093 005306  005337  001444              7$:   DEC     TIME            ;COUNT AWAY
1094 005312  001375                            BNE     7$              ;UNTIL DONE
1095 005314  000005                            RESET                   ;IF ERROR THAN INIT
1096 005316  012777  000010  174074            MOV     #10,@CSR        ;TELL SLAVE WE HAVE AN ERROR
1097
1098                                    ;THIS SECTION MODIFIED FOR STATUS LINE SKEW
1099 005324                              21$:
1100 005324  017737  174070  001450            MOV     @CSR,SAVE       ; READ STATUS
1101 005332  042737  170777  001450            BIC     #170777,SAVE    ; CLEAR DON'T CARES
1102 005340  001771                            BEQ     21$             ; LOOP IF COMPANION NOT READY YET
1103
1104 005342  017737  174052  001450            MOV     @CSR,SAVE       ; RE-READ STATUS
1105 005350  042737  170777  001450            BIC     #170777,SAVE    ; CLEAR DON'T CARES
1106 005356  022737  005000  001450            CMP     #5000,SAVE      ; CHECK FOR GOOD STATUS
1107 005364  001404                            BEQ     24$             ; IF EQ THEN RESYNC
1108 005366                              23$:
1109 005366  022737  004000  001450            CMP     #4000,SAVE      ; ELSE CHECK FOR COMPANION ERROR
1110 005374  001353                            BNE     21$             ; IF NE THEN COMPANION STILL NOT READY
1111 005376                              24$:
1112 005376  104412                            RSYNC                   ;RSYNC ON ERROR
1113
1114
1115                                    ;NO ERROR IN MASTER-WAIT ON SLAVE FOR ITS STATUS
1116 005400  012737  010000  001444  8$:       MOV     #10000,TIME     ;WAIT FOR SLAVE TO COMPLETE CKS
1117 005406  005337  001444             20$:   DEC     TIME            ;COUNT AWAY
1118 005412  001375                            BNE     20$             ;UNTIL DONE
1119 005414  012777  000012  173776            MOV     #12,@CSR        ;TELL SLAVE ALL OK
1120
1121                                    ;THIS SECTION MODIFIED FOR STATUS LINE SKEW
```

```
1122 005422                          9$:
1123 005422  017737  173772  001450        MOV     @CSR,SAVE       ; READ CSR
1124 005430  042737  170777  001450        BIC     #170777,SAVE    ; SAVE 'STAT' BITS
1125 005436  001771                        BEQ     9$              ; IF COMPANION NOT READY THEN LOOP
1126
1127 005440  017737  173754  001450        MOV     @CSR,SAVE       ; RE-READ STATUS
1128 005446  042737  170777  001450        BIC     #170777,SAVE    ; CLEAR DON'T CARES
1129 005454  022737  005000  001450        CMP     #5000,SAVE      ; CHECK FOR COMPANION OK
1130 005462  001405                        BEQ     10$             ; IF SLAVE OK THEN CONTINUE
1131
1132 005464  022737  004000  001450        CMP     #4000,SAVE      ; ELSE CHECK FOR COMPANION ERROR
1133 005472  001353                        BNE     9$              ; IF NE THEN COMPANION STILL NOT READY
1134 005474                          11$:
1135 005474  104412                        RSYNC                   ;RSYNC ON ERROR
1136 005476                          10$:
1137 005476  004737  011360                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.
                                        ;;**********************************************************************
                                        ;;**********************************************************************
005502                                  MTST5:
005502  004737  014050          1$:     JSR     PC,CPUHI                ;NO INTR ALLOWED YET
005506  005077  173706          41$:    CLR     @CSR
005512  022777  000200  173700          CMP     #200,@CSR       ;IS ONLY READY SET
005520  001372                          BNE     41$
005522  017737  173672  001446  38$:    MOV     @CSR,TEMP
005530  042737  100000  001446          BIC     #BIT15,TEMP
005536  052737  000002  001446          BIS     #2,TEMP         ;SET DSTATC IN SLAVE
                                                                ;TELLS SLAVE TO SETUP FOR XFERS
005544  013777  001446  173646          MOV     TEMP,@CSR
005552  032777  001000  173640  40$:    BIT     #1000,@CSR      ;DSTATC HERE SAYS SLAVE
                                                                ;IS READY TO DO XFERS
005560  001774                          BEQ     40$
005562  004537  011340          37$:    JSR     R5,SETVEC       ;SET UP INTR RETURN ADRS
005566  005722                          3$                      ;RETURN TO 3$ ON INTR
005570  004537  011416                  JSR     R5,LDBUF        ;GO LOAD 'DBUF' WITH DATA PTRN
005574  177740                          -32.                    ;DO 32. LOCATIONS
005576  012737  010000  001444          MOV     #10000,TIME     ;SET UP A TIMER VALUE
005604  013777  001472  173602          MOV     XMITM0,@WCR     ;SET UP WORD COUNT
005612  013777  001474  173576          MOV     XMITM0+2,@BAR   ;SET UP BUFFER ADRS
005620  004737  014076                  JSR     PC,CPULO        ;ALLOW THE RDY INTR
005624  013777  001476  173566          MOV     XMITM0+4,@CSR   ;SET UP CONTROL - IE, GO,AND CYCLE

005632  005337  001444          2$:     DEC     TIME            ;WAIT FOR INTR
005636  001375                          BNE     2$              ;UNTIL 0
005640  017737  173554  001126          MOV     @CSR,$BDDAT     ;READ CSR - SHOULD NEVER GET HERE
005646  017737  173546  001446          MOV     @CSR,TEMP
005654  042737  100000  001446          BIC     #BIT15,TEMP
005662  042737  000100  001446          BIC     #100,TEMP       ;DISABLE IE
005670  013777  001446  173522          MOV     TEMP,@CSR
005676  012737  001300  001124          MOV     #1300,$GDDAT    ;LD EXPECTED  STAT C    , RDY, IE
005704  013737  001420  001122          MOV     CSR,$BDADR      ;SET UP CSR ADRS
005712  104401  016062                  TYPE    ,MSG3
005716  104003                  104000+3
005720  000430                          BR      6$              ;GO RESYNC ON ERROR
005722  022626                  3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
005724  017737  173470  001446          MOV     @CSR,TEMP
005732  042737  100000  001446          BIC     #BIT15,TEMP
005740  042737  000100  001446          BIC     #100,TEMP       ;DISABLE IE
005746  013777  001446  173444          MOV     TEMP,@CSR
005754  017737  173434  001126          MOV     @WCR,$BDDAT     ;READ WORD COUNT
005762  001441                          BEQ     9$              ;BR IF ZERO
005764  013737  001414  001122          MOV     WCR,$BDADR      ;SET UP WCR ADRS

```

K3

WHILE SLAVE RECEIVES 5 AND TIMES OUT (DR11-W TO DR11-W ONLY)

```
                                                TYPE    ,MSG3
                                104000+5
                                                BR      6$              ;GO RESYNC ON ER


                                                ;ERROR IN MASTER-WAIT FOR SLAVE STATUS

                                6$:             RESET
006002  000005
006004  012777  000010  173406                  MOV     #10,@CSR        ;TELL SLAVE WE HAVE AN ERROR

                                                ;THIS SECTION MODIFIED FOR STATUS LINE SKEW.
006012                          21$:
006012  017737  173402  001450                  MOV     @CSR,SAVE       ; READ STATUS
006020  042737  170777  001450                  BIC     #170777,SAVE    ; CLEAR DON'T CARES
006026  001771                                  BEQ     21$             ; IF COMPANION NOT READY THEN LOOP

006030  017737  173364  001450                  MOV     @CSR,SAVE       ; RE-READ STATUS
006036  042737  170777  001450                  BIC     #170777,SAVE    ; CLEAR DON'T CARES
006044  022737  005000  001450                  CMP     #5000,SAVE      ; CHECK FOR COMPANION OK
006052  001404                                  BEQ     24$             ; IF EQ THEN RESYNC
006054                          23$:
006054  022737  004000  001450                  CMP     #4000,SAVE      ; CHECK FOR COMPANION ERROR
006062  001353                                  BNE     21$             ; IF NE THEN COMPANION STILL NOT READY
006064                          24$:
006064  104412                                  RSYNC                   ;RSYNC ON ERROR


                                                ;NO ERROR IN MASTER-WAIT FOR SLAVE STATUS

                                                ;THIS SECTION MODIFIED FOR STATUS LINE SKEW.
006066                          9$:
006066  017737  173326  001450                  MOV     @CSR,SAVE       ; READ CSR
006074  042737  170777  001450                  BIC     #170777,SAVE    ; SAVE 'STAT' BITS
006102  001771                                  BEQ     9$              ; IF COMPANION NOT READY THEN LOOP

006104  017737  173310  001450                  MOV     @CSR,SAVE       ; RE-READ STATUS
006112  042737  170777  001450                  BIC     #170777,SAVE    ; CLEAR DON'T CARES
006120  022737  005000  001450                  CMP     #5000,SAVE      ; CHECK FOR COMPANION OK
006126  001410                                  BEQ     10$             ; IF SLAVE OK THEN CONTINUE

006130  022737  004000  001450                  CMP     #4000,SAVE      ; ELSE CHECK FOR COMPANION ERROR
006136  001353                                  BNE     9$              ; IF NE THEN COMPANION STILL NOT READY

006140  012777  000010  173252                  MOV     #10,@CSR        ; REPORT ERROR STATUS TO SLAVE
006146  104412                                  RSYNC                   ; RSYNC ON ERROR
006150                          10$:
006150  012777  000012  173242                  MOV     #12,@CSR        ; TELL SLAVE ALL OK
006156  004737  011360                          JSR     PC,RSTVEC       ; GO RESTORE VECTOR
```

```
1248                                    .SBTTL
1249                                    .SBTTL          SLAVE TESTS
1250                                    .SBTTL
1251
1252
1253                                            ;SLAVE PROTOCOL INIT-TEST1
1254                                    ;;**************************************************************************
1255 006162  032777  020000  173230  SINIT1: BIT     #BIT13,@CSR
1256 006170  001425                          BEQ     1$
1257 006172  104401  006200                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
     006176  000421                          BR      64$                     ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>/CABLE IS NOT INSERTED - HALTING /
     006242                             64$:
1258 006242  000000                          HALT
1259 006244  004737  014050          1$:     JSR     PC,CPUHI
1260 006250  012706  001100                  MOV     #STACK,SP
1261 006254  005077  173140                  CLR     @CSR
1262 006260  032777  001000  173132          BIT     #1000,@CSR              ;WHAT IS DSTATC
1263 006266  001404                          BEQ     3$                      ;IF CLR-BRANCH AND WAIT FOR MASTER
1264 006270  032777  001000  173122  2$:     BIT     #1000,@CSR              ;IF NOT,WAIT FOR CLEAR
1265 006276  001374                          BNE     2$
1266 006300  004737  013754          3$:     JSR     PC,GOCMPN
1267 006304  032777  001000  173106  4$:     BIT     #1000,@CSR              ;HAS MASTER SET DSTATC IN SLAVE?
1268 006312  001774                          BEQ     4$                      ;NOT YET
1269 006314  000137  006330                  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
                                ;;**************************************************************************
                                ;;**************************************************************************

