```
.REM  [
```

IDENTIFICATION

PRODUCT CODE: AC-E8848-MC

PRODUCT NAME: CXBEAB0 M7855 BUS TESTER MODULE

PRODUCT DATE: SEPTEMBER 1978

MAINTAINER: DEC/X11 SUPPORT GROUP

TABLE OF CONTENTS

1.0 ABSTRACT

BEA IS AN IOMOD THAT CAN HANDLE FROM 1 TO 12 UNIBUS EXERCISERS. THE MODULE WILL HAVE THE UBE(S) DOING DATI(S), DATOB(S), DATIP(S), AND DATO(S), THEN CHECKS FOR CORRECT DATA TRANSFERS. THESE TRANSFERS ARE DONE FIRST ON AN NPR LEVEL AND INTERRUPTS WITH A BR7,THEN THE REQUEST IS SEQUENTIALLY LOWERED TO A BR4.

2.0 REQUIREMENTS

HARDWARE:

1 TO 12 UBE(S); WITH MORE THAN ONE UBE, ALL SHOULD HAVE W1 JUMPER EXCEPT THE UBE THAT IS THE FURTHEST FROM THE PROCESSOR ELECTRICALLY.

SOFTWARE:

STORAGE:: BEA REQUIRES:
1. DECIMAL WORDS: 1371
2. OCTAL WORDS: 02533
3. OCTAL BYTES: 5266

3.0 PASS DEFINITION

1 ITERATION CONSISTS OF SETTING UP ONE OR MORE UBE(S) TO DO DATA TRANSFERS, THEN CHECKING THOSE TRANSFERS. 1 PASS WILL EQUAL 12,000 ITERATIONS.

4.0 EXECUTION TIME

ONE PASS WILL RUN IN APPROX. 1 MINUTE, RUNNING ALONE ON AN 11/45.

5.0 CONFIGURATION REQUIREMENTS

DEFAULT PARAMETERS: DEVADR=170000; VECTOR=510; BR1=7, BR2=6; DEVCNT=1

REQUIRED PARAMENTERS: NONE

6.0 DEVICE/OPTION SETUP

1 TO 12 UBE(S); WITH MORE THAN ONE UBE, ALL SHOULD HAVE W1 JUMPER EXCEPT THE UBE THAT IS THE FURTHEST FROM THE PROCESSOR ELECTRICALLY.

7.0 TEST SEQUENCE

A. LOAD DEVICE INTR VECTORS AND GET READ AND WRITE BUFFER SIZES.

B. INITIALIZE REGISTERS.

C. GET PHYSICAL ADDRESSES FOR WRITE AND READ BUFFERS.

D. CLEAR DEVICE REGISTERS AND WRITE BUFFER AREA; AND SET UP READ BUFFER AREA.

E. CHECK FOR DEVICES AND IF NONE LEFT, GO TO H.

F. LOAD DEVICE REGISTERS FOR DESIRED TRANSFER.

G. GO BACK TO E.

H. SET OFF ALL DEVICES SIMULTANEOUSLY.

I. WAIT FOR THEIR INTERRUPTS.

J. IF NOT ALL DEVICES INTERRUPTED, GO BACK TO I.

K. CHECK THE BUFFER AREAS FOR CORRECT TRANSFERS.

L. ROTATE REQUEST LEVELS AND DATA PATTERNS.

M. IF THIS IS NOT THE FINAL ITERATION, GO BACK TO D.

N. INDICATE END OF PASS AND GO BACK TO B.

8.0 OPERATION OPTIONS

THE FIRST REGISTER ADDRESS OF EACH UBE PROCEEDS IN INCREMENTS OF 20. EVERY BIT OF DVID1 CORRESPONDS TO A PARTICULAR UBE ADDRESS BEING PRESENT; I.E., BIT 0 SET = UBE ADDRESS OF 170000, BIT 1 SET = UBE ADDRESS OF 170020, BIT 2 SET = UBE ADDRESS OF 170040, ETC.

THE UBE(S) CAN BE ON THE BUS IN ANY ADDRESSING SEQUENCE. THE ONLY REQUIREMENT IS THAT DVID1 REFLECTS THE ADDRESSES OF EXISTING UBE(S).

9.0 NON-STANDARD PRINTOUTS

ALL PRINTOUT HAVE THE STANDARD FORMATS DESCRIBED IN THE DEC/X11 DOCUMENT WITH THE FOLLOWING ADDITIONS PRINTED BELOW IT:

1. FOR DATA TRANSFER ERRORS (DATI/P OR DATO/B) THE CONTENTS OF THE FOLLOWING REGISTERS PRE-PRINTED AS WELL AS THE MEMORY LOCATION AND CONTENTS. THERE SHOULD BE A DISCREPANCY BETWEEN BEDB AND CONTENTS OF MEMORY ON ERRORS.

   BEDB BECC BEBA BECR1 BECR2 MEM-ADDR MEM-CONTENTS

2. FOR "INTERRUPTS ON ERROR - NOT ON DONE" THE ABOVE REGISTERS WILL ALSO BE PRINTED OUT EXCEPT DISREGARD THE MEMORY ADDRESS AND CONTENTS SINCE IT IS NOT PREVIOUSLY SET UP.

3. FOR "NOT ALL DEVICES INTERRUPTING" ERROR, ONLY TWO ADDITIONAL REGISTERS ARE PRINTED - DV AND MASK. DV IS A TEMPORARY STORAGE LOCATION FOR DVID1 AND MASK IS THE LOCATION INDICATING WHICH DEVICES INTERRUPTED. BIT COMPARISON OF THESE TWO REGISTERS WILL INDICATE WHICH DEVICES DID OR DID NOT INTERRUPT.

