DEC/X11 SYSTEM EXERSIZER MACRO DEFINITION MODULE

.REM |

IDENTIFICATION

PRODUCT CODE: AC-E884C-MC

PRODUCT NAME: CXBEACO M7855 BUS TESTER MODULE

PRODUCT DATE: APRIL 1983

MAINTAINER: DEC/X11 SUPPORT GROUP

TABLE OF CONTENTS

1.0 ABSTRACT

2.0 REQUIREMENTS

3.0 PASS DEFINITION

4.0 EXECUTION TIME

5.0 CONFIGURATION REQUIREMENTS

6.0 DEVICE/OPTION SETUP

7.0 TEST SEQUENCE

8.0 OPERATION OPTIONS

9.0 NON STANDARD PRINTOUTS

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 OF 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

```
                ;**
                ;   EDIT        BY              DATE            REASON
                ;
                ;   001         R. SOLER        25-APR-83       DATA ERRORS WHEN RELOCATING
                ;                                               READ BUFFER OVERLAID ON 32K
                ;                                               BOUNDRY DURING RANDOM RELOCA-
                ;                                               ATION. CLOSE PR#317 AND 1328.
                ;                                               EA BITS WHERE NOT BEING SETUP
                ;                                               FOR THE WBUFEA.
                ;--


229 000000                          IOMOD   <BEAC >,170000,510,7,6,,12000,73
    000000                                  MODULE  140000,BEAC ,170000,510,7,6,,12000,73,,,,,
                                            .TITLE  BEAC DEC/X11 SYSTEM EXERCISER MODULE
                                ;           DDXCOM  VERSION  6.4    28-JAN-82
                                            .LIST   BIN
                                ;***************************************************************************
    000000                              BEGIN:
    000000    102     105     101   MODNAM: .ASCII  /BEAC / ;MODULE NAME.
    000003    103     040
    000005    000                   XFLAG:  .BYTE   OPEN            ;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:    OPEN                    ;SWITCH REGISTER 1
    000020  000000                  SR2:    OPEN                    ;SWITCH REGISTER 2
    000022  000000                  SR3:    OPEN                    ;SWITCH REGISTER 3
    000024  000000                  SR4:    OPEN                    ;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:   OPEN                    ;LOC TO SAVE R0.
    000064  000000                  SVR1:   OPEN                    ;LOC TO SAVE R1.
    000066  000000                  SVR2:   OPEN                    ;LOC TO SAVE R2.
    000070  000000                  SVR3:   OPEN                    ;LOC TO SAVE R3.
    000072  000000                  SVR4:   OPEN                    ;LOC TO SAVE R4.
    000074  000000                  SVR5:   OPEN                    ;LOC TO SAVE R5.
    000076  000000                  SVR6:   OPEN                    ;LOC TO SAVE R6.
    000100  000000                  CSRA:   OPEN                    ;ADDR OF CURRENT CSR.
    000102                          SBADR:                          ;ADDR OF GOOD DATA, OR
```

```
000102  000000                  ACSR:   OPEN                    ;CONTENTS OF CSR.
000104                          WASADR:                         ;ADDR OF BAD DATA, OR
000104  000000                  ASTAT:  OPEN                    ;STATUS REG CONTENTS.
000106                          ERRTYP:                         ;TYPE OF ERROR
000106  000000                  ASB:    OPEN                    ;EXPECTED DATA.
000110  000000                  AWAS:   OPEN                    ;ACTUAL DATA.
000112  000534'                 RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
000114  000000                  WDTO:   OPEN                    ;WORDS TO MEMORY PER ITERATION
000116  000000                  WDFR:   OPEN                    ;WORDS FROM MEMORY PER ITERATION
000120  000000                  INTR:   OPEN                    ;# 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:
                                ;*****************************************************************************
230
```

```
                                .ENABL  AMA

000224  000000          P1:     0       ;1ST DATA PATTERN