006320  004737  013754          STST1:  JSR     PC,GOCMPN       ;TELL MASTER WE ARE READY
006324  004737  013666                  JSR     PC,WTCMPN       ;DATA AVAILABLE?
                                                                ;WAIT ON IT
006330  017737  173066  001450  SL1STR: MOV     @BDR,SAVE       ;GET DATA
006336  022737  177001  001450          CMP     #177001,SAVE    ;TEST TERMINATOR?
006344  001412                        BEQ     4$              ;BR IF SO
006346  013777  001450  173046        MOV     SAVE,@BDR       ;SEND IT BACK
006354  012737  001000  001444        MOV     #1000,TIME      ;DELAY BEFORE CALLING MASTER
006362  005337  001444          5$:     DEC     TIME
006366  001375                        BNE     5$
006370  000753                        BR      STST1           ;GO LOOK FOR MORE DATA
006372  000240          4$:     NOP                             ;
```

* TEST THAT SLAVE INTERRUPTS MASTER:F2 TO ATTN

```
1293                                    .SBTTL  * TEST THAT SLAVE INTERRUPTS MASTER:F2 TO ATTN
1294                                            ;SLAVE PROTOCOL INIT-TEST2
1295                                    ;;**********************************************************************
1296 006374                            SINIT2:
1297
1298 006374  005077  173020                       CLR     @CSR            ;CLR DSTATC OF MASTER
1299 006400  004737  013666                       JSR     PC,WTCMPN
1300 006404  017737  173010  001446               MOV     @CSR,TEMP
1301 006412  042737  100000  001446               BIC     #BIT15,TEMP
1302 006420  052737  000004  001446               BIS     #4,TEMP         ;SET INTERRUPT MASTER
1303 006426  013777  001446  172764               MOV     TEMP,@CSR
1304 006434  017737  172760  001446               MOV     @CSR,TEMP
1305 006442  042737  100000  001446               BIC     #BIT15,TEMP
1306 006450  042737  000004  001446               BIC     #4,TEMP         ;CLR F2
1307 006456  013777  001446  172734               MOV     TEMP,@CSR
1308 006464  012737  010000  001444               MOV     #10000,TIME
1309 006472  032777  001000  172720  2$:          BIT     #1000,@CSR      ;WAIT DSTATC SLAVE TO CLEAR
1310 006500  001404                               BEQ     STST2
1311 006502  005337  001444                       DEC     TIME
1312 006506  001371                               BNE     2$
1313 006510  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
                                        ;;**********************************************************************
                                        ;;**********************************************************************
006512                                  STST2:
006512  004737  013666          4$:     JSR     PC,WTCMPN       ;WAIT FOR MASTER
006516  012737  001000  001444          MOV     #1000,TIME      ;DELAY
006524  005337  001444          5$:     DEC     TIME
006530  001375                          BNE     5$
006532  022777  177002  172662  2$:     CMP     #177002,@BDR    ;TEST TERMINATOR?
006540  001412                          BEQ     3$              ;BR IF SO
006542  017737  172652  001450          MOV     @CSR,SAVE       ;READ THE CSR
006550  042737  170777  001450          BIC     #170777,SAVE    ;SAVE ONLY THE STAT BITS
006556  113777  001451  172634          MOVB    SAVE+1,@CSR     ;ECHO WITH FNCT BITS
006564  000752                          BR      4$              ;LOOK FOR NEXT 'STAT' CODE
006566  004737  013754          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
                                        ;;**********************************************************************
                                        ;;**********************************************************************
006572                                  STST3:
006572  004737  014050                  1$:     JSR     PC,CPUHI        ;NO INTRS ALLOWED YET
006576  012737  001000  001444                  MOV     #1000,TIME
006604  005337  001444                  40$:    DEC     TIME
006610  001375                                  BNE     40$
006612  005077  172602                  41$:    CLR     @CSR
006616  022777  001200  172574                  CMP     #1200,@CSR      ;ONLY DSTATC AND READY SET
006624  001372                                  BNE     41$
006626  004537  011340                  39$:    JSR     R5,SETVEC       ;SET UP THE INTR RETURN ADRS
006632  007016                                  3$                      ;RETURN TO 3$ ON DATO INTR
006634  004537  011400                          JSR     R5,CLRBUF       ;GO CLR 'DBUF'
006640  177740                                  -32.                    ;DO 32. LOCATIONS
006642  012737  010000  001444                  MOV     #10000,TIME     ;SET UP A TIMER VALUE
006650  013777  001464  172536                  MOV     RPRAM,@WCR      ;SET UP WORD COUNT
006656  013777  001466  172532                  MOV     RPRAM+2,@BAR    ;SET UP BUFFER ADRS
006664  004737  014076                          JSR     PC,CPULO        ;ALLOW THE INTR
006670  017737  172524  001446                  MOV     @CSR,TEMP
006676  042737  100000  001446                  BIC     #BIT15,TEMP
006704  052737  000012  001446                  BIS     #12,TEMP        ;FNCT 3&1
006712  013777  001446  172500                  MOV     TEMP,@CSR
006720  013777  001470  172472                  MOV     RPRAM+4,@CSR    ;SET UP CONTROL - FNCT 3 & 1, IE  & GO
006726  005337  001444                  2$:     DEC     TIME            ;WAIT FOR INTR
006732  001375                                  BNE     2$              ;UNTIL ZERO
006734  017737  172460  001126                  MOV     @CSR,$BDDAT     ;READ CSR - SHOULD NEVER GET HERE
006742  017737  172452  001446                  MOV     @CSR,TEMP
006750  042737  100000  001446                  BIC     #BIT15,TEMP
006756  042737  000100  001446                  BIC     #100,TEMP       ;DISABLE IE
006764  013777  001446  172426                  MOV     TEMP,@CSR
006772  012737  004312  001124                  MOV     #4312,$GDDAT    ;LD EXPECTED - STAT A, RDY, IE & FNCT 3 & 1
007000  013737  001420  001122                  MOV     CSR,$BDADR      ;SET UP CSR ADRS
                                        
007006  104401  015776                          TYPE    ,MSG1
007012  104007                          104000+7
007014  000540                                  BR      10$             ;GO RESYNC ON ER
007016  022626                          3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RETURN
007020  017737  172374  001126                  MOV     @CSR,$BDDAT     ;READ STATUS
007026  017737  172366  001446                  MOV     @CSR,TEMP
007034  042737  000100  001446                  BIC     #100,TEMP       ;DISABLE IE
007042  042737  100000  001446                  BIC     #BIT15,TEMP
007050  013777  001446  172342                  MOV     TEMP,@CSR
007056  012737  004312  001124                  MOV     #4312,$GDDAT    ;LD EXPECTED - STAT A, RDY, IE & FNCT 3 & 1
007064  023737  001124  001126                  CMP     $GDDAT,$BDDAT   ;CORRECT?
007072  001407                                  BEQ     4$              ;BR IF SO
007074  013737  001420  001122                  MOV     CSR,$BDADR      ;SET UP CSR ADRS
                                        
007102  104401  015776                          TYPE    ,MSG1
007106  104010                          104000+10
```

```
1392 007110  000502                        BR      10$             ;GO RESYNC ON ERROR
1393 007112  005037  001124          4$:   CLR     $GDDAT          ;LD EXPECTED WC
1394 007116  017737  172272  001126        MOV     @WCR,$BDDAT     ;READ WCR
1395 007124  001407                        BEQ     5$              ;BR IF SO
1396 007126  013737  001414  001122        MOV     WCR,$BDADR      ;SET UP WCR ADRS
1397
1398 007134  104401  015776                TYPE    ,MSG1
1399 007140  104011                104000+11
1400 007142  000465                        BR      10$             ;GO RESYNC ON ERROR
1401 007144  013737  001466  001124  5$:   MOV     RPRAM+2,$GDDAT  ;GET STARTING ADRS OF XFER
1402 007152  013700  001464                MOV     RPRAM,R0        ;GET WC
1403 007156  005400                        NEG     R0              ;GET ACTUAL #
1404 007160  006300                        ASL     R0              ;CONVERT TO WORD
1405 007162  060037  001124                ADD     R0,$GDDAT       ;ADD TO BASE ADRS
1406 007166  017737  172224  001126        MOV     @BAR,$BDDAT     ;READ BAR ADRS
1407 007174  042737  000001  001126        BIC     #BIT00,$BDDAT   ;ELIMINATE LSB
1408 007202  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;CORRECT?
1409 007210  001407                        BEQ     6$              ;BR IF SO
1410 007212  013737  001416  001122        MOV     BAR,$BDADR      ;SET UP BAR ADRS
1411
1412 007220  104401  015776                TYPE    ,MSG1
1413 007224  104012                104000+12
1414 007226  000433                        BR      10$             ;GO RESYNC ON ERROR
1415 007230  013700  001466          6$:   MOV     RPRAM+2,R0      ;GET BUFFER ADRS
1416 007234  013701  001464                MOV     RPRAM,R1        ;GET WORD COUNT
1417 007240  012702  177776          7$:   MOV     #177776,R2      ;GET 1ST DATA PTRN (FLOATING 0)
1418 007244  005003                        CLR     R3              ;R3 SAYS WHEN TO SHIFT PTRN
1419 007246  020220                8$:   CMP     R2,(R0)+        ;COMPARE DATA IN DBUF TO EXPECTED
1420 007250  001011                        BNE     9$              ;BR IF DATA ERROR
1421 007252  005201                        INC     R1              ;COUNT THE WORD COUNT
1422 007254  001435                        BEQ     13$             ;BR IF DATA CHECKS DONE
1423 007256  005102                        COM     R2              ;CONVERT PTRN TO FLOATING 1
1424 007260  005103                        COM     R3              ;TIME TO SHIFT?
1425 007262  001371                        BNE     8$              ;BR IF NOT - GO CK NEXT
1426 007264  006302                        ASL     R2              ;FLOAT PTRN LEFT
1427 007266  005202                        INC     R2              ;KEEP LSB SET
1428 007270  103363                        BCC     7$              ;GO RESET FLOATING PTRN
1429 007272  000765                        BR      8$              ;GO CHECK NEXT
1430 007274  014037  001126          9$:   MOV     -(R0),$BDDAT    ;GET BAD DATA
1431 007300  010037  001122                MOV     R0,$BDADR       ;GET MEM ADRS OF DATA ER
1432 007304  010237  001124                MOV     R2,$GDDAT       ;LD EXPECTED DATA
1433
1434 007310  104401  015776                TYPE    ,MSG1
1435 007314  104013                104000+13
```

```
1437                                            ;ERROR IN SLAVE-WAIT FOR MASTER STATUS
1438
1439 007316  000005                  10$:       RESET                       ;TELL MASTER WE HAVE ERROR
1440
1441 007320  017737  172074  001450  15$:       MOV     @CSR,SAVE           ;LOOP ON ERROR
1442 007326  042737  170777  001450              BIC     #170777,SAVE        ;BUT WAIT FOR MASTER
1443 007334  001404                              BEQ     24$
1444 007336  022737  001000  001450  16$:       CMP     #1000,SAVE
1445 007344  001365                              BNE     15$
1446 007346  104412                  24$:       RSYNC                       ;RSYNC ON ERROR
1447
1448
1449
1450                                            ;NO ERROR IN SLAVE-WAIT FOR MASTER STATUS
1451
1452 007350  112777  000002  172042  13$:       MOVB    #2,@CSR             ;TELL MASTER ALL OK
1453 007356  017737  172036  001450  14$:       MOV     @CSR,SAVE           ;READ CSR
1454 007364  042737  170777  001450              BIC     #170777,SAVE        ;SAVE 'STAT' BITS ONLY
1455 007372  001405                              BEQ     31$                 ;IS ER IN MASTER?
1456 007374  022737  001000  001450              CMP     #1000,SAVE          ;MASTER OK?
1457 007402  001402                              BEQ     30$
1458 007404  000764                              BR      14$
1459 007406  104412                  31$:       RSYNC                       ;RSYNC ON ERROR
1460 007410  004737  011360          30$:       JSR     PC,RSTVEC           ;GO RESTORE VECTOR
1461
1462
1463
```