   DV MASK

```
                                IDMOD   <BEAB >,170000,510,7,6,,12000,73
                                        MODULE  140000,BEAB ,170000,510,7,6,,12000,73
                                        .TITLE  BEAB DEC/X11 SYSTEM EXERCISER MODULE
                                ;       DDXCOM  VERSION  6       23-MAY-78
                                        .LIST   BIN
                                ;*****************************************************************************
000000'                         BEGIN:
000000' 042502  041101      040 MODNAM: .ASCII  /BEAB / ;MODULE NAME.
000005'     000                 XFLAG:  .BYTE   0PEN            ;USED TO KEEP TRACK OF WBUFF USAGE
000006' 170000                  ADDR:   170000+0                ;1ST DEVICE ADDR.
000010' 000510                  VECTOR: 510+0                   ;1ST DEVICE VECTOR.
000012'     340                 BR1:    .BYTE   PRTY7+0         ;1ST BR LEVEL.
000013'     300                 BR2:    .BYTE   PRTY6+0         ;2ND BR LEVEL.
000014' 000001                  DVID1:  +1                      ;DEVICE INDICATOR 1.
000016' 000000                  SR1:    0PEN                    ;SWITCH REGISTER 1
000020' 000000                  SR2:    0PEN                    ;SWITCH REGISTER 2
000022' 000000                  SR3:    0PEN                    ;SWITCH REGISTER 3
000024' 000000                  SR4:    0PEN                    ;SWITCH REGISTER 4
                                ;*****************************************************************************
000026' 140000                  STAT:   140000                  ;STATUS WORD.
000030' 000410'                 INIT:   START                   ;MODULE START ADDR.
000032' 000224'                 SPOINT: MODSP                   ;MODULE STACK POINTER.
000034' 000000                  PASCNT: 0                       ;PASS COUNTER.
000036' 012000                  ICONT:  12000                   ;# OF ITERATIONS PER PASS=12000
000040' 000000                  ICOUNT: 0                       ;LOC TO COUNT ITERATIONS
000042' 000000                  SOFCNT: 0                       ;LOC TO SAVE TOTAL SOFT ERRORS
000044' 000000                  HRDCNT: 0                       ;LOC TO SAVE TOTAL HARD ERRORS
000046' 000000                  SOFPAS: 0                       ;LOC TO SAVE SOFT ERRORS PER PASS
000050' 000000                  HRDPAS: 0                       ;LOC TO SAVE HARD ERRORS PER PASS
000052' 000000                  SYSCNT: 0                       ;# OF SYS ERRORS ACCUMULATED
000054' 000000                  RANNUM: 0                       ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
000056'                         CONFIG:                         ;RESERVED FOR MONITOR USE
000056' 000000                  RES1:   0                       ;RESERVED FOR MONITOR USE
000060' 000000                  RES2:   0                       ;RESERVED FOR MONITOR USE
000062' 000000                  SVR0:   0PEN                    ;LOC TO SAVE R0.
000064' 000000                  SVR1:   0PEN                    ;LOC TO SAVE R1.
000066' 000000                  SVR2:   0PEN                    ;LOC TO SAVE R2.
000070' 000000                  SVR3:   0PEN                    ;LOC TO SAVE R3.
000072' 000000                  SVR4:   0PEN                    ;LOC TO SAVE R4.
000074' 000000                  SVR5:   0PEN                    ;LOC TO SAVE R5.
000076' 000000                  SVR6:   0PEN                    ;LOC TO SAVE R6.
000100' 000000                  CSRA:   0PEN                    ;ADDR OF CURRENT CSR.
000102'                         SRADR:                          ;ADDR OF GOOD DATA, OR
000102' 000000                  ACSR:   0PEN                    ;CONTENTS OF CSR.
000104'                         WASADR:                         ;ADDR OF BAD DATA, OR
000104' 000000                  ASTAT:  0PEN                    ;STATUS REG CONTENTS.
000106'                         ERRTYP:                         ;TYPE OF ERROR
000106' 000000                  ASB:    0PEN                    ;EXPECTED DATA.
000110' 000000                  AWAS:   0PEN                    ;ACTUAL DATA.
000112' 000534'                 RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
000114' 000000                  WDTO:   0PEN                    ;WORDS TO MEMORY PER ITERATION
000116' 000000                  WDFR:   0PEN                    ;WORDS FROM MEMORY PER ITERATION
000120' 000000                  INTR:   0PEN                    ;# OF INTERRUPTS PER ITERATION
000122' 000073                  IDNUM:  73                      ;MODULE IDENTIFICATION NUMBER=73
        000040                          .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
                                        .NLIST
```



```
                                        .WORD   0
                                        .LIST
                                        .ENDR
000224'                         MODSP:
                                ;*****************************************************************************
```

```
                                .ENABL  AMA
                        P1:     0       ;1ST DATA PATTERN
000224' 000000          P2:     137777  ;2ND DATA PATTERN
000226' 137777          P3:     0       ;3RD DATA PATTERN
000230' 000000          P4:     100000  ;4TH DATA PATTERN
000232' 100000          ROTCNT: 0       ;COUNTER FOR # OF TIMES DATA ROTATED
000234' 000000          SIMLGO: 170014  ;ADDR TO SET OFF ALL DEVS SIMULTANEOUSLY
000236' 170014          DV:     0       ;TEMP STORAGE FOR DVID1
000240' 000000          MASK:   0       ;USED TO DROP ANY DEV NOT INTERRUPTING
000242' 000000          MORE:   0       ;WORKING STORAGE FOR DVID1
000244' 000000          B12T15: 170000  ;USED TO CLEAR BITS 12 THRU 15
000246' 170000          BUFSZ:  740     ;SIZE FOR 1 DEV
000250' 000740                  360     ;SIZE FOR 2 DEVS
000252' 000360                  240     ;SIZE FOR 3 DEVS
000254' 000240                  170     ;SIZE FOR 4 DEVS
000256' 000170                  140     ;SIZE FOR 5 DEVS
000260' 000140                  120     ;SIZE FOR 6 DEVS
000262' 000120          DEVCNT: 0       ;TOTAL NUMBER OF DEVS
000264' 000000          DODI:   0       ;DATO OR DATI INDICATOR
000266' 000000          SVDODI: 0       ;STORAGE FOR DODI
000270' 000000          SAVR1:  0       ;LOC TO SAVE R1
000272' 000000          SAVR5:  0       ;LOC TO SAVE R5
000274' 000000          BYTCC:  0       ;CYCLE COUNT FOR A BYTE TRANSFER
000276' 000000          WCC:    0       ;CYCLE COUNT FOR A DATO TRANSFER
000300' 000000          RCC:    0       ;CYCLE COUNT FOR A DATI TRANSFER
000302' 000000          DOCNT:  0       ;# OF DEVS DOING A DATO/P(S)
000304' 000000          DICNT:  0       ;# OF DEVS DOING A DATI/P(S)
000306' 000000          EABIT:  0       ;USED TO SET EXTENDED MEMORY ADDR BITS
000310' 000000          RBUF:   0       ;SIZE OF BUFFER FOR DATI/P(S)
000312' 000000          WBUF:   0       ;SIZE OF BUFFER FOR DATO/P(S)
000314' 000000          WBUFVA: WRTOUT  ;VIRTUAL ADDR OF WRITE BUFFER
000316' 003362'         WBUFPA: OPEN    ;PHYSICAL ADDR OF WRITE BUFFER
000320' 000000          WBUFEA: OPEN    ;EXT MEM ADDR BITS SET BY MONITOR
000322' 000000          RBUFVA: RSEDIN  ;VIRTUAL ADDR OF READ BUFFER
000324' 004324'         RBUFPA: OPEN    ;PHYSICAL ADDR OF READ BUFFER
000326' 000000          RBUFEA: OPEN    ;EXT MEM ADDR BITS SET BY MONITOR
000330' 000000          RBUFSZ: 360     ;TOTAL SIZE OF BUFFER RESERVED FOR READS
000332' 000360          RBFADR: 0       ;USED TO POINT TO LOC WITHIN RBUFSZ
000334' 000000          ENDRBF: 0       ;USED FOR LAST ADDR(PHYSICAL) OF READ BUFF
000336' 000000          WBFADR: 0       ;USED TO POINT TO LOC WITHIN WBUFSZ
000340' 000000          ENDWBF: 0       ;USED FOR LAST ADDR(PHYSICAL) OF WRITE BUFF
000342' 000000          XFR1:   4100    ;FUN 2-DATI-INTR
000344' 004100          XFR2:   5500    ;FUN 2-DATOB-INTR
000346' 005500          XFR3:   22500   ;FUN 1-DATIP/NO ROT-INTR
000350' 022500          XFR4:   3100    ;FUN 1-DATO-INTR
000352' 003100          RQLVL:  60      ;USED TO SET REQUEST LEVELS OF DEVS
000354' 000060          ERBITS: 7740    ;USED TO CHECK  IF ANY ERROR BITS SET
000356' 007740          DVREGS:
000360'                 DR:     0       ;DEV DATA REG CONTENTS
000360' 000000          CC:     0       ;DEV CYCLE COUNT CONTENTS
000362' 000000          BA:     0       ;DEV ADDR REG CONTENTS
000364' 000000          CR1:    0       ;DEV CR1 REG CONTENTS
000366' 000000          CR2:    0       ;DEV CR2 REG CONTENTS
000370' 000000          BADMEM: BADR    ;BAD MEMORY ADDR
000372' 000374'
```



```
000374' 000000          BADR:   0       ;CONTENTS OF BAD MEMORY
000376' 177777                  177777
000400'                 DVMASK:
000400' 000240'         PTR1:   DV      ;THIS POINTER IS USED SO THE NEXT
000402' 000242'         PTR2:   MASK    ;TWO LOCATIONS WILL BE PRINTED OUT FOR
000404' 177777                  -1      ;AN ERR WHCIH DEVICES DID NOT INTERRUPT
000406' 000000          TOCNT:  0       ;TIME OUT COUNTER

000410' 012737  000361  000114' START:  MOV     #361,WDTO       ;AT LEAST 361 WORDS TO MEM/ITERATION
000416' 012737  000361  000116'         MOV     #361,WDFR       ;AT LEAST 361 WORDS FROM MEM/ITERATION
000424' 012737  000001  000120'         MOV     #1,INTR         ;AT LEAST 1 INTERRUPT /ITERATION
000432' 013700  000014'                 MOV     DVID1,R0        ;SAVE DEV COUNT
000436' 006200          2$:     ASR     R0              ;SHIFT IN A COUNT
000440' 001412                  BEQ     3$              ;BR OUT IF NONE LEFT
000442' 103375                  BCC     2$              ;BR BACK IF NO BIT IN THIS POSITION
000444' 062737  000361  000114'         ADD     #361,WDTO       ;361 MORE WORDS TO MEM
000452' 062737  000361  000116'         ADD     #361,WDFR       ;361 MORE WORDS FROM MEM
000460' 005237  000120'                 INC     INTR            ;1 MORE INTERRUPT
000464' 000764                  BR      2$              ;GO CHECK FOR MORE
000466' 013737  000014' 000240' 3$:     MOV     DVID1,DV        ;MOVE DVID1 TO TEMP STORAGE
000474' 043737  000246' 000240'         BIC     B12T15,DV       ;ENSURE BITS 12-15 ARE CLEARED
000502' 013737  000240' 000242'         MOV     DV,MASK         ;INIT MASK BITS EQUAL TO DVID1
000510' 013737  000240' 000244'         MOV     DV,MORE         ;THEN SET UP WORKING AREA
000516' 001002                  BNE     1$              ;GO TO 1$ IF THERE ARE DEVS
000520' 000137  001404'                 JMP     DROP            ;ELSE DROP THE MODULE
000524'                 1$:
000524' 004737  001646'         JSR     PC,CLRREG       ;CLEAR ALL DEV REGS
000530' 004737  001410'         JSR     PC,SETUP        ;SET UP AVAILABLE VEC LOCATIONS AND
                                                        ;GET BUFFER SIZES FOR DATO(S) & DATI(S)

                        ;***********************************************
                        ;***********************************************
                                ;NOTE: IN THE FOLLOWING COMMENTS THE TERM READ IS USED
                                ;TO DENOTE DATI OR DATIP AND THE TERM WRITE DENOTES
                                ;DATO OR DATOB.
                        ;***********************************************
                        ;***********************************************
                        ;THIS IS THE LOCATION THE PROGRAM WILL PROCEED
                        ;WITH AFTER AN END OF PASS

000534'                 RESTRT:
000534' 104415  000000' 000324'         GETPAS,REGIN, RBUFVA            ;GET PHYSICAL ADDRESS FROM 16-BIT RBUFVA
000542' 013737  000326' 000336'         MOV     RBUFPA,ENDRBF   ;MOVE READ BUFF ADDR TO END-OF-READ BUFF REG
000550' 062737  000736  000336'         ADD     #736,ENDRBF     ;ADD 736 TO GET LAST ADDR OF BUFFER

000556' 104415  000000' 000316'         GETPAS,REGIN, WBUFVA            ;GET PHYSICAL ADDRESS FROM 16-BIT WBUFVA
000564' 013737  000320' 000342'         MOV     WBUFPA,ENDWBF   ;MOVE WRITE BUFF ADDR TO END-OF-WRITE BUFF REG
000572' 062737  000736  000342'         ADD     #736,ENDWBF     ;ADD 736 TO GET LAST ADDR OF BUFFER

000600' 004737  002234'                 JSR     PC,FARITS       ;SET BITS 0 & 1 OF REA EQUAL TO
                                                        ;BITS 4 & 5 OF RBUFEA RESPECTIVELY
000604' 012737  000001  000266'         MOV     #1,DODI         ;INIT XFER INDICATOR TO 1 SO THE
                                                        ;1ST INC WILL SIGNIFY A DATI
000612' 005037  000234'                 CLR     ROTCNT          ;CLEAR PATTERN ROTATION COUNT
000616' 012737  000224' 000272'         MOV     #P1,SAVR1       ;INIT SAVR1 TO 1ST PATTERN
```

```
                                        MOV     #XFR1,SAVR5     ;INIT SAVR5 TO 1ST XFER FUNCTION
                                        MOV     #60,RQLVL       ;INIT REQUEST LEVEL TO NPR/INTR BR7

                                ;THIS IS THE LOCATION THE PROGRAM WILL PROCEED WITH
                                ;AFTER AN ITERATION

000640'                         RERREST:
000640' 023737  000242' 000240'         CMP     MASK,DV         ;# OF DEVS THAT SHOULD INTR=# OF DEVS THAT DID?
000646' 001402                          BEQ     1$              ;IF EQUAL,GO TO 2$
000650' 004737  001410'                 JSR     PC,SETUP        ;ELSE GET NEW DEV COUNTS & BUFFER SIZES
000654'                         1$:
000654' 013737  000266' 000270'         MOV     DODI,SVDODI     ;SAVE DODI SETTING
000662' 013737  000242' 000244'         MOV     MASK,MORE       ;RESET MORE BITS
000670' 013737  000242' 000240'         MOV     MASK,DV         ;IF ANY DEVS DROPPED, ALTER DV
000676' 001002                          BNE     2$              ;IF NOT, CONTINUE ON
000700' 000137  001404'                 JMP     DROP            ;ELSE DROP MODULE
000704'                         2$:
000704' 004737  002402'                 JSR     PC,RESDAT       ;RESTORE DATA IN READ BUFFER
                                                                ;AND CLEAR WRITE BUFFER
000710' 013700  000006'                 MOV     ADDR,R0         ;PUT DEVICE ADR INTO REG. 0
000714' 013737  000320' 000340'         MOV     WRUFPA,WBFADR   ;USE WBFADR AS WRITE BUFF ADDR