000226  137777          P2:     137777  ;2ND DATA PATTERN
000230  000000          P3:     0       ;3RD DATA PATTERN
000232  100000          P4:     100000  ;4TH DATA PATTERN
000234  000000          ROTCNT: 0       ;COUNTER FOR # OF TIMES DATA ROTATED
000236  170014          SIMLGO: 170014  ;ADDR TO SET OFF ALL DEVS SIMULTANEOUSLY
000240  000000          DV:     0       ;TEMP STORAGE FOR DVID1
000242  000000          MASK:   0       ;USED TO DROP ANY DEV NOT INTERRUPTING
000244  000000          MORE:   0       ;WORKING STORAGE FOR DVID1
000246  170000          B12T15: 170000  ;USED TO CLEAR BITS 12 THRU 15
000250  000740          BUFSZ:  740     ;SIZE FOR 1 DEV
000252  000360                  360     ;SIZE FOR 2 DEVS
000254  000240                  240     ;SIZE FOR 3 DEVS
000256  000170                  170     ;SIZE FOR 4 DEVS
000260  000140                  140     ;SIZE FOR 5 DEVS
000262  000120                  120     ;SIZE FOR 6 DEVS
000264  000000          DEVCNT: 0       ;TOTAL NUMBER OF DEVS
000266  000000          DODI:   0       ;DATO OR DATI INDICATOR
000270  000000          SVDODI: 0       ;STORAGE FOR DODI
000272  000000          SAVR1:  0       ;LOC TO SAVE R1
000274  000000          SAVR5:  0       ;LOC TO SAVE R5
000276  000000          BYTCC:  0       ;CYCLE COUNT FOR A BYTE TRANSFER
000300  000000          WCC:    0       ;CYCLE COUNT FOR A DATO TRANSFER
000302  000000          RCC:    0       ;CYCLE COUNT FOR A DATI TRANSFER
000304  000000          DOCNT:  0       ;# OF DEVS DOING A DATO/B(S)
000306  000000          DICNT:  0       ;# OF DEVS DOINGA DATI/P(S)
000310  000000          EABIT:  0       ;USED TO SET EXTENDED MEMORY ADDR BITS
000312  000000          RBUF:   0       ;SIZE OF BUFFER FOR DATI/P(S)
000314  000000          WBUF:   0       ;SIZE OF BUFFER FOR DATO/B(S)
000316  003444'         WBUFVA: WRTOUT  ;VIRTUAL ADDR OF WRITE BUFFER
000320  000000          WBUFPA: OPEN    ;PHYSICAL ADDR OF WRITE BUFFER
000322  000000          WBUFEA: OPEN    ;EXT MEM ADDR BITS SET BY MONITOR
000324  004406'         RBUFVA: REEDIN  ;VIRTUAL ADDR OF READ BUFFER
000326  000000          RBUFPA: OPEN    ;PHYSICAL ADDR OF READ BUFFER
000330  000000          RBUFEA: OPEN    ;EXT MEM ADDR BITS SET BY MONITOR
000332  000360          RBUFSZ: 360     ;TOTAL SIZE OF BUFFER RESERVED FOR READS
000334  000000          RBFADR: 0       ;USED TO POINT TO LOC WITHIN RBUFSZ
000336  000000          ENDRBF: 0       ;USED FOR LAST ADDR(PHYSICAL) OF READ BUFF
000340  000000          WBFADR: 0       ;USED TO POINT TO LOC WITHIN WBUFSZ
000342  000000          ENDWBF: 0       ;USED FOR LAST ADDR(PHYSICAL) OF WRITE BUFF
000344  004100          XFR1:   4100    ;FUN 2-DATI-INTR
000346  005500          XFR2:   5500    ;FUN 2-DATOB-INTR
000350  022500          XFR3:   22500   ;FUN 1-DATIP/NO ROT-INTR
000352  003100          XFR4:   3100    ;FUN 1-DATO-INTR
000354  000060          RQLVL:  60      ;USED TO SET REQUEST LEVELS OF DEVS
000356  007740          ERBITS: 7740    ;USED TO CHECK  IF ANY ERROR BITS SET
000360                  DVREGS:
000360  000000          DB:     0       ;DEV DATA REG CONTENTS
000362  000000          CC:     0       ;DEV CYCLE COUNT CONTENTS
000364  000000          BA:     0       ;DEV ADDR REG CONTENTS
000366  000000          CR1:    0       ;DEV CR1 REG CONTENTS
000370  000000          CR2:    0       ;DEV CR2 REG CONTENTS
000372  000374'         BADMEM: BADR    ;BAD MEMORY ADDR
000374  000000          BADR:   0       ;CONTENTS OF BAD MEMORY
```

```
306 000376  177777                          177777
307 000400                          DVMASK:          ;THIS POINTER IS USED SO THE NEXT
308 000400  000240'                 PTR1:   DV       ;TWO LOCATIONS WILL BE PRINTED OUT FOR
309 000402  000242'                 PTR2:   MASK     ;AN ERR WHCIH DEVICES DID NOT INTERRUPT
310 000404  177777                          -1
311 000406  000000                  TOCNT:  0        ;TIME OUT COUNTER
312
313 000410  012737  000361  000114' START:  MOV      #361,WDTO        ;AT LEAST 361 WORDS TO MEM/ITERATION
314 000416  012737  000361  000116'          MOV      #361,WDFR        ;AT LEAST 361 WORDS FROM MEM/ITERATION