```
1465                                        ;BURST MODE WORD XFER TEST DETERMINATION
1466                                ;;**********************************************************************
1467                                        ;DETERMINE IF BURST XFERS AND BURST DATA LATE ARE TO BE
1468                                        ;DONE
1469                                ;;**********************************************************************
1470
1471 007414  004737  013666                  JSR     PC,WTCMPN       ;WAIT FOR MASTER TO INDICATE
1472                                                                  ;WHAT BOARD TYPE IT IS
1473 007420  012737  001000  001444  SLBRD:  MOV     #1000,TIME      ;WAIT ON MASTER
1474 007426  005337  001444          1$:     DEC     TIME
1475 007432  001375                          BNE     1$
1476 007434  013737  001434  001442          MOV     BOARD,TOTAL     ;DETERMINE IF BOTH MASTER
1477                                                                  ;AND SLAVE ARE DR11W'S
1478 007442  067737  171754  001442          ADD     @BDR,TOTAL
1479 007450  023727  001442  000256          CMP     TOTAL,#256
1480 007456  001406                          BEQ     2$              ;DO BURST MODE
1481
1482
1483 007460  005077  171736                  CLR     @BDR
1484 007464  004737  013754                  JSR     PC,GOCMPN
1485 007470  000137  011156                  JMP     SLVFIN          ;NO BURST MODE
1486 007474  012777  000001  171720  2$:     MOV     #1,@BDR
1487 007502  004737  013754                  JSR     PC,GOCMPN
1488
```

```
                                        .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
                                        ;;**********************************************************************
                                        ;;**********************************************************************
007506                                  STST4:
007506  004737  014050          1$:     JSR     PC,CPUHI        ;NO INTRS ALLOWED YET
007512  012737  001000  001444          MOV     #1000,TIME
007520  005337  001444          40$:    DEC     TIME
007524  001375                          BNE     40$
007526  005077  171666          41$:    CLR     @CSR
007532  022777  001200  171660          CMP     #1200,@CSR      ;ONLY DSTATC AND READY SET
007540  001372                          BNE     41$
007542  004537  011340          39$:    JSR     R5,SETVEC       ;SET UP THE INTR RETURN ADRS
007546  007732                          3$                      ;RETURN TO 3$ ON DATO INTR
007550  004537  011400                  JSR     R5,CLRBUF       ;GO CLR 'DBUF'
007554  177740                          -32.                    ;DO 32. LOCATIONS
007556  012737  010000  001444          MOV     #10000,TIME     ;SET UP A TIMER VALUE
007564  013777  001464  171622          MOV     RPRAM,@WCR      ;SET UP WORD COUNT
007572  013777  001466  171616          MOV     RPRAM+2,@BAR    ;SET UP BUFFER ADRS
007600  004737  014076                  JSR     PC,CPULO
                                                                ;ALLOW THE INTR
007604  017737  171610  001446          MOV     @CSR,TEMP
007612  042737  100000  001446          BIC     #BIT15,TEMP
007620  052737  000002  001446          BIS     #2,TEMP         ;FNCT 1
007626  013777  001446  171564          MOV     TEMP,@CSR
007634  012777  000103  171556          MOV     #103,@CSR       ;SET UP CONTROL - FNCT  1, IE  & GO
007642  005337  001444          2$:     DEC     TIME            ;WAIT FOR INTR
007646  001375                          BNE     2$              ;UNTIL ZERO
007650  017737  171544  001126          MOV     @CSR,$BDDAT     ;READ CSR - SHOULD NEVER GET HERE
007656  017737  171536  001446          MOV     @CSR,TEMP
007664  042737  100000  001446          BIC     #BIT15,TEMP
007672  042737  000100  001446          BIC     #100,TEMP       ;DISABLE IE
007700  013777  001446  171512          MOV     TEMP,@CSR
007706  012737  000302  001124          MOV     #302,$GDDAT     ;LD EXPECTED -  RDY, IE & FNCT  1
007714  013737  001420  001122          MOV     CSR,$BDADR      ;SET UP CSR ADRS

007722  104401  016030                  TYPE    ,MSG2
007726  104007                  104000+7
007730  000540                          BR      10$             ;GO RESYNC ON ER
007732  022626                  3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RETURN
007734  017737  171460  001126          MOV     @CSR,$BDDAT     ;READ STATUS
007742  017737  171452  001446          MOV     @CSR,TEMP
007750  042737  100000  001446          BIC     #BIT15,TEMP
007756  042737  000100  001446          BIC     #100,TEMP       ;DISABLE IE
007764  013777  001446  171426          MOV     TEMP,@CSR
007772  012737  000302  001124          MOV     #302,$GDDAT     ;LD EXPECTED -  RDY, IE & FNCT  1
010000  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
010006  001407                          BEQ     4$              ;BR IF SO
010010  013737  001420  001122          MOV     CSR,$BDADR      ;SET UP CSR ADRS

010016  104401  016030                  TYPE    ,MSG2
010022  104010                  104000+10
010024  000502                          BR      10$             ;GO RESYNC ON ERROR
010026  005037  001124          4$:     CLR     $GDDAT          ;LD EXPECTED WC
```

```
1547 010032  017737  171356  001126          MOV     &WCR,$BDDAT     ;READ WCR
1548 010040  001407                          BEQ     5$              ;BR IF SO
1549 010042  013737  001414  001122          MOV     WCR,$BDADR      ;SET UP WCR ADRS
1550
1551 010050  104401  016030                  TYPE    ,MSG2
1552 010054  104011                  104000+11
1553 010056  000465                          BR      10$             ;GO RESYNC ON ERROR
1554 010060  013737  001466  001124  5$:     MOV     RPRAM+2,$GDDAT  ;GET STARTING ADRS OF XFER
1555 010066  013700  001464                  MOV     RPRAM,R0        ;GET WC
1556 010072  005400                          NEG     R0              ;GET ACTUAL #
1557 010074  006300                          ASL     R0              ;CONVERT TO WORD
1558 010076  060037  001124                  ADD     R0,$GDDAT       ;ADD TO BASE ADRS
1559 010102  017737  171310  001126          MOV     &BAR,$BDDAT     ;READ BAR ADRS
1560 010110  042737  000001  001126          BIC     #BIT00,$BDDAT   ;ELIMINATE LSB
1561 010116  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1562 010124  001407                          BEQ     6$              ;BR IF SO
1563 010126  013737  001416  001122          MOV     BAR,$BDADR      ;SET UP BAR ADRS
1564
1565 010134  104401  016030                  TYPE    ,MSG2
1566 010140  104012                  104000+12
1567 010142  000433                          BR      10$             ;GO RESYNC ON ERROR
1568 010144  013700  001466          6$:     MOV     RPRAM+2,R0      ;GET BUFFER ADRS
1569 010150  013701  001464                  MOV     RPRAM,R1        ;GET WORD COUNT
1570 010154  012702  177776          7$:     MOV     #177776,R2      ;GET 1ST DATA PTRN (FLOATING 0)
1571 010160  005003                          CLR     R3              ;R3 SAYS WHEN TO SHIFT PTRN
1572 010162  020220                  8$:     CMP     R2,(R0)+        ;COMPARE DATA IN DBUF TO EXPECTED
1573 010164  001011                          BNE     9$              ;BR IF DATA ERROR
1574 010166  005201                          INC     R1              ;COUNT THE WORD COUNT
1575 010170  001452                          BEQ     13$             ;BR IF DATA CHECKS DONE
1576 010172  005102                          COM     R2              ;CONVERT PTRN TO FLOATING 1
1577 010174  005103                          COM     R3              ;TIME TO SHIFT?
1578 010176  001371                          BNE     8$              ;BR IF NOT - GO CK NEXT
1579 010200  006302                          ASL     R2              ;FLOAT PTRN LEFT
1580 010202  005202                          INC     R2              ;KEEP LSB SET
1581 010204  103363                          BCC     7$              ;GO RESET FLOATING PTRN
1582 010206  000765                          BR      8$              ;GO CHECK NEXT
1583 010210  014037  001126          9$:     MOV     -(R0),$BDDAT    ;GET BAD DATA
1584 010214  010037  001122                  MOV     R0,$BDADR       ;GET MEM ADRS OF DATA ER
1585 010220  010237  001124                  MOV     R2,$GDDAT       ;LD EXPECTED DATA
1586
1587 010224  104401  016030                  TYPE    ,MSG2
1588 010230  104013                  104000+13
```