000722' 163737  000314' 000340'         SUB     WRUF,WBFADR     ;INIT TO WRITE BUFF LESS WBUF
000730' 013737  000326' 000334'         MOV     RBUFPA,RBFADR   ;USE RBFADR AS READ BUFF ADDR
000736' 163737  000312' 000334'         SUB     RBUF,RBFADR     ;INIT TO READ BUFF LESS RBUF
000744'                         3$:
000744' 006237  000244'                 ASR     MORE            ;CHECK FOR ANOTHER DEV
000750' 103404                          BCS     4$              ;IF THERE, CONTINUE
000752' 001440                          BEQ     GO              ;AND IF NOT, GO TO GO
000754' 062700  000020                  ADD     #20,R0          ;ELSE ADD 20 FOR NEXT ADDR
000760' 000771                          BR      3$              ;AND CHECK FOR MORE
000762'                         4$:
000762' 004737  001742'                 JSR     PC,CHEKPX       ;SEE IF R1 & R5 NEED INITIALIZATION
000766' 005237  000266'                 INC     DODI            ;INC XFER INDICATOR
000772' 032737  000001  000266'         BIT     #BIT0,DODI      ;IF BIT 0 OF DODI IS 1
001000' 001003                          BNE     5$              ;GO SET UP REGS FOR DATO(S)
001002' 004737  002070'                 JSR     PC,RREGS        ;ELSE SET UP REGS FOR DATI(S)
001006' 000402                          BR      6$              ;GO LOAD THE REGISTERS
001010'                         5$:
001010' 004737  002016'                 JSR     PC,WREGS        ;SET REGS FOR WRITE XFER INFO

                                ;AT THIS POINT THE DEVICE REGISTERS WILL BE LOADED TO DO
                                ;TRANSFERS

001014'                         6$:
001014' 012120                          MOV     (R1)+,(R0)+     ;SET UP DATA BUFFER REG
001016' 011220                          MOV     (R2),(R0)+      ;SET UP CYCLE COUNT REG.
001020' 011320                          MOV     (R3),(R0)+      ;SET UP ADDR REG
001022' 013760  000310' 000010          MOV     EABIT,10(R0)    ;SET UP EXT ADDR BITS IN CR2
001030' 012510                          MOV     (R5)+,(R0)      ;SET UP CR1
001032' 053710  000354'                 BIS     RQLVL,(R0)      ;SET REQUEST LEVELS
001036' 062700  000012                  ADD     #12,R0          ;ADD 12 TO GET NEXT DEV ADDR

001042' 010137  000272'                 MOV     R1,SAVR1        ;SAVE CONTENTS OF R1(LAST DATA PATTERN)
001046' 010537  000274'                 MOV     R5,SAVR5        ;SAVE CONTENTS OF R5(LAST XFER INSTR.)
001052' 000734                          BR      3$              ;GO LOAD ANOTHER DEVICE
```



```
001054'                         GO:
001054' 005037  000406'                 CLR     TOCNT           ;CLEAR TIME OUT COUNT
001060' 005037  000242'                 CLR     MASK            ;CLEAR ALL BITS IN MASK
                                                                ;TO BE RESET IN INTR SERV RTNS
001064' 005277  177146                  INC     @SIMLGO         ;SET OFF ALL DEVS
001070'                         BRK:
001070' 104407  000000'                 BREAK$,REGIN            ;TEMPORARY RETURN TO MONITOR....
001074' 104407  000000'                 BREAK$,REGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001100' 023737  000240' 000242'         CMP     DV,MASK         ;ALL DEVS INTR?
001106' 001411                          BEQ     DATACK          ;IF YES, GO TO DATACK
001110' 005337  000406'                 DEC     TOCNT           ;ELSE DEC TIME OUT COUNT
001114' 001365                          BNE     BRK             ;AND BREAK IF NOT 0
001116' 104403  000000' 003122'         MSGN$,BEGIN,NOINTR      ;ASCII MESSAGE CALL WITH COMMON HEADER
001124' 005004                          CLR     R4              ;R4 INVALID IN ERR REPORT
001126' 004737  002502'                 JSR     PC,GETDEV       ;REPORT NON-INTR'G DEVICES

001132'                         DATACK:
001132' 013737  000240' 000244'         MOV     DV,MORE         ;SET UP MORE
001140' 013737  000270' 000266'         MOV     SVDODI,DODI     ;RESTORE DODI TO SETTING WHEN
                                                                ;LOADING 1ST DEV
001146' 013700  000006'                 MOV     ADDR,R0         ;INIT R0 TO 1ST DEV  ADDR
001152' 012737  004324' 000334'         MOV     #RBEGIN,RBFADR  ;RBFADR=1ST LOC IN READ BUFFER
001160' 012737  003362' 000340'         MOV     #WRTOUT,WBFADR  ;WBFADR=1ST LOC IN WRITE BFFER
001166' 005003                          CLR     R3              ;USED IN DODATA RTN
001170'                         MORDEV:
001170' 006237  000244'                 ASR     MORE            ;CHECK FOR OTHER DEVS
001174' 103405                          BCS     2$              ;IF THERE CONTINUE
001176' 001001                          BNE     1$              ;IF SOME LEFT,GO CHECK FOR MORE
001200' 000447                          BR      NPASS           ;ELSE GO TO END OF PASS RTN
001202'                         1$:
001202' 062700  000020                  ADD     #20,R0          ;GET NEXT DEV ADDR
001206' 000770                          BR      MORDEV          ;GO CHECK FOR THAT DEV
001210'                         2$:
001210' 105760  000006                  TSTB    6(R0)           ;IS READY BIT SET IN CR1?
001214' 100407                          BMI     3$              ;IF SET, CONTINUE
001216' 104403  000000' 003102'         MSGN$,BEGIN,NOSET       ;ASCII MESSAGE CALL WITH COMMON HEADER
001224' 005004                          CLR     R4              ;R4 INVALID IN ERR REPORT
001226' 004737  002556'                 JSR     PC,RDDATA       ;ELSE TYPE OUT CONTENTS OF ALL DEV REGS
001232' 000426                          BR      NXT             ;AND GO CHECK FOR NEXT DEV
001234'                         3$:
001234' 033760  000356' 000016          BIT     ERBITS,16(R0)   ;SEE IF ANY ERROR BITS SET
001242' 001407                          BEQ     4$              ;IF NONE SET, CONTINUE
001244' 104403  000000' 003106'         MSGN$,BEGIN,ERRSET      ;ASCII MESSAGE CALL WITH COMMON HEADER
001252' 005004                          CLR     R4              ;R4 INVALID IN ERR REPORT
001254' 004737  002556'                 JSR     PC,RDDATA       ;ELSE TYPE OUT CONTENTS OF ALL DEV REGS
001260' 000413                          BR      NXT             ;AND GO CHECK FOR NEXT DEV
001262'                         4$:
001262' 005237  000266'                 INC     DODI            ;INC XFER INDICATOR
001266' 032737  000001  000266'         BIT     #BIT0,DODI      ;DATI OR DATO ?
001274' 001003                          BNE     5$              ;IF DATO,GO  DO IT
001276' 004737  002110'                 JSR     PC,DIDATA       ;CHECK DATI XFERS
001302' 000402                          BR      NXT             ;CHECK FOR NEXT DEV
001304'                         5$:
001304' 004737  002162'                 JSR     PC,DODATA       ;CHECK DATO XFERS
001310'                         NXT:
001310' 062700  000020                  ADD     #20,R0          ;ADD 20 FOR NEXT DEV ADDR
```

```
                                ;JMP     MORDEV          ;GO CHECK FOR ANOTHER DEV
                        NPASS:
001320  004737  001646  001170  ;JSR     PC,CLRREG       ;CLEAR ALL DEV REGS
001324  004737  002276  001170  ;JSR     PC,ROTRQS       ;SHIFT REQUEST TO NEXT BR LEVEL
001330  022737  000352  000274  ;CMP     #XFR4,SAVR5     ;HAS LAST XFER BEEN DONE?
001336  001010                  ;BNE     1$              ;IF NOT, CONTINUE
                                ;***************************************************
                                ;***************************************************
                                ;AFTER P2 & P4 HAVE BEEN ROTATED 16 TIMES, ALLOW THE POINTER
                                ;R1 TO INCREMENT THRU THE END OF THE READ BUFFER BUT DON'T ALLOW ROTATION
                                ;***************************************************
                                ;***************************************************

001340  022737  000020  000234  ;CMP     #16.,ROTCNT     ;ELSE HAS P2&P4 BEEN ROTATED 16 TIMES?
001346  001404                  ;BEQ     1$              ;IF YES DON'T ROTATE PATTERNS ANYMORE
001350  005237  000234          ;INC     ROTCNT          ;ELSE INC ROTATION COUNT, AND
001354  004737  002342          ;JSR     PC,ROTPAT       ;ROTATE DATA PATTERNS
001360                  1S:
001360  032737  000001  000264  ;BIT     #BIT0,DEVCNT    ;IS THERE AN ODD # OF DEVS?
001366  001402                  ;BEQ     2$              ;IF NOT,GO TO 5$
001370  004737  002704          ;JSR     PC,SWAPCC       ;ELSE SWAP RCC WITH WCC & RBUF
                                ;WITH WBUF FOR CORRECT XFER INFO
001374                  2S:
001374  104413  000000          ;ENDITS,BEGIN           ;SIGNAL END OF ITERATION.
                                ;MONITOR SHALL TEST END OF PASS
001400  000137  000640          ;JMP     REREST          ;ELSE GO TO REREST
                                ;***************************************************

001404                  DROP:
001404  104410  000000          ;ENDS,BEGIN             ;

                                ;***************************************************
                                ;***************************************************
                                ;THIS ROUTINE WILL SET UP THE DEVICE INTO VECTORS, COUNT THE
                                ;NUMBER OF DEVICES, SET THE TRANSFER SIZE OF THE READ AND WRITE
                                ;BUFFERS, AND SET UP THE READ AND WRITE CYCLE COUNTS.
                                ;***************************************************
                                ;***************************************************

001410                  SETUP:
001410  005037  000306  ;CLR     DICNT           ;CLEAR REG COUNTING DEVS DOING DATI(S)
001414  005037  000304  ;CLR     DOCNT           ;CLEAR REG COUNTING DEVS DOING DATO(S)
001420  012700  002742  ;MOV     #ISR1,R0        ;SET R0 TO 1ST INTR SERVICE RTN
001424  013701  000010  ;MOV     @VCTOR,R1       ;SET R1 TO 1ST INTR VEC ADDR
001430  012737  000001  000266  ;MOV     #1,DODI         ;INIT XFR INDICATOR TO 1
001436                  CKMORE:
001436  006237  000244  ;ASR     MORE            ;CHECK FOR A DEVICE
001442  103406          ;BCS     1$              ;IF THERE, CONTINUE
001444  001420          ;BEQ     2$              ;IF NONE LEFT AT ALL, GET OUT
001446  062700  000010  ;ADD     #10,R0          ;ELSE INC INTR SERV POINTER BY 10
001452  062701  000004  ;ADD     #4,R1           ;AND INC INTR VEC LOC BY 4
001456  000767          ;BR      CKMORE          ;GO BACK AND CHECK FOR ANOTHER DEV
001460                  1S:
001460  005237  000266  ;INC     DODI            ;INC XFER INDICATOR
001464  004737  001622  ;JSR     PC,INCCNT       ;INC DATI OR DATO COUNT
001470  010021          ;MOV     R0,(R1)+        ;MOVE ADDR OF INTR SERV RTN TO VEC LOC
```



```
001472  113721  000012  ;MOVB    @R1,(R1)+       ;MOVE BR1 VALUE TO PSW VEC
001476  105721          ;TSTB    (R1)+           ;GET BACK TO AN EVEN ADDR
001500  062700  000010  ;ADD     #10,R0          ;ADD 10 TO R0 FOR NEXT INTR SERV RTN
001504  000754          ;BR      CKMORE          ;SEE IF THERE ARE MORE DEVS
001506                  2S:
001506  013737  000266  000264  ;MOV     DODI,DEVCNT     ;DEV XFER INDICATOR HAS # OF DEVS
001514  005337  000264  ;DEC     DEVCNT          ;- 1, SO DEC DEVCNT BY 1
001520                  GETBUF:
001520  012700  000246  ;MOV     #BUFSZ-2,R0     ;SET R0 TO SIZE TABLE
001524  005001          ;CLR     R1              ;CLEAR R1
001526                  1S:
001526  005720          ;TST     (R0)+           ;MOVE POINTER TO NEXT VALUE
001530  005201          ;INC     R1              ;INC COUNTER
001532  020137  000306  ;CMP     R1,DICNT        ;R1=# OF DATI DEVS ?
001536  103773          ;BLO     1$              ;IF NOT,INC COUNTER & POINTER
001540  011037  000312  ;MOV     (R0),RBUF       ;ELSE R0 HAS SIZE OF READ BUFF PER DEV
                        ;1 WORD = 2 LOCATIONS(OR BYTES)
001544  011037  000302  ;MOV     (R0),RCC        ;AND ALSO IS DATI XFER COUNT
001550  006237  000302  ;ASR     RCC             ;WHEN IT IS HALVED
001554  005437  000302  ;NEG     RCC             ;GET 2'S COMP FOR DEV CC REG
001560  005737  000304  ;TST     DOCNT           ;IF DOCNT IS 0
001564  001405          ;BEQ     2S              ;GO USE THE VALUE R0 POINTS TO

001566  020137  000304  ;CMP     R1,DOCNT        ;R1=# OF DATO DEVS ?
001572  001402          ;BEQ     2S              ;YES, CONTINUE
001574  005301          ;DEC     R1              ;NO, DEC R1
001576  005740          ;TST     -(R0)           ;AND DEC POINTER
001600                  2S:
001600  011037  000314  ;MOV     (R0),WBUF       ;R0 HAS SIZE OF WRITE BUFF PER DEV
                        ;1 WORD = 2 LOCATIONS(OR BYTES)
001604  011037  000300  ;MOV     (R0),WCC        ;AND ALSO IS DATO XFER COUNT
001610  006237  000300  ;ASR     WCC             ;WHEN IT IS HALVED
001614  005437  000300  ;NEG     WCC             ;GET 2'S COMP FOR DEV CC REG
001620  000207          ;RTS     PC              ;RETURN
                                ;***************************************************

                                ;***************************************************
                                ;***************************************************
                                ;THIS ROUTINE WILL COUNT THE NUMBER OF DEVICES DOING DATI(S)
                                ;AND THOSE DOING DATO(S)
                                ;***************************************************
                                ;***************************************************

001622                  INCCNT:
001622  032737  000001  000266  ;BIT     #BIT0,DODI      ;IF BIT 0 IS 1
001630  001003          ;BNE     INCDO           ;GO INC DOCNT
001632  005237  000306  ;INC     DICNT           ;ELSE INC DICNT
001636  000402          ;BR      EXINC           ;AND EXIT RTN
001640                  INCDO:
001640  005237  000304  ;INC     DOCNT           ;INC DOCNT
001644                  EXINC:
001644  000207          ;RTS     PC              ;RETURN
                                ;***************************************************

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