315 000424  012737  000001  000120'          MOV      #1,INTR          ;AT LEAST 1 INTERRUPT /ITERATION
316 000432  013700  000014'                  MOV      DVID1,R0         ;SAVE DEV COUNT
317 000436  006200                  2$:     ASR      R0               ;SHIFT IN A COUNT
318 000440  001412                          BEQ      3$               ;BR OUT IF NONE LEFT
319 000442  103375                          BCC      2$               ;BR BACK IF NO BIT IN THIS POSITION
320 000444  062737  000361  000114'          ADD      #361,WDTO        ;361 MORE WORDS TO MEM
321 000452  062737  000361  000116'          ADD      #361,WDFR        ;361 MORE WORDS FROM MEM
322 000460  005237  000120'                  INC      INTR             ;1 MORE INTERRUPT
323 000464  000764                          BR       2$               ;GO CHECK FOR MORE
324 000466  013737  000014'  000240' 3$:     MOV      DVID1,DV         ;MOVE DVID1 TO TEMP STORAGE
325 000474  043737  000246'  000240'          BIC      B12T15,DV        ;ENSURE BITS 12-15 ARE CLEARED
326 000502  013737  000240'  000242'          MOV      DV,MASK          ;INIT MASK BITS EQUAL TO DVID1
327 000510  013737  000240'  000244'          MOV      DV,MORE          ;THEN SET UP WORKING AREA
328 000516  001002                          BNE      1$               ;GO TO 1$ IF THERE ARE DEVS
329 000520  000137  001404'                  JMP      DROP             ;ELSE DROP THE MODULE
330 000524                          1$:
331 000524  004737  001646'                  JSR      PC,CLRREG        ;CLEAR ALL DEV REGS
332 000530  004737  001410'                  JSR      PC,SETUP         ;SET UP AVAILABLE VEC LOCATIONS AND
333                                                                   ;GET BUFFER SIZES FOR DATO(S) & DATI(S)
334
335                                 ;*****************************************************
336                                 ;NOTE: IN THE FOLLOWING COMMENTS THE TERM READ IS USED
337                                 ;TO DENOTE DATI OR DATIP AND THE TERM WRITE DENOTES
338                                 ;DATO OR DATOB.
339                                 ;*****************************************************
                                    ;*****************************************************
340                                 ;THIS IS THE LOCATION THE PROGRAM WILL PROCEED
341                                 ;WITH AFTER AN END OF PASS
342
343 000534                          RESTRT:
344 000534  104415  000000' 000324'          GETPA$,BEGIN, RBUFVA             ;GET PHYSICAL ADDRESS FROM 16-BIT RBUFVA
345 000542  013737  000326' 000336'          MOV      RBUFPA,ENDRBF   ;MOVE READ BUFF ADDR TO END-OF-READ BUFF REG
346 000550  062737  000736  000336'          ADD      #736,ENDRBF     ;ADD 736 TO GET LAST ADDR OF BUFFER
347
348 000556  104415  000000' 000316'          GETPA$,BEGIN, WBUFVA             ;GET PHYSICAL ADDRESS FROM 16-BIT WBUFVA
349 000564  013737  000320' 000342'          MOV      WBUFPA,ENDWBF   ;MOVE WRITE BUFF ADDR TO END-OF-WRITE BUFF REG
350 000572  062737  000736  000342'          ADD      #736,ENDWBF     ;ADD 736 TO GET LAST ADDR OF BUFFER
351
352
353                                 ;RS001  JSR      PC,FABITS        ;SET BITS 0 & 1 OF REA EQUAL TO
354                                                                   ;BITS 4 & 5 OF RBUFEA RESPECTIVELY
355 000600  012737  000001  000266'          MOV      #1,DODI          ;INIT XFER INDICATOR TO 1 SO THE
356                                                                   ;1ST INC WILL SIGNIFY A DATI
357 000606  005037  000234'                  CLR      ROTCNT           ;CLEAR PATTERN ROTATION COUNT
358 000612  012737  000224'  000272'          MOV      #P1,SAVR1        ;INIT SAVR1 TO 1ST PATTERN
359 000620  012737  000344'  000274'          MOV      #XFR1,SAVR5      ;INIT SAVR5 TO 1ST XFER FUNCTION
360 000626  012737  000060  000354'          MOV      #60,RQLVL        ;INIT REQUEST LEVEL TO NPR/INTR BR7
```

```
                                        ;THIS IS THE LOCATION THE PROGRAM WILL PROCEED WITH
                                        ;AFTER AN ITERATION
                                        