```
1590                                                ;ERROR IN SLAVE-WAIT FOR MASTER STATUS
1591
1592 010232  000005                        10$:     RESET
1593 010234  012777  000010  171156                 MOV     #10,@CSR        ;TELL MASTER WE HAVE AN ERROR
1594
1595                                                ;THIS SECTION MODIFIED FOR STATUS LINE SKEW.
1596 010242                                15$:
1597 010242  017737  171152  001450                 MOV     @CSR,SAVE       ; READ STATUS
1598 010250  042737  170777  001450                 BIC     #170777,SAVE    ; CLEAR DON'T CARES
1599 010256  001771                                 BEQ     15$             ; IF COMPANION NOT READY THEN LOOP
1600
1601 010260  017737  171134  001450                 MOV     @CSR,SAVE       ; RE-READ STATUS
1602 010266  042737  170777  001450                 BIC     #170777,SAVE    ; CLEAR DON'T CARES
1603 010274  022737  005000  001450                 CMP     #5000,SAVE      ; CHECK FOR COMPANION OK
1604 010302  001404                                 BEQ     24$             ; IF EQ THEN RESYNC
1605 010304                                16$:
1606 010304  022737  004000  001450                 CMP     #4000,SAVE      ; ELSE CHECK FOR COMPANION ERROR
1607 010312  001353                                 BNE     15$             ; IF NE THEN COMPANION STILL NOT READY
1608 010314                                24$:
1609 010314  104412                                 RSYNC                   ; RSYNC ON ERROR
1610
1611
1612
1613                                                ;NO ERROR IN SLAVE-WAIT FOR MASTER STATUS
1614
1615 010316  012777  000012  171074  13$:     MOV     #12,@CSR        ; TELL MASTER ALL OK
1616
1617                                                ;THIS SECTION MODIFIED FOR STATUS LINE SKEW.
1618 010324                                14$:
1619 010324  017737  171070  001450                 MOV     @CSR,SAVE       ; READ CSR
1620 010332  042737  170777  001450                 BIC     #170777,SAVE    ; SAVE 'STAT' BITS ONLY
1621 010340  001771                                 BEQ     14$             ; IF COMPANION NOT READY THEN LOOP
1622
1623 010342  017737  171052  001450                 MOV     @CSR,SAVE       ; RE-READ STATUS
1624 010350  042737  170777  001450                 BIC     #170777,SAVE    ; CLEAR DON'T CARES
1625 010356  022737  005000  001450                 CMP     #5000,SAVE      ; CHECK FOR COMPANION OK
1626 010364  001405                                 BEQ     30$             ; IF MASTER OK THEN CONTINUE
1627
1628 010366  022737  004000  001450                 CMP     #4000,SAVE      ; ELSE CHECK FOR COMPANION ERROR
1629 010374  001353                                 BNE     14$             ; IF NE THEN COMPANION STILL NOT READY
1630 010376                                31$:
1631 010376  104412                                 RSYNC                   ; RSYNC ON ERROR
1632 010400                                30$:
1633 010400  004737  011360                         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
                                ;;**********************************************************************
                                ;;**********************************************************************
010404                          STST5:
010404  004737  014050          1$:     JSR     PC,CPUHI        ;NO INTRS ALLOWED YET
010410  012737  001000  001444          MOV     #1000,TIME      ;DELAY ON MASTER
010416  005337  001444          40$:    DEC     TIME
010422  001375                          BNE     40$
010424  005077  170770          41$:    CLR     @CSR
010430  022777  001200  170762          CMP     #1200,@CSR      ;ONLY DSTATC AND READY
010436  001372                          BNE     41$
010440  004537  011340          39$:    JSR     R5,SETVEC       ;SET UP THE INTR RETURN ADRS
010444  010550                          2$                      ;RETURN TO 2$ ON DATO INTR
010446  004537  011400                  JSR     R5,CLRBUF       ;GO CLR 'DBUF'
010452  177740                          -32.                    ;DO 32. LOCATIONS
010454  012737  040000  001444          MOV     #40000,TIME     ;SET UP A TIMER VALUE
010462  013777  001500  170724          MOV     RCVTMO,@WCR     ;SET UP WORD COUNT
010470  013777  001502  170720          MOV     RCVTMO+2,@BAR   ;SET UP BUFFER ADRS
010476  004737  014076                  JSR     PC,CPULO        ;ALLOW THE INTR
010502  017737  170712  001446          MOV     @CSR,TEMP
010510  042737  100000  001446          BIC     #BIT15,TEMP
010516  052737  000002  001446          BIS     #2,TEMP         ;FNCT1
010524  013777  001446  170666          MOV     TEMP,@CSR
010532  013777  001504  170660          MOV     RCVTMO+4,@CSR   ;SET UP CONTROL:IE,FNCT1 & GO
                                        ;TIMEOUT
010540  005337  001444          25$:    DEC     TIME            ;WAIT FOR MASTER TO SETUP;
010544  001375                          BNE     25$             ;THEN SLAVE BUS WILL BE HELD FOR DURATION
                                                                ;OF BURST XFERS;TIMEOUT WILL OCCUR WITH SUBSEQUENT
                                                                ;RELEASE OF BUS AND PROGRAM CONTINUATION
010546  000416                  BR      3$                      ;GO TO CHECKS
010550  017737  170644  001446  2$:     MOV     @CSR,TEMP
010556  042737  100000  001446          BIC     #BIT15,TEMP
010564  042737  000100  001446          BIC     #100,TEMP       ;DISABLE IE
010572  013777  001446  170620          MOV     TEMP,@CSR
010600  104017                  104000+17
010602  000471                  BR      10$
010604  017737  170610  001126  3$:     MOV     @CSR,$BDDAT     ;READ STATUS
010612  032737  000200  001126          BIT     #BIT7,$BDDAT    ;TEST READY BIT
010620  001412                          BEQ     5$
010622  013737  001446  001124          MOV     TEMP,$GDDAT
010630  013737  001420  001122          MOV     CSR,$BDADR
010636  104401  016062                  TYPE    ,MSG3
010642  104014                  104000+14
010644  000450                  BR      10$
010646  012777  100000  170544  5$:     MOV     #BIT15,@CSR     ;GO TO EIR
010654  017737  170540  001126          MOV     @CSR,$BDDAT     ;SAVE EIR
```

```
 1692 010662  032737  001000  001126          BIT     #BIT9,$BDDAT    ;TEST BURST DATA LATE BIT
 1693 010670  001015                          BNE     4$
 1694 010672  042737  000400  001126          BIC     #BIT8,$BDDAT
 1695 010700  012737  101000  001124          MOV     #101000,$GDDAT
 1696 010706  013737  001420  001122          MOV     CSR,$BDADR
 1697 010714  104401  016062                  TYPE    ,MSG3
 1698 010720  104022                  104000+22
 1699 010722  000421                          BR      10$             ;GO RESYNC ON ERROR
 1700 010724  012737  177773  001124  4$:     MOV     #177773,$GDDAT          ;LD EXPECTED WC
 1701 010732  017737  170456  001126          MOV     @WCR,$BDDAT     ;READ WCR
 1702 010740  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
 1703 010746  001441                          BEQ     13$             ;BR IF SO
 1704 010750  013737  001414  001122          MOV     WCR,$BDADR      ;SET UP WCR ADRS
 1705 010756  104401  016062                  TYPE    ,MSG3
 1706 010762  104011                  104000+11
 1707 010764  000400                          BR      10$             ;GO RESYNC ON ERROR
 1708
 1709
 1710
 1711                                         ;ERROR IN SLAVE-WAIT FOR MASTER STATUS
 1712
 1713 010766  000005                  10$:    RESET                   ;IF ERROR,INIT
 1714 010770  012777  000010  170422          MOV     #10,@CSR
 1715
 1716                                         ;THIS SECTION MODIFIED FOR STATUS LINE SKEW.
 1717 010776                          15$:
 1718 010776  017737  170416  001450          MOV     @CSR,SAVE       ; READ STATUS
 1719 011004  042737  170777  001450          BIC     #170777,SAVE    ; CLEAR DON'T CARES
 1720 011012  001771                          BEQ     15$             ; IF COMPANION NOT READY THEN LOOP
 1721
 1722 011014  017737  170400  001450          MOV     @CSR,SAVE       ; RE-READ STATUS
 1723 011022  042737  170777  001450          BIC     #170777,SAVE    ; CLEAR DON'T CARES
 1724 011030  022737  005000  001450          CMP     #5000,SAVE      ; CHECK FOR COMPANION OK
 1725 011036  001404                          BEQ     24$             ; IF EQ THEN RESYNC
 1726 011040                          16$:
 1727 011040  022737  004000  001450          CMP     #4000,SAVE      ; ELSE CHECK FOR COMPANION ERROR
 1728 011046  001353                          BNE     15$             ; IF NE THEN COMPANION STILL NOT READY
 1729 011050                          24$:
 1730 011050  104412                          RSYNC                   ; RSYNC ON ERROR
 1731
 1732
 1733
 1734                                         ;NO ERROR IN SLAVE-WAIT FOR MASTER STATUS
 1735
 1736 011052  000005                  13$:    RESET                   ;SLAVE HAS TIMED OUT LEAVING
 1737                                                                 ;READY CLEAR. FNCT BITS TO DSTAT
 1738                                                                 ;BITS COMMUNICATION CAN BE
 1739                                                                 ;RESUMED BY SETTING READY.RESET
 1740                                                                 ;WILL SET READY.
 1741 011054  012737  010000  001444          MOV #10000,TIME
 1742 011062  005337  001444          11$:    DEC TIME
 1743 011066  001375                          BNE 11$
 1744 011070  012777  000012  170322          MOV     #12,@CSR        ;TELL MASTER ALL OK
 1745
 1746                                         ;THIS SECTION MODIFIED FOR STATUS LINE SKEW.
 1747 011076                          14$:
 1748 011076  017737  170316  001450          MOV     @CSR,SAVE       ; READ CSR
```

```
1749 011104  042737  170777  001450          BIC     #170777,SAVE    ; SAVE 'STAT' BITS ONLY
1750 011112  001771                          BEQ     14$             ; IF COMPANION NOT READY THEN LOOP
1751
1752 011114  017737  170300  001450          MOV     @CSR,SAVE       ; RE-READ STATUS
1753 011122  042737  170777  001450          BIC     #170777,SAVE    ; CLEAR DON'T CARES
1754 011130  022737  005000  001450          CMP     #5000,SAVE      ; CHECK FOR COMPANION OK
1755 011136  001405                          BEQ     EOPT            ; IF MASTER OK THEN END OF PASS
1756
1757 011140  022737  004000  001450          CMP     #4000,SAVE      ; ELSE CHECK FOR COMPANION ERROR
1758 011146  001353                          BNE     14$             ; IF NE THEN COMPANION STILL NOT READY
1759
1760 011150  104412                          RSYNC                   ; RSYNC ON ERROR
1761 011152                          EOPT:
1762 011152  004737  011360                  JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1763 011156                          SLVFIN:
1764 011156  104407                          CKSWR                   ;GO CHECK SWR
1765 011160  005737  001452                  TST     MSTER           ;WERE WE STARTED AS MASTER?
1766 011164  001055                          BNE     EOPTA           ;BR IF NOT
1767 011166  005337  001454                  DEC     ICOUNT          ;COUNT TEST PASS
1768 011172  001052                          BNE     EOPTA           ;BR IF NOT DUE FOR 'END PASS' MSG
1769 011174  012737  000144  001454          MOV     #144,ICOUNT     ;RESET PASS COUNT TO  100 ITERATIONS
```

```
1771                                    .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

011202                                  $EOP:
011202  000240                                NOP
011204  005037  001102                      CLR     $TSTNM          ;;ZERO THE TEST NUMBER
011210  005237  001100                      INC     $PASS           ;;INCREMENT THE PASS NUMBER
011214  042737  100000  001100              BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
011222  005327                              DEC     (PC)+           ;;LOOP?
011224  000001                  $EOPCT: .WORD   1
011226  003022                              BGT     $DOAGN          ;;YES
011230  012737                              MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
011232  000001                  $ENDCT: .WORD   1
011234  011224                              $EOPCT
011236  104401  011303                      TYPE    ,$ENDMG         ;;TYPE "END PASS #"
011242  013746  001100                      MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
011246  104405                              TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
011250  104401  011300                      TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
011254  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
011260  001405                              BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
011262  000005                              RESET                   ;;CLEAR THE WORLD
011264  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
011266  000240                              NOP                     ;;SAVE ROOM
011270  000240                              NOP                     ;;FOR
011272  000240                              NOP                     ;;ACT11
011274                                  $DOAGN:
011274  000137                              JMP     @(PC)+          ;;RETURN
011276  011320                  $RTNAD: .WORD   EOPTA
011300     377      377      000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
011303     015      012      105  $ENDMG: .ASCIZ  <15><12>/END PASS #/
011306     116      104      040
011311     120      101      123
011314     123      040      043
011317     000
1772 011320  012737  001000  001444  EOPTA:  MOV     #1000,TIME      ;SET UP A COUNT
1773 011326  005337  001444          EOPTB:  DEC     TIME            ;STALL FOR OTHER CPU TO BECOME SLAVE
1774 011332  001375                          BNE     EOPTB           ;UNTIL TIME=0
1775 011334  000137  002760                  JMP     MINIT1          ;NOW BECOME MASTER
```

```
                                        .SBTTL
                                        .SBTTL  PROGRAM SUBROUTINES

                                        ;;*************************************************************************
                                        ;THIS ROUTINE SETS THE PRIORITY LEVEL FOR NO INTERRUPT -
                                        ;SETS UP THE  INTERRUPT TO RETURN ON INTERRUPT
                                        ;TO THE ADDRESS INDICATED ((R5)) BY THE CALL +2
                                        ;;*************************************************************************
011340  004737  014050          SETVEC: JSR     PC,CPUHI        ;SET UP FOR NO INTERRUPT
011344  012577  170054                  MOV     (R5)+,@DRVCT0   ;SET UP INTR RETURN ADRS
011350  012777  000200  170050          MOV     #200,@DRVCT2    ;KEEP PRIORITY LEVEL AT TOP ON INTR
011356  000205                          RTS     R5              ;EXIT