```
                                ;THIS ROUTINE WILL CLEAR ALL THE REGISTERS OF ALL EXISTING DEVICES
                                ;AS DEFINED BY DVID1
                                ;*************************************************
                                ;*************************************************

001646'                         CLRREG:
001646' 013737  000240' 000244'         MOV     DV,MORE         ;RESTORE MORE
001654' 013701  000006'                 MOV     ADDR,R1         ;INIT R1 TO 1ST DEV ADDR
001660'                         1$:
001660' 006237  000244'                 ASR     MORE            ;IS THERE A DEVICE?
001664' 103404                          BCS     2$              ;IF YES,GO SET IT UP
001666' 001421                          BEQ     3$              ;IF NONE LEFT, - GET OUT
001670' 062701  000020                  ADD     #20,R1          ;ELSE INC R1 TO NEXT DEV ADDR
001674' 000771                          BR      1$              ;AND GO SEE IF IT'S THERE
001676'                         2$:
001676' 005011                          CLR     (R1)            ;CLEAR DATA BUFFER REG
001700' 005061  000002                  CLR     2(R1)           ;CLEAR CYCLE COUNT REG
001704' 005061  000004                  CLR     4(R1)           ;CLEAR BUFFER ADDR REG
001710' 005061  000006                  CLR     6(R1)           ;CLEAR CR1 REG
001714' 005061  000010                  CLR     10(R1)          ;CLEAR ERROR CLEAR REG
001720' 005061  000016                  CLR     16(R1)          ;CLEAR CR2 REG
001724' 062701  000020                  ADD     #20,R1          ;INC R1 TO NEXT DEV ADDR
001730' 000753                          BR      1$              ;GO CHECK FOR ANOTHER DEV
001732'                         3$:
001732' 013737  000240' 000244'         MOV     DV,MORE         ;RESTORE MORE
001740' 000207                          RTS     PC              ;RETURN
                                ;*************************************************