000634                                  REREST:
000634  023737  000242' 000240'                CMP     MASK,DV         ;# OF DEVS THAT SHOULD INTR=# OF DEVS THAT DID?
000642  001402                                BEQ     1$              ;IF EQUAL,GO TO 2$
000644  004737  001410'                        JSR     PC,SETUP        ;ELSE GET NEW DEV COUNTS & BUFFER SIZES
000650                                  1$:
000650  013737  000266' 000270'                MOV     DODI,SVDODI     ;SAVE DODI SETTING
000656  013737  000242' 000244'                MOV     MASK,MORE       ;RESET MORE BITS
000664  013737  000242' 000240'                MOV     MASK,DV         ;IF ANY DEVS DROPPED, ALTER DV
000672  001002                                BNE     2$              ;IF NOT, CONTINUE ON
000674  000137  001404'                        JMP     DROP            ;ELSE DROP MODULE
000700                                  2$:
000700  004737  002464'                        JSR     PC,RESDAT       ;RESTORE DATA IN READ BUFFER
                                                                        ;AND CLEAR WRITE BUFFER
000704  013700  000006'                        MOV     ADDR,R0         ;PUT DEVICE ADR INTO REG. 0
000710  013737  000320' 000340'                MOV     WBUFPA,WBFADR   ;USE WBFADR AS WRITE BUFF ADDR
000716  163737  000314' 000340'                SUB     WBUF,WBFADR     ;INIT TO WRITE BUFF LESS WBUF
000724  013737  000326' 000334'                MOV     RBUFPA,RBFADR   ;USE RBFADR AS READ BUFF ADDR
000732  163737  000312' 000334'                SUB     RBUF,RBFADR     ;INIT TO READ BUFF LESS RBUF
                                        
000740                                  3$:
000740  006137  000244'                        ASR     MORE            ;CHECK FOR ANOTHER DEV
000744  103401                                BCS     4$              ;IF THERE, CONTINUE
000746  001402                                BEQ     GO              ;AND IF NOT, GO TO GO
000750  062700  000020                        ADD     #20,R0          ;ELSE ADD 20 FOR NEXT ADDR
000754  000771                                BR      3$              ;AND CHECK FOR MORE
000756                                  4$:
000756  004737  001742'                        JSR     PC,CHEKPX       ;SEE IF R1 & R5 NEED INITIALIZATION
000762  005237  000266'                        INC     DODI            ;INC XFER INDICATOR
000766  032737  000001  000266'               BIT     #BIT0,DODI      ;IF BIT 0 OF DODI IS 1
000774  001003                                BNE     5$              ;GO SET UP REGS FOR DATO(S)
000776  004737  002070'                        JSR     PC,RREGS        ;ELSE SET UP REGS FOR DATI(S)
001002  000402                                BR      6$              ;GO LOAD THE REGISTERS
001004                                  5$:
001004  004737  002016'                        JSR     PC,WREGS        ;SET REGS FOR WRITE XFER INFO
                                        
                                        ;AT THIS POINT THE DEVICE REGISTERS WILL BE LOADED TO DO
                                        ;TRANSFERS
                                        
001010                                  6$:
001010  004737  002234'                        JSR     PC,EABITS       ;SET EXTENDED ADDRESS BITS IN EABIT     ;RS001
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  000732                                BR      3$              ;GO LOAD ANOTHER DEVICE
001054                                  GO:
001054  005037  000406'                        CLR     TOCNT           ;CLEAR TIME OUT COUNT
```

```
418 001060  005037  000242'                 CLR     MASK            ;CLEAR ALL BITS IN MASK
419                                                                 ;TO BE RESET IN INTR SERV RTNS
420 001064  005277  177146                  INC     @SIMLGO         ;SET OFF ALL DEVS
421 001070                          BRK:
422 001070  104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
    001074  104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
423 001100  023737  000240'  000242'        CMP     DV,MASK         ;ALL DEVS INTR?
424 001106  001411                          BEQ     DATACK          ;IF YES, GO TO DATACK