                                        ;;*************************************************************************
                                        ;THIS ROUTINE RESTORES THE  INTERRUPT VECTOR TO A HALT
                                        ;AND RAISES THE PRIORITY LEVEL
                                        ;;*************************************************************************
011360  013777  001426  170036  RSTVEC: MOV     DRVCT2,@DRVCT0  ;POINT VECTOR TO HALT
011366  005077  170034                  CLR     @DRVCT2         ;SET UP HALT
011372  004737  014050                  JSR     PC,CPUHI        ;RAISE PRIORITY LEVEL
011376  000207                          RTS     PC              ;EXIT

                                        ;;*************************************************************************
                                        ;THIS ROUTINE CLEARS THE 'DBUF' BEFORE A 'DATI' XFER
                                        ;THE # OF LOCATIONS IN 'DBUF' TO BE CLEARED IS SPECIFIED BY
                                        ;THE VALUE IN THE CALL +2 - WHEN ALL CLEARED THE RETURN IS TO
                                        ;THE CALL +4
                                        ;;*************************************************************************
011400  012500                  CLRBUF: MOV     (R5)+,R0        ;GET THE LOC COUNT
011402  012701  016236                  MOV     #DBUF,R1        ;GET 1ST ADRS
011406  005021                  1$:     CLR     (R1)+           ;CLR MEM LOC
011410  005200                          INC     R0              ;COUNT LOC
011412  001375                          BNE     1$              ;UNTIL ALL DONE
011414  000205                          RTS     R5              ;EXIT
```

```
                                        ;;*****************************************************************
                                        ;THIS ROUTINE LOADS 'DBUF' WITH A FLOATING ZERO/ONE PATTERN
                                        ;THE # OF LOCATIONS IN 'DBUF' TO BE LOADED IS SPECIFIED BY
                                        ;THE VALUE IN THE CALL +2 - WHEN ALL LOADED THE RETURN IS TO CALL +4
                                        ;;*****************************************************************
011416  012500                  LDBUF:  MOV     (R5)+,R0        ;GET LOC COUNT
011420  012701  016236                  MOV     #DBUF,R1        ;GET 1ST ADRS
011424  012702  177776          1$:     MOV     #177776,R2      ;SET UP FLOATING ZERO PTRN
011430  005003                          CLR     R3              ;R3 SAYS WHEN TO SHIFT PTRN
011432  010221                  2$:     MOV     R2,(R1)+        ;LOAD MEM WITH PTRN
011434  005200                          INC     R0              ;COUNT LOC
011436  001001                          BNE     3$              ;BR IF MORE
011440  000205                          RTS     R5              ;EXIT
011442  005102                  3$:     COM     R2              ;CONVERT PTRN TO FLOATING 1
011444  005103                          COM     R3              ;SHOULD WE SHIFT?
011446  001371                          BNE     2$              ;BR IF NOT - THIS WILL BE A FLOATING 1
011450  006302                          ASL     R2              ;FLOAT ZERO LEFT
011452  005202                          INC     R2              ;KEEP LSB SET
011454  103363                          BCC     1$              ;GO RESET FLOATING PATRN
011456  000765                          BR      2$              ;GO LOAD NEXT PATRN
```

```
1834                                    .SBTTL  SYSMAC ROUTINES
1835
1836                                .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
                                    ;*

011460  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
011464  100002                         BPL     1$              ;;BR IF YES
011466  000000                         HALT                    ;;HALT HERE IF NO TERMINAL
011470  000407                         BR      3$              ;;LEAVE
011472  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
011474  017600  000002                 MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
011500  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
011502  001005                         BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
011504  005726                         TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
011506  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
011510  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
011514  000002                         RTI                     ;;RETURN
011516  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
011522  001430                         BEQ     8$
011524  122716  000200                 CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
011530  001006                         BNE     5$
011532  005726                         TST     (SP)+           ;;POP  <CR><LF> EQUIV
011534  104401                         TYPE                    ;;TYPE A CR AND LF
011536  001165                         $CRLF
011540  105037  011756                 CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
011544  000755                         BR      2$              ;;GET NEXT CHARACTER
011546  004737  011630          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
011552  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
011556  001350                         BNE     2$              ;;IF NO GO GET NEXT CHAR.
011560  013746  001154                 MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
011564  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
011570  002770                         BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
011572  004737  011630                 JSR     PC,$TYPEC       ;;GO TYPE A NULL
011576  105337  011756                 DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
011602  000770                         BR      7$              ;;LOOP

                                    ;HORIZONTAL TAB PROCESSOR

011604  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
011610  004737  011630          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
011614  132737  000007  011756         BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
011622  001372                         BNE     9$              ;;TAB STOP
```

```
011624  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
011626  000724                          BR      2$              ;;GET NEXT CHARACTER
011630                          $TYPEC:
011630  105777  167310                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?          ;MJD001
011634  100022                          BPL     10$             ;;BR IF NOT                     ;MJD001
011636  017746  167304                  MOV     @$TKB,-(SP)     ;;GET CHAR                      ;MJD001
011642  042716  177600                  BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS         ;MJD001
011646  122716  000023                  CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF                 ;MJD001
011652  001012                          BNE     102$            ;;BR IF NOT                     ;MJD001
011654                          101$:                                                           ;MJD001
011654  105777  167264                  TSTB    @$TKS           ;;WAIT FOR CHAR                 ;MJD001
011660  100375                          BPL     101$                                            ;MJD001
011662  117716  167260                  MOVB    @$TKB,(SP)      ;;GET CHAR                      ;MJD001
011666  042716  177600                  BIC     #177600,(SP)    ;;STRIP IT                      ;MJD001
011672  122716  000021                  CMPB    #$XON,(SP)      ;;WAS IT XON?                   ;MJD001
011676  001366                          BNE     101$            ;;BR IF NOT                     ;MJD001
011700                          102$:                                                           ;MJD001
011700  005726                          TST     (SP)+           ;;FIX STACK                     ;MJD001
011702                          10$:                                                            ;MJD001
011702  105777  167242                  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
011706  100375                          BPL     10$                                             ;MJD001
011710  126627  000002  000021          CMPB    2(SP),#$XON     ;;IS CHARACTER A RANDOM XON?    ;RAN001
011716  001420                          BEQ     $TYPEX          ;;BRANCH IF YES                 ;RAN001
011720  116677  000002  167224          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
011726  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
011734  001003                          BNE     1$              ;;BRANCH IF NO
011736  105037  011756                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
011742  000406                          BR      $TYPEX          ;;EXIT
011744  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
011752  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
011754  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
011756  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
011760  000207                  $TYPEX: RTS     PC
```

```
 1838                                   .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

011762  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
011766  116637  000001  012205          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
011774  112637  012207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
012000  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
012004  000406                          BR      $TYPON
012006  112737  000001  012205  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
012014  112737  000006  012207          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
012022  112737  000005  012204  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
012030  010346                          MOV     R3,-(SP)        ;;SAVE R3
012032  010446                          MOV     R4,-(SP)        ;;SAVE R4
012034  010546                          MOV     R5,-(SP)        ;;SAVE R5
012036  113704  012207                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
012042  005404                          NEG     R4
012044  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
012050  110437  012206                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
012054  113704  012205                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
012060  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
012064  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
012066  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
012070  000404                          BR      3$              ;;GO DO MSB
012072  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
012074  006105                          ROL     R5
012076  006105                          ROL     R5
012100  010503                          MOV     R5,R3
012102  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
012104  105337  012206                  DECB    $OMODE          ;;TYPE THIS DIGIT?
012110  100016                          BPL     7$              ;;BR IF NO
012112  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
012116  001002                          BNE     4$              ;;TEST FOR 0
012120  005704                          TST     R4              ;;SUPPRESS THIS 0?
012122  001403                          BEQ     5$              ;;BR IF YES
012124  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
```

```
012126  052703  000060            BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
012132  052703  000040      5$:   BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
012136  110337  012202            MOVB    R3,8$           ;;SAVE FOR TYPING
012142  104401  012202            TYPE    ,8$             ;;GO TYPE THIS DIGIT
012146  105337  012204      7$:   DECB    $OCNT           ;;COUNT BY 1
012152  003347                    BGT     2$              ;;BR IF MORE TO DO
012154  002402                    BLT     6$              ;;BR IF DONE
012156  005204                    INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
012160  000744                    BR      2$              ;;GO DO THE LAST DIGIT
012162  012605              6$:   MOV     (SP)+,R5        ;;RESTORE R5
012164  012604                    MOV     (SP)+,R4        ;;RESTORE R4
012166  012603                    MOV     (SP)+,R3        ;;RESTORE R3
012170  016666  000002  000004    MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
012176  012616                    MOV     (SP)+,(SP)
012200  000002                    RTI                     ;;RETURN
012202     000              8$:   .BYTE   0               ;;STORAGE FOR ASCII DIGIT
012203     000                    .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
012204     000        $OCNT:      .BYTE   0               ;;OCTAL DIGIT COUNTER
012205     000        $OFILL:     .BYTE   0               ;;ZERO FILL SWITCH
012206  000000        $OMODE:     .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
 1840                                   .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

 012210                                 $TYPDS:
 012210  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 012212  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 012214  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 012216  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 012220  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 012222  012746  020200                 MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
 012226  016605  000020                 MOV     20(SP),R5       ;;GET THE INPUT NUMBER
 012232  100004                         BPL     1$              ;;BR IF INPUT IS POS.
 012234  005405                         NEG     R5              ;;MAKE THE BINARY NUMBER POS.
 012236  112766  000055  000001         MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
 012244  005000                 1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
 012246  012703  012424                 MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
 012252  112723  000040                 MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
 012256  005002                 2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
 012260  016001  012414                 MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
 012264  160105                 3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
 012266  002402                         BLT     4$              ;;BR IF DONE
 012270  005202                         INC     R2              ;;INCREASE THE BCD DIGIT BY 1
 012272  000774                         BR      3$
 012274  060105                 4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
 012276  005702                         TST     R2              ;;CHECK IF BCD DIGIT=0
 012300  001002                         BNE     5$              ;;FALL THROUGH IF 0
 012302  105716                         TSTB    (SP)            ;;STILL DOING LEADING 0'S?
 012304  100407                         BMI     7$              ;;BR IF YES
 012306  106316                 5$:     ASLB    (SP)            ;;MSD?
 012310  103003                         BCC     6$              ;;BR IF NO
 012312  116663  000001  177777         MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
 012320  052702  000060         6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
 012324  052702  000040         7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 012330  110223                         MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 012332  005720                         TST     (R0)+           ;;JUST INCREMENTING
 012334  020027  000010                 CMP     R0,#10          ;;CHECK THE TABLE INDEX
 012340  002746                         BLT     2$              ;;GO DO THE NEXT DIGIT
 012342  003002                         BGT     8$              ;;GO TO EXIT
 012344  010502                         MOV     R5,R2           ;;GET THE LSD
 012346  000764                         BR      6$              ;;GO CHANGE TO ASCII
 012350  105726                 8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
 012352  100003                         BPL     9$              ;;BR IF NO
 012354  116663  177777  177776         MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
 012362  105013                 9$:     CLRB    (R3)            ;;SET THE TERMINATOR
 012364  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
 012366  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
 012370  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
 012372  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
```