                                ;*************************************************
                                ;*************************************************
                                ;THIS ROUTINE WILL KEEP TRACK OF THE DATA PATTERNS AND TRANSFER
                                ;FUNCTIONS USED TO LOAD THE DEVICES
                                ;*************************************************
                                ;*************************************************

001742'                         CHKPX:
001742' 013701  000272'                 MOV     SAVR1,R1        ;RESTORE R1 TO PATTERN IN SAVR1
001746' 013705  000274'                 MOV     SAVR5,R5        ;RESTORE R5 TO XFR FUNCTION IN SAVR5
001752' 022705  000352'                 CMP     #XFR4,R5        ;DOES R5 POINT TO LAST XFER?
001756' 100002                          BPL     1$              ;IF NOT, CONTINUE
001760' 012705  000344'                 MOV     #XFR1,R5        ;ELSE INIT TO 1ST XFER
001764'                         1$:
001764' 022701  000232'                 CMP     #P4,R1          ;IF R1 DOES NOT EXCEED LAST PATTERN
001770' 100011                          BPL     3$              ;ADDRESS, EXIT ROUTINE
001772' 022737  000020  000234'         CMP     #16.,ROTCNT     ;HAVE P2&P4 BEEN ROTATED 16 TIMES?
002000' 001003                          BNE     2$              ;IF NOT,INIT R1
002002' 023701  000336'                 CMP     ENDRBF,R1       ;IS R1 AT LAST ADDR OF READ BUFF?
002006' 100002                          BPL     3$              ;IF LESS,GO TO EXINIT
002010'                         2$:
002010' 012701  000224'                 MOV     #P1,R1          ;ELSE INIT R1 TO 1ST DATA PATTERN
002014'                         3$:
002014' 000207                          RTS     PC              ;RETURN
                                ;*************************************************