425 001110  005337  000406'                 DEC     TOCNT           ;ELSE DEC TIME OUT COUNT
426 001114  001365                          BNE     BRK             ;AND BREAK IF NOT 0
427 001116  104403  000000'  003204'        MSGN$,BEGIN,NOINTR      ;ASCII MESSAGE CALL WITH COMMON HEADER
428 001124  005004                          CLR     R4              ;R4 INVALID IN ERR REPORT
429 001126  004737  002564'                 JSR     PC,GETDEV       ;REPORT NON-INTR'G DEVICES
430
431 001132                          DATACK:
432 001132  013737  000240'  000244'        MOV     DV,MORE         ;SET UP MORE
433 001140  013737  000270'  000266'        MOV     SVDODI,DODI     ;RESTORE DODI TO SETTING WHEN
434                                                                 ;LOADING 1ST DEV
435 001146  013700  000006'                 MOV     ADDR,R0         ;INIT R0 TO 1ST DEV  ADDR
436 001152  012737  004406'  000334'        MOV     #REEDIN,RBFADR  ;RBFADR=1ST LOC IN READ BUFFER
437 001160  012737  003444'  000340'        MOV     #WRTOUT,WBFADR  ;WBFADR=1ST LOC IN WRITE BFFER
438 001166  005003                          CLR     R3              ;USED IN DODATA RTN
439 001170                          MORDEV:
440 001170  006237  000244'                 ASR     MORE            ;CHECK FOR OTHER DEVS
441 001174  103405                          BCS     2$              ;IF THERE CONTINUE
442 001176  001001                          BNE     1$              ;IF SOME LEFT,GO CHECK FOR MORE
443 001200  000447                          BR      NPASS           ;ELSE GO TO END OF PASS RTN
444 001202                          1$:
445 001202  062700  000020                  ADD     #20,R0          ;GET NEXT DEV ADDR
446 001206  000770                          BR      MORDEV          ;GO CHECK FOR THAT DEV
447 001210                          2$:
448 001210  105760  000006                  TSTB    6(R0)           ;IS READY BIT SET IN CR1?
449 001214  100407                          BMI     3$              ;IF SET, CONTINUE
450 001216  104403  000000'  003164'        MSGN$,BEGIN,NOSET       ;ASCII MESSAGE CALL WITH COMMON HEADER
451 001224  005004                          CLR     R4              ;R4 INVALID IN ERR REPORT
452 001226  004737  000[illegible]0'        JSR     PC,BDDATA       ;ELSE TYPE OUT CONTENTS OF ALL DEV REGS
453 001232  000425                          BR      NXT             ;AND GO CHECK FOR NEXT DEV
454 001234                          3$:
    001234  033760  000356'  000016        BIT     ERBITS,16(R0)   ;SEE IF ANY ERROR BITS SET
4'  001242  001407                          BEQ     4$              ;IF NONE SET, CONTINUE
457 001244  104403  000000'  003170'        MSGN$,BEGIN,ERRSET      ;ASCII MESSAGE CALL WITH COMMON HEADER
458 001252  005004                          CLR     R4              ;R4 INVALID IN ERR REPORT
459 001254  004737  002640'                 JSR     PC,BDDATA       ;ELSE TYPE OUT CONTENTS OF ALL DEV REGS
460 001260  000413                          BR      NXT             ;AND GO CHECK FOR NEXT DEV
461 001262                          4$:
462 001262  005237  000266'                 INC     DODI            ;INC XFER INDICATOR
463 001266  032737  000001  000266'        BIT     #BIT0,DODI      ;DATI OR DATO ?
464 001274  001003                          BNE     5$              ;IF DATO,GO  DO IT
465 001276  004737  002110'                 JSR     PC,DIDATA       ;CHECK DATI XFERS
466 001302  000402                          BR      NXT             ;CHECK FOR NEXT DEV
467 001304                          5$:
468 001304  004737  002162'                 JSR     PC,DODATA       ;CHECK DATO XFERS
469 001310                          NXT:
470 001310  062700  000020                  ADD     #20,R0          ;ADD 20 FOR NEXT DEV ADDR
471 001314  000137  001170'                 JMP     MORDEV          ;GO CHECK FOR ANOTHER DEV
472 001320                          NPASS:
473 001320  004737  001646'                 JSR     PC,CLRREG       ;CLEAR ALL DEV REGS
```

```
474 001324  004737  002360'                 JSR     PC,ROTRQS       ;SHIFT REQUEST TO NEXT BR LEVEL
475 001330  022737  000352' 000274'         CMP     #XFR4,SAVR5     ;HAS LAST XFER BEEN DONE?