```
012374  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
012376  104401  012424                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
012402  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
012410  012616                          MOV     (SP)+,(SP)
012412  000002                          RTI                     ;;RETURN TO USER
012414  023420                  $DTBL:  10000.
012416  001750                          1000.
012420  000144                          100.
012422  000012                          10.
012424                          $DBLK:  .BLKW   4
```

ERROR HANDLER ROUTINE

```
1842                                    .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

012434                                  $ERROR:
012434  104407                                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
012436  104407                                  CKSWR           ;GO LOOK FOR SWR CHANGE
012440  105237  001103                7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
012444  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
012446  013777  001102  166466          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
012454  032777  002000  166456          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
012462  001402                          BEQ     1$              ;;NO - SKIP
012464  104401  001160                  TYPE    .$BELL          ;;RING BELL
012470  005237  001112                1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
012474  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
012500  162737  000002  001116          SUB     #2,$ERRPC
012506  117737  166404  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
012514  032777  020000  166416          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
012522  001004                          BNE     20$             ;;SKIP TYPEOUTS
012524  004737  012550                  JSR     PC,SWRCK        ;;GO TO USER ERROR ROUTINE
012530  104401  001165                  TYPE    .$CRLF
012534                                  20$:
012534  005777  166400                2$:     TST     @SWR            ;;HALT ON ERROR
012540  100002                          BPL     3$              ;;SKIP IF CONTINUE
012542  000000                          HALT                    ;;HALT ON ERROR!
012544  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
012546                                  3$:
012546  000002                          RTI                     ;;RETURN
1843                                    ;;**********************************************************************
1844                                    ;GO TYPE ERROR
1845                                    ;GO UPDATE SOFTWARE SWR IF 'CNTRL/G'
1846                                    ;;**********************************************************************
1847  012550  004737  012560        SWRCK:  JSR     PC,$ERRTYP      ;GO TYPE ERROR
1848  012554  104407                       CKSWR                   ;GO LOOK FOR SWR CHANGE
1849  012556  000207                       RTS     PC              ;RETURN TO ERROR HANDLER
```

```
 1651                                   .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.

012560                                  $ERRTYP:
012560  104401  001165                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012564  010046                                MOV     R0,-(SP)        ;;SAVE R0
012566  005000                                CLR     R0              ;;PICKUP THE ITEM INDEX
012570  153700  001114                        BISB    @#$ITEMB,R0
012574  001004                                BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                        ;;TYPE THE PC OF THE ERROR
012576  013746  001116                        MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;ERROR ADDRESS
012602  104402                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012604  000426                                BR      6$              ;;GET OUT
012606  005300                          1$:   DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
012610  006300                                ASL     R0              ;;      WORK FOR THE ERROR TABLE
012612  006300                                ASL     R0
012614  006300                                ASL     R0
012616  062700  001170                        ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
012622  012037  012632                        MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
012626  001404                                BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
012630  104401                                TYPE                    ;;TYPE THE "ERROR MESSAGE"
012632  000000                          2$:   .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
012634  104401  001165                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012640  012037  012650                  3$:   MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
012644  001404                                BEQ     5$              ;;SKIP TYPEOUT IF 0
012646  104401                                TYPE                    ;;TYPE THE "DATA HEADER"
012650  000000                          4$:   .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
012652  104401  001165                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012656  011000                          5$:   MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
012660  001004                                BNE     7$              ;;GO TYPE THE DATA
012662  012600                          6$:   MOV     (SP)+,R0        ;;RESTORE R0
012664  104401  001165                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012670  000207                                RTS     PC              ;;RETURN
012672                                  7$:
012672  013046                                MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
012674  104402                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012676  005710                                TST     (R0)            ;;IS THERE ANOTHER NUMBER?
012700  001770                                BEQ     6$              ;;BR IF NO
012702  104401  012710                        TYPE    ,8$             ;;TYPE TWO(2) SPACES
012706  000771                                BR      7$              ;;LOOP
012710    040     040     000         8$:   .ASCIZ  /  /              ;;TWO(2) SPACES
                                              .EVEN
```

```
 1853                                   .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

 012714                                 $SCOPE:
 012714  104407                                 CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
 012716  104407                                 CKSWR                   ;GO LOOK FOR SWR CHANGE
                                        ;@@@@@START OF CODE FOR THE XOR TESTER@@@@@
 012720  000416                         $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
 012722  013746  000004                         MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 012726  012737  012746  000004                 MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
 012734  005737  177060                         TST     @#177060        ;;TIME OUT ON XOR?
 012740  012637  000004                         MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 012744  000404                                 BR      $SVLAD          ;;GO TO THE NEXT TEST
 012746  022626                         5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
 012750  012637  000004                         MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 012754  000406                                 BR      $OVER           ;;LOOP ON THE PRESENT TEST
 012756                                 6$:;@@@@@END OF CODE FOR THE XOR TESTER@@@@@
 012756  105237  001102                 $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
 012762  011637  001106                         MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 012766  105037  001103                         CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
 012772  013777  001102  166142         $OVER:  MOV     $TSTNM,@#DISPLAY ;;DISPLAY TEST NUMBER
 013000  013716  001106                         MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 013004  000002                                 RTI                     ;;FIXES PS
```

```
 1855                                   .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.
013006  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
013014  001074                          BNE     15$             ;;BRANCH IF NO
013016  105777  166122                  TSTB    @$TKS           ;;CHAR THERE?
013022  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
013024  117746  166116                  MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
013030  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
013034  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
013040  001062                          BNE     15$             ;;NO, RETURN TO USER
013042  123727  001134  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
013050  001456                          BEQ     15$             ;;BRANCH IF YES

013052  104401  013543                  TYPE    .$CNTLG         ;;ECHO THE CONTROL-G (^G)
013056  104401  013550          $GTSWR: TYPE    .$MSWR          ;;TYPE CURRENT CONTENTS
013062  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
013066  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
013070  104401  013561                  TYPE    .$MNEW          ;;PROMPT FOR NEW SWR
013074  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
013076  005046                          CLR     -(SP)           ;;THE NEW SWR
013100  105777  166040          7$:     TSTB    @$TKS           ;;CHAR THERE?
013104  100375                          BPL     7$              ;;IF NOT TRY AGAIN

013106  117746  166034                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
013112  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII


013116  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
013122  001005                          BNE     10$             ;;BRANCH IF NOT
013124  104401  013536                  TYPE    .$CNTLU         ;;YES, ECHO CONTROL-U (^U)
013130  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
013134  000757                          BR      19$             ;;LET'S TRY IT AGAIN


013136  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
013142  001022                          BNE     16$             ;;BRANCH IF NO
013144  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
013150  001403                          BEQ     11$             ;;BRANCH IF YES
013152  016677  000002  165760          MOV     2(SP),@SWR      ;;SAVE NEW SWR
013160  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
013164  104401  001165          14$:    TYPE    .$CRLF          ;;ECHO <CR> AND <LF>
013170  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
013176  001003                          BNE     15$             ;;BRANCH IF NOT
013200  012777  000100  165736          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
013206  000002                  15$:    RTI                     ;;RETURN
013210  004737  011630          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
013214  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
013220  002420                          BLT     18$             ;;BRANCH IF YES
```

```
013222  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
013226  003015                          BGT     18$             ;;BRANCH IF YES
013230  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
013234  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
013240  001403                          BEQ     17$             ;;BRANCH IF YES
013242  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
013244  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
013246  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
013250  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
013254  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
013260  000707                          BR      7$              ;;GET THE NEXT ONE
013262  104401  001164          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
013266  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
                                ;

013270  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
013272  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
013300  105777  165640          1$:     TSTB    @$TKS           ;;WAIT FOR
013304  100375                          BPL     1$              ;;A CHARACTER
013306  117766  165634  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
013314  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
013322  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
013330  001013                          BNE     3$              ;;BRANCH IF NO
013332  105777  165606          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
013336  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
013340  117746  165602                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
013344  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
013350  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
013354  001366                          BNE     2$              ;;IF NOT DISCARD IT
013356  000750                          BR      1$              ;;YES, RESUME
013360  026627  000004  000021  3$:     CMP     4(SP),#$XON     ;;IS IT A RANDOM XON?           ;RAN001
013366  001744                          BEQ     1$              ;;BRANCH IF YES                 ;RAN001
013370  026627  000004  000140          CMP     4(SP),#140      ;;IS IT UPPER CASE?
013376  002407                          BLT     4$              ;;BRANCH IF YES
013400  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
013406  003003                          BGT     4$              ;;BRANCH IF YES
013410  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
013416  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

013420  010346          $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
013422  012703  013526  1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
013426  022703  013536  2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
```

```
013432  101405                          BLOS    4$              ;;BR IF YES
013434  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
013436  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
013440  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
013444  001003                          BNE     3$              ;;SKIP IF NOT
013446  104401  001164          4$:     TYPE    ,$QUES          ;;TYPE A '?'
013452  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
013454  111337  013524          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
013460  104401  013524                  TYPE    ,9$
013464  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
013470  001356                          BNE     2$              ;;LOOP IF NOT RETURN
013472  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
013476  104401  001166                  TYPE    ,$LF            ;;TYPE A LINE FEED
013502  012603                          MOV     (SP)+,R3        ;;RESTORE R3
013504  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
013506  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
013514  012766  013526  000004          MOV     #$TTYIN,4(SP)
013522  000002                          RTI                     ;;RETURN
013524     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
013525     000                  .BYTE   0                       ;;TERMINATOR
013526                  $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
013536     136     125     015  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
013541     012     000
013543     136     107     015  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
013546     012     000
013550     015     012     123  $MSWR:  .ASCIZ  <15><12>/SWR = /
013553     127     122     040
013556     075     040     000
013561     040     040     116  $MNEW:  .ASCIZ  /  NEW = /
013564     105     127     040
013567     075     040     000
```

```
1857                                    .EVEN
1858                                    .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.