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



```
                                ;*************************************************
                                ;THIS ROUTINE WILL DETERMINE IF THE TRANSFER IS A DATO OR A DATOB
                                ;AND SET THE CYCLE COUNT APPROXIMATELY VIA R2 AND ALSO SET R3 TO
                                ;THE CORRECT WRITE BUFFER ADDRES
                                ;*************************************************
                                ;*************************************************

002016'                         WREGS:
002016' 012702  000300'                 MOV     #WCC,R2         ;SET R2 FOR A DATO CYCLE COUNT
002022' 032715  000400                  BIT     #BIT8,(R5)      ;IF BIT 8 OF XFER FUNC IS 0
002026' 001412                          BEQ     1$              ;ALL SET, FINISH UP
002030' 032715  001000                  BIT     #BIT9,(R5)      ;IF BIT 9 OF XFER FUNCTION
002034' 001407                          BEQ     1$              ;ALL SET, FINISH UP
002036' 012702  000276'                 MOV     #BYTCC,R2       ;ELSE SET R2=ADDR OF BYTCC
002042' 013737  000300' 000276'         MOV     WCC,BYTCC       ;MOVE VALUE OF WCC TO BYTCC AND
002050' 006337  000276'                 ASL     BYTCC           ;DOUBLE BY SHIFTING LEFT
002054'                         1$:                             ;ELSE LEAVE R2 = WCC
002054' 063737  000314' 000340'         ADD     WBUF,WBFADR     ;INC WBFADR ADDR BY VALUE IN WBUF
002062' 012703  000340'                 MOV     #WBFADR,R3      ;SET R3 = TO ADDR OF WBFADR
002066' 000207                          RTS     PC              ;RETURN
                                ;*************************************************


                                ;*************************************************
                                ;*************************************************
                                ;THIS ROUTINE WILL SET UP R3 WITH VALUES FOR DOING READS
                                ;*************************************************
                                ;*************************************************

002070'                         RREGS:
002070' 012702  000302'                 MOV     #RCC,R2         ;SET R2 FOR A DATI CYCLE COUNT
002074' 063737  000312' 000334'         ADD     RBUF,RBFADR     ;INC RBFADR ADDR BY VALUE IN RBUF
002102' 012703  000334'                 MOV     #RBFADR,R3      ;SET R3 = TO VALUE IN RBFADR
002106' 000207                          RTS     PC              ;RETURN
                                ;*************************************************


                                ;*************************************************
                                ;*************************************************
                                ;THIS ROUTINE CHECKS FOR A CORRECT DATI TRANSFER SINCE ALL OF THE
                                ;READ TRANSFER BUFFER SIZE CONTAINS THE SAME DATA, ONLY ONE LOCATION
                                ;IS COMPARED WITH THE CONTENTS IN THE DEVICE DATA BUFFER REGISTER.
                                ;*************************************************
                                ;*************************************************

002110'                         DIDATA:
002110' 013703  000334'                 MOV     RBFADR,R3       ;SET R3 = 1ST READ BUFF ADDR
002114' 063703  000312'                 ADD     RBUF,R3         ;GET LAST ADDR BY ADDING BUFF SIZE
002120' 013704  000334'                 MOV     RBFADR,R4       ;SET R4=PRESENT READ BUFF ADDR
002124'                         1$:
002124' 021024                          CMP     (R0),(R4)+      ;IS VALUE IN DEV DATA REG=TO THAT IN READ BUFF?
002126' 001407                          BEQ     2$              ;IF EQUAL, CONTINUE
002130' 104403  000000' 003112'         MSGN$,BFGIN,DIERR       ;ASCII MESSAGE CALL WITH COMMON HEADER
002136' 005744                          TST     -(R4)           ;RETURN R4 TO ERR LOCATION
002140' 004737  002556'                 JSR     PC,BDDATA       ;AND TYPE OUT CONTENTS OF REGS
002144' 000402                          BR      3$              ;GO EXIT
002146'                         2$:
002146' 020403                          CMP     R4,R3           ;IS R4 = LAST XFER LOCATION?
```

```
002150' 103765                  BLO     1$              ;IF NOT,CHECK NEXT ADDR
002152'                 3$:
002152' 063737  000312' 000334' ADD     RBUF,RBFADR     ;INC READ BUFF ADDR BY RBUF VALUE
002160' 000207                  RTS     PC              ;RETURN
                        ;*************************************************************
                        ;*************************************************
                        ;*************************************************
                        ;THIS ROUTINE CHECKS FOR CORRECT DATO TRANSFERS ALL VALUES IN THE
                        ;WRITE TRANSFER BUFFER IS COMPARED WITH THE CONTENTS IN THE DEVICE
                        ;DATA BUFFER REGISTER
                        ;*************************************************
                        ;*************************************************
002162'                 DODATA:
002162' 013703  000340' MOV     WBFADR,R3       ;SET R3 = PRESENT WRITE BUFF ADDR
002166' 063703  000314' ADD     WBUF,R3         ;INC R3 BY # OF XFER LOCATIONS
002172' 013704  000340' MOV     WBFADR,R4       ;SET R4 = PRESENT WRITE BUFF ADDR
002176'                 1$:
002176' 021024          CMP     (R0),(R4)+      ;IS DEV DATA REG = VALUE IN BUFF ADDR?
002200' 001407          BEQ     2$              ;IF EQUAL, CONTINUE
002202' 104403  000000' 003116' MSGNS,BEGIN,DOERR ;ASCII MESSAGE CALL WITH COMMON HEADER
002210' 005744          TST     -(R4)           ;RETURN R4 TO ERR LOCATION
002212' 004737  002556' JSR     PC,RDDATA
002216' 000402          BR      3$
002220'                 2$:
002220' 020403          CMP     R4,R3           ;IS R4 AT LAST XFER LOCATION?
002222' 103765          BLO     1$              ;IF NOT,GO CHECK NEXT ADDR
002224'                 3$:
002224' 063737  000314' 000340' ADD     WBUF,WBFADR     ;INC WRITE BUFF BY VALUE IN WBUF
002232' 000207          RTS     PC              ;RETURN
                        ;*************************************************************
                        ;*************************************************
                        ;*************************************************
                        ;THIS ROUTINE WILL SET THE CORRECT EXTENDED MEMORY ADDRESS BITS
                        ;OF THE CR2 REGISTER IN ACCORDANCE WITH THE EXTENDED MEMORY ADDRESS
                        ;BITS AS SET BY THE MONITOR IN RBUFEA.
                        ;*************************************************
                        ;*************************************************
002234'                 EABITS:
002234' 005037  000310' CLR     EABIT           ;CLEAR EXT MEM BITS
002240' 032737  000020  000330' BIT     #BIT4,RBUFEA    ;IS BIT 4 OF RBUFEA SET?
002246' 001403          BEQ     1$              ;IF NOT, CONTINUE
002250' 052737  000001  000310' BIS     #BIT0,EABIT     ;ELSE SET BIT 0 OF EABIT
002256'                 1$:
002256' 032737  000040  000330' BIT     #BIT5,RBUFEA    ;IS BIT 5 OF RBUFEA SET?
002264' 001403          BEQ     2$              ;IF NOT, RETURN
002266' 052737  000002  000310' BIS     #BIT1,EABIT     ;ELSE SET BIT 1 OF EABIT
002274'                 2$:
002274' 000207          RTS     PC              ;RETURN
                        ;*************************************************************
```



```
                        ;THIS ROUTINE WILL CAUSE THE REQUEST LEVEL OF ALL DEVICES TO
                        ;BE AT A LOWER LEVEL THAN IN THE PREVIOUS ITERATION---ONLY
                        ;ONE LEVEL IS SET AT ANY ONE TIME EXCEPT FOR AN NPR ,THEN AN INTERRUPT
                        ;LEVEL MUST BE SET FOR HARDWARE CONSIDERATIONS
002276'                 ROTRQS:
002276' 022737  000060  000354' CMP     #60,RQLVL       ;IS REQUEST AT NPR WITH BR7 INTR?
002304' 001004          BNE     1$              ;IF NOT, CONTINUE
002306' 012737  000020  000354' MOV     #20,RQLVL       ;ELSE SET RFQUEST LEVEL TO BR7
002314' 000411          BR      2$              ;AND RETURN
002316'                 1$:
002316' 006237  000354' ASR     RQLVL           ;SHIFT REQUEST LEVEL TO NEXT BR
002322' 022737  000001  000354' CMP     #1,RQLVL        ;IF RFQUEST LEVEL DOES NOT = 1
002330' 001003          BNE     2$              ;THEN RETURN
002332' 012737  000060  000354' MOV     #60,RQLVL       ;ELSE RESET REQUEST TO NPR/BR7 INTR
002340'                 2$:
002340' 000207          RTS     PC              ;RETURN
                        ;*************************************************************
                        ;*************************************************
                        ;*************************************************
                        ;THIS ROUTINE WILL ROTATE THE DATA PATTERNS TO TRY TO EXERCISE
                        ;THE DATA LINES.
                        ;*************************************************
                        ;*************************************************
002342'                 ROTPAT:
002342' 012701  000226' MOV     #P2,R1          ;INIT R1 TO ADDR OF P2
002346'                 1$:
002346' 032711  000001  BIT     #BIT0,(R1)      ;IF BIT 0 OF PATTERN IS 0
002352' 001402          BFQ     2$              ;GO CLEAR C-BIT
002354' 000261          SEC                     ;ELSE SET C-BIT
002356' 000401          BR      3$              ;AND CONTINUE
002360'                 2$:
002360' 000241          CLC                     ;CLEAR C-BIT
002362'                 3$:
002362' 006011          ROR     (R1)            ;ROTATE DATA TO THE RIGHT
002364' 022701  000232' CMP     #P4,R1          ;DOES R1 POINT TO LAST PATTERN?
002370' 001403          BEQ     4$              ;IF IT DOES,EXIT
002372' 012701  000232' MOV     #P4,R1          ;ELSE SET R1=LAST PATTERN ADDR
002376' 000763          BR      1$              ;AND GO BACK FOR MORE
002400'                 4$:
002400' 000207          RTS     PC              ;RETURN
                        ;*************************************************************
                        ;THIS ROUTINE WILL SET UP THE READ BUFFER WITH ONE OR BOTH DATA
                        ;PATTERNS DEPENDING ON THE NUMBER OF DEVICES AND CLEAR THE WRITE
                        ;BUFFER SO THE FOLLOWING TRANSFERS CAN BE COMPARED CORRECTLY.
                        ;*************************************************
                        ;*************************************************
002402'                 RESDAT:
002402' 012702  004324' MOV     #REEDIN,R2      ;SET R2=1ST ADDR OF READ BUFF
002406' 013703  000226' MOV     P2,R3           ;SET R3= VALUE IN P2
002412' 012704  004324' MOV     #REEDIN,R4      ;SET R4= ADDR OF READ BUFFER
```

```
002416' 063702  000312'                 ADD     RBUF,R2         ;INC R2 TO LAST XFER ADDR
002422'                         1S:
002422' 010324                          MOV     R3,(R4)+        ;MOVE PATTERN INTO READ BUFF ADDR
002424' 022704  005262'                 CMP     #READTN+736,R4  ;IS R4 LAST ADDR IN READ BUFF?
002430' 103415                          BLO     3$              ;IF IT IS,GO TAKE CARE OF IT
002432' 020204                          CMP     R2,R4           ;IS R4 AT LAST XFER ADDR?
002434' 101372                          BHI     1$              ;IF NOT,GO TO NEXT ADDR
002436' 063702  000312'                 ADD     RBUF,R2         ;INC R2 TO LAST ADDR OF NEXT XFER
002442' 023703  000232'                 CMP     P4,R3           ;IS R3 = VALUE IN P4?
002446' 001403                          BEQ     2$              ;IF IT IS,GO TAKE CARE OF IT
002450' 013703  000232'                 MOV     P4,R3           ;ELSE SET R3 = P4
002454' 000762                          BR      1$              ;AND GO AGAIN
002456'                         2S:
002456' 013703  000226'                 MOV     P2,R3           ;RESET R3 = P2
002462' 000757                          BR      1$              ;AND GO AGAIN
002464'                         3S:
002464' 012704  003362'                 MOV     #WRTOUT,R4      ;SET R4 TO WRITE BUFFER
002470'                         4S:
002470' 005024                          CLR     (R4)+           ;CLEAR BUFFER
002472' 022704  004320'                 CMP     #WRTOUT+736,R4  ;END OF BUFFER?
002476' 101374                          BHI     4$              ;IF NOT,CLEAR NEXT WORD OF BUFF
002500' 000207                          RTS     PC              ;RETURN
                                ;*****************************************************

                                ;*************************************************
                                ;*************************************************
                                ;THIS ROUTINE WILL COMPARE BITS IN DV WITH THOSE IN MASK TO
                                ;DETERMINE WHICH DEVICES DID NOT INTERRUPT, THEN PRINT THEM.
                                ;*************************************************
                                ;*************************************************
002502'                         GETDEV:

002502' 012701  000001                  MOV     #1,R1           ;SET BIT 0 IN R1
002506' 013700  000006'                 MOV     @ADDR,R0        ;INIT R0 TO 1ST DEV ADDR
002512' 005004                          CLR     R4              ;CLEAR R4 AS NOT USED IN RDDATA RTN
002514'                         1S:
002514' 030137  000240'                 BIT     R1,DV           ;BIT TEST DV TO SEE IF DEV THERE
002520' 001007                          BNE     3$              ;IF THERE, CHECK MASK REG.
002522'                         2S:
002522' 006301                          ASL     R1              ;ELSE SHIFT TO NEXT BIT
002524' 032701  010000                  BIT     #10000,R1       ;AT BIT 12?
002530' 001011                          BNE     GETOUT          ;IF YES, EXIT
002532' 062700  000020                  ADD     #20,R0          ;ELSE GET NEXT DEV ADDR
002536' 000766                          BR      1$              ;AND SEE IF ITS THERE
002540'                         3S:
002540' 030137  000242'                 BIT     R1,MASK         ;DID DEV INTR?
002544' 001366                          BNE     2$              ;IF YES, GO CHECK NEXT ONE
002546' 004737  002556'                 JSR     PC,RDDATA       ;ELSE REPORT ALL ITS REGS.
002552' 000763                          BR      2$              ;THEN GO CHECK NEXT BIT
002554'                         GETOUT:
002554' 000207                          RTS     PC              ;EXIT
                                ;*****************************************************

                                ;*************************************************
                                ;*************************************************
                                ;THIS ROUTINE WILL LIST ALL THE CONTENTS OF THE DEVICE REGISTERS
```



```
                                ;ON REGISTERING AN INCORRECT DATA COMPARISON OR FOR AN INTERRUPT
                                ;OR ERROR
                                ;*************************************************
                                ;*************************************************
002556'                         RDDATA:
002556' 010437  000374'                 MOV     R4,RADR         ;SET UP BAD MEM FOR ERR TYPEOUT
002562' 010037  000360'                 MOV     R0,DB           ;DB HAS DEV DATA REG ADDR
002566' 013737  000360'  000362'        MOV     DB,CC           ;MOVE DB TO CC AND ADD 2
002574' 062737  000002   000362'        ADD     #2,CC           ;TO GET DEV CYCLE COUNT REG ADDR
002602' 013737  000362'  000364'        MOV     CC,BA           ;MOVE CC TO BA AND ADD 2
002610' 062737  000002   000364'        ADD     #2,BA           ;TO GET DEV BUFF ADDR REG ADDR
002616' 013737  000364'  000366'        MOV     BA,CR1          ;MOVE BA TO CR1 AND ADD 2
002624' 062737  000002   000366'        ADD     #2,CR1          ;TO GET DEV CR1 REG ADDR
002632' 013737  000366'  000370'        MOV     CR1,CR2         ;MOVE CR1 TO CR2 AND ADD 10
002640' 062737  000010   000370'        ADD     #10,CR2         ;TO GET DEV CR2 REG ADDR
002646' 013737  000366'  000100'        MOV     CR1,CSRA        ;GET RECR1 ADDR
002654' 017737  175506   000102'        MOV     @CR1,ACSR       ;GET CONTENTS OF RECR1

002662' 012737  000001   000106'        MOV     #1,ERRTYP       ;DATA ERROR
                                        ;*****************************************************************
002670' 104405  000000'  000360'        ERRDRS,REGIN,DVREGS     ;
                                        ;*****************************************************************

002676' 005060  000010                  CLR     10(R0)          ;CLEAR ALL ERRORS
002702' 000207                          RTS     PC              ;RETURN
                                ;*****************************************************

                                ;*************************************************
                                ;*************************************************
                                ;THIS ROUTINE WILL SWAP READ CYCLE COUNT AND BUFFER SIZE WITH THE
                                ;WRITE CYCLE COUNT AND BUFFER SIZE ON AN ODD NUMBER OF DEVICES
                                ;BECAUSE ON THE FIRST ITERATION, ASSUMING A DEVICE COUNT OF 5,
                                ;3 OF THOSE DEVICES WILL BE DOING DATI(S) AND THE BUFFER SIZE
                                ;WILL BE SMALLER THAN THE 2 DEVICES DOING DATO(S), THEN ON THE
                                ;NEXT ITERATION THERE WILL BE 2 DEVICES DOING DATI(S) AND
                                ;3 DOING DATO(S) SO THE CYCLE COUNT AND BUFFER SIZES MUST REFLECT
                                ;THIS CHANGE.
                                ;*************************************************
                                ;*************************************************
002704'                         SWAPCC:
002704' 013702  000302'                 MOV     RCC,R2          ;SAVE RCC
002710' 013737  000300'  000302'        MOV     WCC,RCC         ;GET NEW RCC FROM WCC
002716' 010237  000300'                 MOV     R2,WCC          ;GET NEW WCC FROM OLD RCC
002722' 013702  000312'                 MOV     RBUF,R2         ;SAVE RBUF
002726' 013737  000314'  000312'        MOV     WBUF,RBUF       ;GET NEW RBUF FROM WBUF
002734' 010237  000314'                 MOV     R2,WBUF         ;GET NEW WBUF FROM OLD RBUF
002740' 000207                          RTS     PC              ;RETURN


002742'                         ISR1:
002742' 052737  000001   000242'        BIS     #BIT0,MASK      ;SET BIT0 TO INDICATE DEV 1 INTERR'D
002750' 000002                          RTI                     ;RETURN
002752'                         ISR2:
002752' 052737  000002   000242'        BIS     #BIT1,MASK      ;SET BIT1 TO INDICATE DEV 2 INTERR'D
002760' 000002                          RTI                     ;RETURN
002762'                         ISR3:
```

```
  937  002762' 052737  000004  000242'          BIS     #BIT2,@MASK       ;SET BIT2 TO INDICATE DEV 3 INTERR'D
  938  002770' 000002                           RTI                       ;RETURN
  939  002772'                          ISR4:
  940  002772' 052737  000010  000242'          BIS     #BIT3,@MASK       ;SET BIT3 TO INDICATE DEV 4 INTERR'D
  941  003000' 000002                           RTI                       ;RETURN
  942  003002'                          ISR5:
  943  003002' 052737  000020  000242'          BIS     #BIT4,@MASK       ;SET BIT4 TO INDICATE DEV 5 INTERR'D
  944  003010' 000002                           RTI                       ;RETURN
  945  003012'                          ISR6:
  946  003012' 052737  000040  000242'          BIS     #BIT5,@MASK       ;SET BIT5 TO INDICATE DEV 6 INTERR'D
  947  003020' 000002                           RTI                       ;RETURN
  948  003022'                          ISR7:
  949  003022' 052737  000100  000242'          BIS     #BIT6,@MASK       ;SET BIT6 TO INDICATE DEV 7 INTERR'D
  950  003030' 000002                           RTI                       ;RETURN
  951  003032'                          ISR8:
  952  003032' 052737  000200  000242'          BIS     #BIT7,@MASK       ;SET BIT7 TO INDICATE DEV 8 INTERR'D
  953  003040' 000002                           RTI                       ;RETURN
  954  003042'                          ISR9:
  955  003042' 052737  000400  000242'          BIS     #BIT8,@MASK       ;SET BIT8 TO INDICATE DEV 9 INTERR'D
  956  003050' 000002                           RTI                       ;RETURN
  957  003052'                          ISR10:
  958  003052' 052737  001000  000242'          BIS     #BIT9,@MASK       ;SET BIT9 TO INDICATE DEV 10 INTERR'D
  959  003060' 000002                           RTI                       ;RETURN
  960  003062'                          ISR11:
  961  003062' 052737  002000  000242'          BIS     #BIT10,@MASK      ;SET BIT10 TO INDICATE DEV 11 INTERR'D
  962  003070' 000002                           RTI                       ;RETURN
  963  003072'                          ISR12:
  964  003072' 052737  004000  000242'          BIS     #BIT11,@MASK      ;SET BIT11 TO INDICATE DEV 12 INTERR'D
  965  003100' 000002                           RTI                       ;RETURN
  966
  967  003102' 003200'                  NOSET:  MSSG3
  968  003104' 177777                           -1
  969  003106' 003232'                  ERRSET: MSSG4
  970  003110' 177777                           -1
  971  003112' 003126'                  DIERR:  MSSG1
  972  003114' 177777                           -1
  973  003116' 003153'                  DOERR:  MSSG2