013572  010046                          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
013574  016600  000002                          MOV     2(SP),R0        ;;GET TRAP ADDRESS
013600  005740                                  TST     -(R0)           ;;BACKUP BY 2
013602  111000                                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
013604  022700  000026                          CMP     #$TERM,R0       ;;CHECK FOR OUT OF BOUNDS
013610  003002                                  BGT     .+6             ;;BR IF OK
013612  000000                                  HALT                    ;;OUT OF BOUNDS
013614  000776                                  BR      .-2             ;;HANGUP
013616  006300                                  ASL     R0              ;;POSITION FOR INDEXING
013620  016000  013640                          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
013624  000200                                  RTS     R0              ;;GO TO ROUTINE

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

013626  011646                          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
013630  016666  000004  000002                  MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
013636  000002                                  RTI                     ;;RESTORE THE PSW

                                        .SBTTL  TRAP TABLE

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

                                        ;       ROUTINE
                                        ;       -------
013640  013626                          $TRPAD: .WORD   $TRAP2
013642  011460                                  $TYPE   ;;CALL-TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
013644  012006                                  $TYPOC  ;;CALL-TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
013646  011762                                  $TYPOS  ;;CALL-TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
013650  012022                                  $TYPON  ;;CALL-TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
013652  012210                                  $TYPDS  ;;CALL-TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

013654  013056                                  $GTSWR  ;;CALL-GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

013656  013006                                  $CKSWR  ;;CALL-CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
013660  013270                                  $RDCHR  ;;CALL-RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
013662  013420                                  $RDLIN  ;;CALL-RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
1859 013664  014124                           RESYNC  ;;CALL-RSYNC    TRAP+12(104412) RESYNC PROGRAM
1860         000026                     $TERM=.-$TRPAD
1861
```

```
1863                                    ;;*****************************************************************
1864
1865                                    .SBTTL  INTERBOARD COMMUNICATION SUBROUTINES
1866                                            ;WTCMPN
1867                                    ;;*****************************************************************
1868
1869 013666  012737  010000  001436  WTCMPN: MOV     #10000,ABORT    ;SETUP ABORT TIMER
1870 013674  032777  001000  165516           BIT     #1000,@CSR
1871 013702  001010                          BNE     2$              ;IF DSTATC IS ONE,WAIT FOR 0
1872
1873
1874 013704  032777  001000  165506  1$:     BIT     #1000,@CSR      ;WAIT FOR DSTATC=1
1875 013712  001017                          BNE     4$              ;GO AHEAD;IT HAS COME
1876 013714  005337  001436                  DEC     ABORT
1877 013720  001371                          BNE     1$              ;IF NOT STUCK IN WAIT LOOP,
1878                                                                  ;KEEP WAITING
1879 013722  000407                          BR      3$              ;WAITED TOO LONG;ABORT PROGRAM
1880 013724  032777  001000  165466  2$:     BIT     #1000,@CSR      ;WAIT DSTATC=0
1881 013732  001407                          BEQ     4$              ;GO AHEAD;IT HAS COME
1882 013734  005337  001436                  DEC     ABORT           ;KEEP WAITING
1883 013740  001371                          BNE     2$
1884
1885 013742  011637  001440          3$:     MOV     (SP),TESTPC     ;SAVE LOCATION WHERE PROGRAM
1886                                                                  ;ORIGINATED
1887 013746  104016                  104000+16
1888 013750  104412                          RSYNC                   ;RESYNCHRONIZE BOTH PROGRAMS
1889                                                                  ;FROM START
1890 013752  000207                  4$:     RTS     PC
1891
1892
1893                                    ;;*****************************************************************
1894                                            ;GOCMPN
1895                                    ;;*****************************************************************
1896
1897 013754                           GOCMPN:
1898 013754  032777  000002  165436           BIT     #2,@CSR         ;IS F1 CLR?
1899 013762  001415                          BEQ     1$              ;IF SO,SET F1
1900 013764  017737  165430  001446          MOV     @CSR,TEMP
1901 013772  042737  100000  001446          BIC     #BIT15,TEMP
1902 014000  042737  000002  001446          BIC     #2,TEMP         ;F1 IS SET,CLR IT
1903 014006  013777  001446  165404          MOV     TEMP,@CSR
1904 014014  000207                          RTS     PC
1905 014016  017737  165376  001446  1$:     MOV     @CSR,TEMP
1906 014024  042737  100000  001446          BIC     #BIT15,TEMP
1907 014032  052737  000002  001446          BIS     #2,TEMP
1908 014040  013777  001446  165352          MOV     TEMP,@CSR
1909 014046  000207                          RTS     PC
1910
1911
```

```
                                .SBTTL  CPU PRIORITY SUBROUTINES
                                ;;*****************************************************************
                                ;CPUHI
                                ;;*****************************************************************
014050                          CPUHI:
014050  023727  001434  000126          CMP     BOARD,#126      ;IS THIS CPU LSI?
014056  001003                        BNE     1$              ;NO,IT IS UNIBUS
014060  106427                        106427
014062  000200                        200
014064  000207                        RTS     PC
                                
014066  012737  000340  177776  1$:     MOV     #340,PS         ;SET HIGH PRIOR. IN UNIBUS
014074  000207                        RTS     PC
                                
                                ;;*****************************************************************
                                ;CPULO
                                ;;*****************************************************************
014076                          CPULO:
014076  023727  001434  000126          CMP     BOARD,#126
014104  001003                        BNE     1$
014106  106427                        106427
014110  000000                        0
014112  000207                        RTS     PC
014114  012737  000000  177776  1$:     MOV     #0,PS
014122  000207                        RTS     PC
```

```
1939                                    .SBTTL  INTERPROCESSOR PROGRAM RESYNCHRONIZATION  ROUTINE
1940                                    ;;**********************************************************************
1941                                                    ;RSYNC
1942                                    ;;**********************************************************************
1943
1944 014124  004737  011360            RESYNC: JSR     PC,RSTVEC
1945
1946                                            ;MOVE TESTPC VALUE TO BDR FOR COMPANION
1947
1948 014130  013777  001440  165264            MOV     TESTPC,@BDR     ; SAVE ERROR PC
1949
1950 014136  012706  001100                    MOV     #STACK,SP
1951 014142  005737  001452                    TST     MSTER           ;WAS THIS CPU STARTED AS MASTER?
1952 014146  001404                            BEQ     MRDELY          ;YES
1953 014150  012737  000017  001444  SLDELY: MOV     #15.,TIME       ;SLAVE CPU;WAIT FOR MASTER TO REACH ITS
1954                                                                    ;RESYNC ROUTINE
1955 014156  000403                            BR      WTSYNC
1956 014160  012737  000036  001444  MRDELY: MOV     #30.,TIME
1957 014166                            WTSYNC:
1958 014166  005001                    1$:     CLR     R1
1959 014170  005301                    2$:     DEC     R1
1960 014172  001376                            BNE     2$
1961 014174  005337  001444                    DEC     TIME
1962 014200  001372                            BNE     1$
1963 014202  000005                            RESET
1964 014204  104401  014212                    TYPE    ,65$            ;;TYPE ASCIZ STRING
     014210  000407                            BR      64$             ;;GET OVER THE ASCIZ
                                    ;;65$:  .ASCIZ  <CRLF>/*RESYNC.../<CRLF>
     014230                         64$:
1965 014230  000137  002742                    JMP     SYNCST
1966
1967                                    .SBTTL  ASCII MESSAGES
1968 014234       124       105       123  EM1:    .ASCII  /TEST 1    STATUS: MASTER/<CRLF>
     014237       124       040       061
     014242       040       040       040
     014245       040       123       124
     014250       101       124       125
     014253       123       072       040
     014256       115       101       123
     014261       124       105       122
     014264       200
1969 014265       123       114       101          .ASCIZ  /SLAVE FAILED TO ECHO DBR CONTENTS/
     014270       126       105       040
     014273       106       101       111
     014276       114       105       104
     014301       040       124       117
     014304       040       105       103
     014307       110       117       040
     014312       104       102       122
     014315       040       103       117
     014320       116       124       105
     014323       116       124       123
     014326       000
1970 014327       040       123       124  EM2:    .ASCII  / STATUS: MASTER/<CRLF>
     014332       101       124       125
     014335       123       072       040
     014340       115       101       123
```

F6


```
        014343  124  105  122
        014346  200
1971    014347  123  114  101          .ASCIZ  /SLAVE FAILED TO ECHO 'STAT' BITS/
        014352  126  105  040
        014355  106  101  111
        014360  114  105  104
        014363  040  124  117
        014366  040  105  103
        014371  110  117  040
        014374  047  123  124
        014377  101  124  047
        014402  040  102  111
        014405  124  123  000
1972    014410  040  123  124  EM3:    .ASCII  / STATUS: MASTER/<CRLF>
        014413  101  124  125
        014416  123  072  040
        014421  115  101  123
        014424  124  105  122
        014427  200
1973    014430  106  101  111          .ASCIZ  /FAILED TO INTR ON A 'DATI'/
        014433  114  105  104
        014436  040  124  117
        014441  040  111  116
        014444  124  122  040
        014447  117  116  040
        014452  101  040  047
        014455  104  101  124
        014460  111  047  000
1974    014463  040  123  124  EM4:    .ASCII  / STATUS: MASTER/<CRLF>
        014466  101  124  125
        014471  123  072  040
        014474  115  101  123
        014477  124  105  122
        014502  200
1975    014503  123  124  101          .ASCIZ  /STATUS ER ON 'DATI'/
        014506  124  125  123
        014511  040  105  122
        014514  040  117  116
        014517  040  047  104
        014522  101  124  111
        014525  047  000
1976    014527  040  123  124  EM5:    .ASCII  / STATUS: MASTER/<CRLF>
        014532  101  124  125
        014535  123  072  040
        014540  115  101  123
        014543  124  105  122
        014546  200
1977    014547  127  117  122          .ASCIZ  /WORD COUNT ER ON 'DATI'/
        014552  104  040  103
        014555  117  125  116
        014560  124  040  105
        014563  122  040  117
        014566  116  040  047
        014571  104  101  124
        014574  111  047  000
1978    014577  040  123  124  EM6:    .ASCII  / STATUS: MASTER/<CRLF>
        014602  101  124  125
```

```
                014605  123  072  040
                014610  115  101  123
                014613  124  105  122
                014616  200
1979            014617  102  125  106          .ASCIZ  /BUFFER ADRS ER ON 'DATI'/
                014622  106  105  122
                014625  040  101  104
                014630  122  123  040
                014633  105  122  040
                014636  117  116  040
                014641  047  104  101
                014644  124  111  047
                014647  000
1980            014650  040  123  124  EM7:    .ASCII  / STATUS: SLAVE/<CRLF>
                014653  101  124  125
                014656  123  072  040
                014661  123  114  101
                014664  126  105  200
1981            014667  106  101  111          .ASCIZ  /FAILED TO INTR ON A 'DATO'/
                014672  114  105  104
                014675  040  124  117
                014700  040  111  116
                014703  124  122  040
                014706  117  116  040
                014711  101  040  047
                014714  104  101  124
                014717  117  047  000
1982            014722  040  123  124  EM10:   .ASCII  / STATUS: SLAVE/<CRLF>
                014725  101  124  125
                014730  123  072  040
                014733  123  114  101
                014736  126  105  200
1983            014741  123  124  101          .ASCIZ  /STATUS ER ON 'DATO'/
                014744  124  125  123
                014747  040  105  122
                014752  040  117  116
                014755  040  047  104
                014760  101  124  117
                014763  047  000
1984            014765  040  123  124  EM11:   .ASCII  / STATUS: SLAVE/<CRLF>
                014770  101  124  125
                014773  123  072  040
                014776  123  114  101
                015001  126  105  200
1985            015004  127  117  122          .ASCIZ  /WORD COUNT ER ON 'DATO'/
                015007  104  040  103
                015012  117  125  116
                015015  124  040  105
                015020  122  040  117
                015023  116  040  047
                015026  104  101  124
                015031  117  047  000
1986            015034  040  123  124  EM12:   .ASCII  / STATUS: SLAVE/<CRLF>
                015037  101  124  125
                015042  123  072  040
                015045  123  114  101
                015050  126  105  200
```