  974  003120' 177777                           -1
  975  003122' 003304'                  NOINTR: MSSG5
  976  003124' 177777                           -1
  977  003126' 042045  052101  020111   MSSG1:  .ASCIZ  "%DATI OR DATIP ERROR"
  978  003134' 051117  042040  052101
  979  003142' 050111  042440  051122
  980  003150' 051117     000
  981  003153'    045  040504  047524   MSSG2:  .ASCIZ  "%DATO OR DATOB ERROR"
  982  003160' 047440  020122  040504
  983  003166' 047524  020102  051105
  984  003174' 047522  000122
  985  003200' 042045  053105  041511   MSSG3:  .ASCIZ  "%DEVICE READY BIT NOT SET"
  986  003206' 020105  042522  042101
  987  003214' 020131  044502  020124
  988  003222' 047516  020124  042523
  989  003230' 000124
  990  003232' 042045  053105  041511   MSSG4:  .ASCIZ  "%DEVICE INTERRUPTED ON ERROR--NOT ON DONE"
  991  003240' 020105  047111  042524
  992  003246' 051122  050125  042524
```



```
  993  003254' 020104  047117  042440
  994  003262' 051122  051117  026455
  995  003270' 047516  020124  047117
  996  003276' 042040  047117  000105
  997  003304' 052045  042510  043040   MSSG5:  .ASCIZ  "%THE FOLLOWING DEVICE(S) DID NOT INTERRUPT:%"
  998  003312' 046117  047514  044527
  999  003320' 043516  042040  053105
 1000  003326' 041511  024105  024523
 1001  003334' 042040  042111  047040
 1002  003342' 052117  044440  052116
 1003  003350' 051105  052522  052120
 1004  003356' 022472     000
 1005          003362'                   .EVEN
 1006  003362' 000361                   WRTOUT: .BLKW   361
 1007  004324' 000361                   RREDIN: .BLKW   361
 1008
 1009          005266'                   .=.
 1010
 1011          000001                   .END
```

```
ACSR    000102R          246#    897*
ADDR    000006R          212#    396     453     608     858
ADDR22= 001000           264#
ASB     000106R          250#
ASTAT   000104R          248#
AWAS    000110R          251#
BA      000364R          317#    890*    891*    892
BADMEM  000372R          320#
BADR    000374R          320     321#    886*
BDDATA  002556R          470     477     709     738     872     885#
BEGIN   000000R          209#    362     366     439     440     445     468     475     512     518     707     736     901
BIT0  = 000001           264#    411     481     507     589     760     798     931
BIT1  = 000002           264#    764     934
BIT10 = 002000           264#    961
BIT11 = 004000           264#    964
BIT12 = 010000           264#
BIT13 = 020000           264#
BIT14 = 040000           264#
BIT15 = 100000           264#
BIT2  = 000004           264#    937
BIT3  = 000010           264#    940
BIT4  = 000020           264#    758     943
BIT5  = 000040           264#    762     946
BIT6  = 000100           264#    949
BIT7  = 000200           264#    952
BIT8  = 000400           264#    666     955
BIT9  = 001000           264#    668     958
BREAK$= 104407           264#    439     440
BRK     001070R          438#    444
BR1     000012R          214#    545
BR2     000013R          215#
BTOD$ = 104421           264#
BUFSZ   000250R          276#    553
BYTCC   000276R          289#    670     671*    672*
BIT15   000246R          277#    341
CC      000362R          316#    888*    889*    890
CDATA$= 104412           264#
CHEKPX  001742R          409     637#
CKMORE  001436R          534#    540     548
CLRREG  001646R          347     491     606#
CONFIG  000056R          234#
CR1     000366R          318#    892*    893*    894     896     897
CR2     000370R          319#    894*    895*
CSRA    000100R          244#    896*
DATACK  001132R          442     449#
DATCK$= 104411           264#
DATER$= 104404           264#
DB      000360R          315#    887*    888
DEVCNT  000264R          294#    507     550*    551*
DTCNT   000306R          293#    529*    558     591*
DIDATA  002110R          483     700#
DIERR   003112R          707     971#
DOCNT   000304R          292#    530*    565     568     594*
DODATA  002162R          486     729#
DODI    000266R          295#    373*    388     410*    411     451*    480*    481     533*    542*    550     589
DOERR   003116R          736     973#
```



```
DROP    001404R          345     392     517#
DV      000240R          274#    324     340*    341*    342     343     384     390*    441     450     607     625     861
DVID1   000014R          216#    332     340
DVMASK  000400R          323#
DVREGS  000360R          314#    901
EABIT   000310R          294#    425     757*    760*    764*
EABITS  002234R          371     756#
ENDIT$= 104413           264#    512
ENDRBP  000336R          305#    363*    364*    648
ENDWBP  000342R          307#    367*    368*
END$  = 104410           264#    518
ERBITS  000356R          313#    473
ERRSET  003106R          475     969#
ERRTYP  000106R          249#    899*
EXINC   001644R          592     595#
EXIT$ = 104400           264#
GETBUF  001520R          552#
GETDEV  002502R          447     855#
GETDUT  002554R          866     874#
GETPAS= 104415           264#    362     366
GO      001054R          405     433#
GWBUF$= 104414           264#
HRDCNT  000044R          229#
HRDER$= 104405           264#    901
HRDPAS  000050R          231#
ICONT   000036R          226#
ICOUNT  000040R          227#
IDNUM   000122R          256#
INCCNT  001622R          543     588#
INCDO   001640R          590     593#
INIT    000030R          223#
INTR    000120R          255#    331*    338*
ISR1    002742R          531     930#
ISR10   003052R          957#
ISR11   003062R          960#
ISR12   003072R          963#
ISR2    002752R          933#
ISR3    002762R          936#
ISR4    002772R          939#
ISR5    003002R          942#
ISR6    003012R          945#
ISR7    003022R          948#
ISR8    003032R          951#
ISR9    003042R          954#
MAP22$= 104416           264#
MASK    000242R          275#    325     342*    384     389     390     435*    441     870     931*    934*    937*    940*
                         943*    946*    949*    952*    955*    958*    961*    964*
MODNAM  000000R          210#
MODSP   000224R          224     262#
MORDEV  001170R          457#    464     489
MORE    000244R          276#    343*    389*    403*    450*    458*    535*    607*    610*    625*
MSGN$ = 104403           264#    445     468     475     707     736
MSGS$ = 104402           264#
MSG$  = 104401           264#
MSSG1   003126R          971     977#
MSSG2   003153R          973     981#
```

```
MSSG3   003200R          967     985#
MSSG4   003232R          969     990#
MSSG5   003304R          975     997#
NOINTR  003122R          445     975#
NOSET   003102R          468     967#
NPASS   001320R          461     490#
NULL  = 000000           264#
NXT     001310R          471     478     484     487#
OPEN  = 000000           211     217     218     219     220     237     238     239     240     241     242     243     244
                         246     248     250     251     253     254     255     264#    298     299     301     302
OTOA$ = 104420           264#
PASCNT  000034R          225#
PIRQS = 000004           264#
POPSP = 005726           264#
POPSP2= 022626           264#
PRTY  = 000000           264#
PRTY0 = 000000           264#
PRTY1 = 000040           264#
PRTY2 = 000100           264#
PRTY3 = 000140           264#
PRTY4 = 000200           264#
PRTY5 = 000240           264#
PRTY6 = 000300           215     264#
PRTY7 = 000340           214     264#
PS    = 177776           264#
PSW   = 177776           264#
PTR1    000400R          324#
PTR2    000402R          325#
PUSH  = 005746           264#
PUSH2 = 024646           264#
P1      000224R          268#    376     651
P2      000226R          269#    796     823     838
P3      000230R          270#
P4      000232R          271#    644     806     808     833     835
RAND$ = 104417           264#
RANNUM  000054R          233#
RBFADR  000334R          304#    399*    400*    454*    687*    688     701     703     715*
RBUF    000312R          295#    400     560*    687     702     715     825     832     924     925*
RBUFEA  000330R          302#    758     762
RBUFPA  000326R          301#    363     399
RBUFSZ  000332R          303#
RBUFVA  000324R          300#    362
RCC     000302R          291#    562*    563*    564*    686     921     922*
REEDIN  00432R           300     454     821     824     828     1007#
RERRST  000640R          383#    514
RESDAT  002402R          394     820#
RESTRT  000534R          252     361#
RES1    000056R          235#
RES2    000060R          236#
ROTCNT  000234R          272#    375*    502     504*    646
ROTPAT  002342R          505     795#
ROTRQS  002276R          492     774#
RQLVL   000354R          312#    378*    427     775     777*    780*    781     783*
RREGS   002070R          413     685#
RSTRT   000112R          252#
SAVRI   000272R          287#    376*    430*    638
```



```
SAVR5   000274R          288#    377*    431*    493     639
SBADR   000102R          245#
SETUP   001410R          348     386     528#
SIMLGO  000236R          273#    437*
SOFCNT  000042R          228#
SOFERS= 104406           264#
SOFPAS  000046R          230#
SPOINT  000032R          224#
SPSIZ = 000040             1#    257
SR1     000016R          217#
SR2     000020R          218#
SR3     000022R          219#
SR4     000024R          220#
START   000410R          223     329#
STAT    000026R          222#
SVDODI  000270R          286#    388*    451
SVR0    000062R          237#
SVR1    000064R          238#
SVR2    000066R          239#
SVR3    000070R          240#
SVR4    000072R          241#
SVR5    000074R          242#
SVR6    000076R          243#
SWAPCC  002704R          509     920#
SYSCNT  000052R          232#
TOCNT   000406R          327#    434*    443*
TRPDBD= 000022           264#
VECTOR  000010R          213#    532
WASADR  000104R          247#
WBFADR  000340R          306#    397*    398*    455*    674*    675     730     732     744*
WBUF    000314R          296#    398     573*    674     731     744     925     926*
WBUFEA  000322R          299#
WBUFPA  000320R          298#    367     397
WBUFVA  000316R          297#    366
WCC     000300R          290#    575*    576*    577*    665     671     922     923*
WDFR    000116R          254#    330*    337*
WDTO    000114R          253#    329*    336*
WREGS   002016R          416     664#
WRTOUT  003362R          297     455     841     844     1006#
XFLAG   000005R          211#
XFR1    000344R          308#    377     642
XFR2    000346R          309#
XFR3    000350R          310#
XFR4    000352R          311#    493     640
.     = 005266R          1005#   1006#   1007#   1009#

. ABS.  000000     000
        005266     001

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 XBEAB0,XBEAB0/SOL/CRF:SYM=DDXCOM,XBEAB0
 RUN-TIME:  2  3  .4 SECONDS
```

RUN-TIME RATIO: 18/5=3.3
CORE USED: 7K (13 PAGES)