```
1987 015053  102  125  106          .ASCIZ  /BUFFER ADRS ER ON 'DATO'/
     015056  106  105  122
     015061  040  101  104
     015064  122  123  040
     015067  105  122  040
     015072  117  116  040
     015075  047  104  101
     015100  124  117  047
     015103  000
1988 015104  040  123  124  EM13:  .ASCII  / STATUS: SLAVE/<CRLF>
     015107  101  124  125
     015112  123  072  040
     015115  123  114  101
     015120  126  105  200
1989 015123  104  101  124          .ASCIZ  /DATA ER ON 'DATO'/
     015126  101  040  105
     015131  122  040  117
     015134  116  040  047
     015137  104  101  124
     015142  117  047  000
1990 015145  123  124  101  EM14:  .ASCII  /STATUS: SLAVE/<CRLF>
     015150  124  125  123
     015153  072  040  123
     015156  114  101  126
     015161  105  200
1991 015163  122  105  101          .ASCIZ  /READY BIT WAS SET - IT SHOULD BE CLEAR/<CRLF>
     015166  104  131  040
     015171  102  111  124
     015174  040  127  101
     015177  123  040  123
     015202  105  124  040
     015205  055  040  111
     015210  124  040  123
     015213  110  117  125
     015216  114  104  040
     015221  102  105  040
     015224  103  114  105
     015227  101  122  200
     015232  000
1992 015233  040  123  124  EM15:  .ASCII  / STATUS: MASTER/<CRLF>
     015236  101  124  125
     015241  123  072  040
     015244  115  101  123
     015247  124  105  122
     015252  200
1993 013253  115  101  123          .ASCII  /MASTER CANNOT START COMMUNICATION WITH SLAVE-/
     015256  124  105  122
     015261  040  103  101
     015264  116  116  117
     015267  124  040  123
     015272  124  101  122
     015275  124  040  103
     015300  117  115  115
     015303  125  116  111
     015306  103  101  124
     015311  111  117  116
     015314  040  127  111
```

```
                015317  124  110  040
                015322  123  114  101
                015325  126  105  055
1994            015330  127  101  111          .ASCIZ  /WAITING FOR DSTATC /<CRLF>
                015333  124  111  116
                015336  107  040  106
                015341  117  122  040
                015344  104  123  124
                015347  101  124  103
                015352  040  200  000
1995            015355  123  124  125  EM16:   .ASCIZ  /STUCK WAITING FOR COMPANION FOR GO-AHEAD/<CRLF>
                015360  103  113  040
                015363  127  101  111
                015366  124  111  116
                015371  107  040  106
                015374  117  122  040
                015377  103  117  115
                015402  120  101  116
                015405  111  117  116
                015410  040  106  117
                015413  122  040  107
                015416  117  055  101
                015421  110  105  101
                015424  104  200  000
1996            015427  124  105  123  EM17:   .ASCII  /TEST 5 TIME OUT ERROR-/
                015432  124  040  065
                015435  040  124  111
                015440  115  105  040
                015443  117  125  124
                015446  040  105  122
                015451  122  117  122
                015454  055
1997            015455  123  114  101          .ASCIZ  /SLAVE SHOULD NOT HAVE  INTERRUPTED/<CRLF>
                015460  126  105  040
                015463  123  110  117
                015466  125  114  104
                015471  040  116  117
                015474  124  040  110
                015477  101  126  105
                015502  040  040  111
                015505  116  124  105
                015510  122  122  125
                015513  120  124  105
                015516  104  200  000
1998
1999            015521  040  123  124  EM20:   .ASCII  / STATUS: MASTER/<CRLF>
                015524  101  124  125
                015527  123  072  040
                015532  115  101  123
                015535  124  105  122
                015540  200
2000            015541  115  101  123          .ASCII  /MASTER IS LEFT WAITING FOR INTERRUPT TO OCCUR/
                015544  124  105  122
                015547  040  111  123
                015552  040  114  105
                015555  106  124  040
                015560  127  101  111
```

```
        015563  124  111  116
        015566  107  040  106
        015571  117  122  040
        015574  111  116  124
        015577  105  122  122
        015602  125  120  124
        015605  040  124  117
        015610  040  117  103
        015613  103  125  122
2001    015616  126  111  101          .ASCIZ  /VIA ATTN SET/<CRLF>
        015621  040  101  124
        015624  124  116  040
        015627  123  105  124
        015632  200  000
2002    015634  124  105  123  EM21:   .ASCII  /TEST 2   STATUS: MASTER/<CRLF>
        015637  124  040  062
        015642  040  040  040
        015645  123  124  101
        015650  124  125  123
        015653  072  040  115
        015656  101  123  124
        015661  105  122  200
2003    015664  105  122  122          .ASCIZ  /ERROR BIT IN MASTER CSR NOT CLEAR/
        015667  117  122  040
        015672  102  111  124
        015675  040  111  116
        015700  040  115  101
        015703  123  124  105
        015706  122  040  103
        015711  123  122  040
        015714  116  117  124
        015717  040  103  114
        015722  105  101  122
        015725  000
2004    015726  102  125  122  EM22:   .ASCIZ  /BURST DATA LATE BIT OF EIR WAS NOT SET/<CRLF>
        015731  123  124  040
        015734  104  101  124
        015737  101  040  114
        015742  101  124  105
        015745  040  102  111
        015750  124  040  117
        015753  106  040  105
        015756  111  122  040
        015761  127  101  123
        015764  040  116  117
        015767  124  040  123
        015772  105  124  200
        015775  000
2005
2006    015776  200  124  105  MSG1:   .ASCIZ  <CRLF>  /TEST 3-BLOCK MODE XFERS/<CRLF>
        016001  123  124  040
        016004  063  055  102
        016007  114  117  103
        016012  113  040  115
        016015  117  104  105
        016020  040  130  106
        016023  105  122  123
```

```
                016026  200     000
2007            016030  200     124     105     MSG2:   .ASCIZ  <CRLF>  /TEST 4-BURST MODE XFERS/<CRLF>
                016033  123     124     040
                016036  064     055     102
                016041  125     122     123
                016044  124     040     115
                016047  117     104     105
                016052  040     130     106
                016055  105     122     123
                016060  200     000
2008            016062  200     124     105     MSG3:   .ASCIZ  <CRLF>/TEST 5-BURST DATA LATE/<CRLF>
                016065  123     124     040
                016070  065     055     102
                016073  125     122     123
                016076  124     040     104
                016101  101     124     101
                016104  040     114     101
                016107  124     105     200
                016112  000
2009            016113  105     122     122     DH1:    .ASCIZ  /ERRPC   BUSADR  EXPCT   RCVD/
                016116  120     103     040
                016121  040     040     102
                016124  125     123     101
                016127  104     122     040
                016132  040     105     130
                016135  120     103     124
                016140  040     040     040
                016143  122     103     126
                016146  104     000
2010            016150  105     122     122     DH2:    .ASCIZ  /ERRPC   MEMADR  EXPCT   RCVD/
                016153  120     103     040
                016156  040     040     115
                016161  105     115     101
                016164  104     122     040
                016167  040     105     130
                016172  120     103     124
                016175  040     040     040
                016200  122     103     126
                016203  104     000
2011            016205  105     122     122     DH3:    .ASCIZ  /ERRPC/
                016210  120     103     000
2012                                            .EVEN
2013            016214  001116  001122  001124  DT1:    $ERRPC,$BDADR,$GDDAT,$BDDAT,0
                016222  001126  000000
2014            016226  001116  000000          DT2:    $ERRPC,0
2015            016232  001440  000000          DT3:    TESTPC,0
2016                                            ;;**********************************************************************
2017                                            ;'DBUF' IS THE WORKING AREA IN MEM FOR ALL NPR OPERATIONS
2018                                            ;;**********************************************************************
2019            016236  000000                  DBUF:   0               ;1ST ADRS OF DATA BUFFER
2020                    000001                          .END
```

SYMBOL TABLE

```
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  011400
CNT     001430
CPUHI   014050
CPULO   014076
CR    = 000015
CRLF  = 000200
CSR     001420
DBUF    016236
DDISP = 177570
DH1     016113
DH2     016150
DH3     016205
DISPLA  001142
DISPRE  000174
DRVCT0  001424
DRVCT2  001426
DRVECT  001412
DSWR  = 177570
DT1     016214
DT2     016226
DT3     016232
EMTVEC= 000030
EM1     014234
EM10    014722
EM11    014765
EM12    015034
EM13    015104
EM14    015145
EM15    015233
EM16    015355
EM17    015427
EM2     014327
EM20    015521
EM21    015634
EM22    015726
EM3     014410
EM4     014463
EM5     014527
EM6     014577
EM7     014650
EOPT    011152
EOPTA   011320
EOPTB   011326
ERROR = 104000
ERRVEC= 000004
GOCMPN  013754
GTSWR = 104406
HT    = 000011
ICOUNT  001454
INTADR  001410
IOTVEC= 000020
LDBUF   011416
LF    = 000012
MINIT1  002760
MINIT2  003226
MORS    002524
MRDELY  014160
MSBRD   004550
MSG1    015776
MSG2    016030
MSG3    016062
MSTER   001452
MTPS  = 106427
MTST1   003106
MTST2   003526
MTST3   003652
MTST4   004620
MTST5   005502
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  014124
RPRAM   001464
RSTVEC  011360
RSYNC = 104412
RTRN    003462
R6    =%000006
R7    =%000007
SAVE    001450
SCOPE = 000004
SETUP2  001704
SETUP3  001730
SETVEC  011340
SINIT1  006162
SINIT2  006374
SLBRD   007420
SLDELY  014150
SLVFIN  011156
SL1STR  006330
STACK = 001100
START   001522
START1  001506
START2  001514
STKLMT= 177774
STST1   006320
STST2   006512
STST3   006572
STST4   007506
STST5   010404
SWR     001140
SWRCK   012550
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  013666
WTSYNC  014166
XMITMO  001472
XPRAM   001456
$AUTOB  001134
$BDADR  001122
$BDDAT  001126
$BELL   001160
$CHARC  011756
$CKSWR  013006
$CMTAG  001100
$CM3  = 000000
$CNTLG  013543
$CNTLU  013536
$CRLF   001165
$DBLK   012424
$DOAGN  011274
$DTBL   012414
$ENDAD  011264
$ENDCT  011232
$ENDMG  011303
$ENULL  011300
$EOP    011202
$EOPCT  011224
$ERFLG  001103
$ERMAX  001115
$ERROR  012434
$ERRPC  001116
$ERRTB  001170
$ERRTY  012560
$ERTTL  001112
$FILLC  001156
$FILLS  001155
$GDADR  001120
$GDDAT  001124
$GET42  011254
$GTSWR  013056
$HD   = 000003
$ICNT   001104
$INTAG  001135
$ITEMB  001114
$LF     001166
$LPADR  001106
$LPERR  001110
$MNEW   013561
$MSWR   013550
$NULL   001154
$OCNT   012204
$OMODE  012206
$OVER   012772
$PASS   001100
$QUES   001164
$RDCHR  013270
$RDLIN  013420
$RDSZ = 000010
$RTNAD  011276
$SCOPE  012714
$SETUP= 000107
$STUP = 177777
$SVLAD  012756
$SWR  = 122000
$SWRMK= 000000
$TERM = 000026
$TKB    001146
$TKS    001144
$TN   = 000001
$TPB    001152
$TPFLG  001157
$TPS    001150
$TRAP   013572
$TRAP2  013626
$TRP  = 000013
$TRPAD  013640
$TSTNM  001102
$TTYIN  013526
$TYPDS  012210
$TYPE   011460
$TYPEC  011630
$TYPEX  011760
$TYPOC  012006
$TYPON  012022
$TYPOS  011762
$XOFF = 000023
$XON  = 000021
$XTSTR  012720
$$GET4= 000000
$OFILL  012205
```

M6

SYMBOL TABLE

. ABS. 016240 000
        000000 001
ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 65456 WORDS (0 PAGES)
DYNAMIC MEMORY: 20060 WORDS (77 PAGES)
ELAPSED TIME: 00:02:21
ÇZDRKB.BIN,CZDRKB.LST/-SP=SYSMAC.SML,CZDRKB.P11