476 001336  001010                          BNE     1$              ;IF NOT,  CONTINUE
477                                 ;*************************************************
                                    ;*************************************************
478                                 ;AFTER P2 & P4 HAVE BEEN ROTATED 16 TIMES, ALLOW THE POINTER
479                                 ;R1 TO INCREMENT THRU THE END OF THE READ BUFFER BUT DON'T ALLOW ROTATION
480                                 ;*************************************************
                                    ;*************************************************
481
482 001340  022737  000020  000234'         CMP     #16.,ROTCNT     ;ELSE HAS P2&P4 BEEN ROTATED 16 TIMES?
483 001346  001404                          BEQ     1$              ;IF YES DON'T ROTATE PATTERNS ANYMORE
484 001350  005237  000234'                 INC     ROTCNT          ;ELSE INC ROTATION COUNT, AND
485 001354  004737  002424'                 JSR     PC,ROTPAT       ;ROTATE DATA PATTERNS
486 001360                          1$:
487 001360  032737  000001  000264'         BIT     #BIT0,DEVCNT    ;IS THERE AN ODD # OF DEVS?
488 001366  001402                          BEQ     2$              ;IF NOT,GO TO 5$
489 001370  004737  003766'                 JSR     PC,SWAPCC       ;ELSE SWAP RCC WITH WCC & RBUF
490                                                                 ;WITH WBUF FOR CORRECT XFER INFO
491 001374                          2$:
    001374  104413  000000'                 ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
                                                                    ;MONITOR SHALL TEST END OF PASS
492 001400  000137  000034'                 JMP     REREST          ;ELSE GO TO REREST
493                                 ;*************************************************************
494
495 001404                          DROP:
496 001404  104410  000000'                 END$,BEGIN              ;
497
498                                 ;*************************************************
                                    ;*************************************************
499                                 ;THIS ROUTINE WILL SET UP THE DEVICE INTO VECTORS, COUNT THE
500                                 ;NUMBER OF DEVICES, SET THE TRANSFER SIZE OF THE READ AND WRITE
501                                 ;BUFFERS, AND SET UP THE READ AND WRITE CYCLE COUNTS.
502                                 ;*************************************************
                                    ;*************************************************
503
504 001410                          SETUP:
505 001410  005037  000306'                         CLR     DICNT           ;CLEAR REG COUNTING DEVS DOING DATI(S)
506 001414  005037  000304'                         CLR     DOCNT           ;CLEAR REG COUNTING DEVS DOING DATO(S)
507 001420  012700  003024'                         MOV     #ISR1,R0        ;SET R0 TO 1ST INTR SERVICE RTN
508 001424  013701  000010'                         MOV     VECTOR,R1       ;SET R1 TO 1ST INTR VEC ADDR
509 001430  012737  000001  000266'                 MOV     #1,DODI         ;INIT XFR INDICATOR TO 1
510 001436                          CKMORE:
511 001436  006237  000244'                         ASR     MORE            ;CHECK FOR A DEVICE
512 001442  103406                                  BCS     1$              ;IF THERE, CONTINUE
513 001444  001420                                  BEQ     2$              ;IF NONE LEFT AT ALL, GET OUT
514 001446  062700  000010                          ADD     #10,R0          ;ELSE INC INTR SERV POINTER BY 10
515 001452  062701  000004                          ADD     #4,R1           ;AND INC INTR VEC LOC BY 4
516 001456  000767                                  BR      CKMORE          ;GO BACK AND CHECK FOR ANOTHER DEV
517 001460                          1$:
518 001460  005237  000266'                         INC     DODI            ;INC XFER INDICATOR
519 001464  004737  001622'                         JSR     PC,INCCNT       ;INC DATI OR DATO COUNT
520 001470  010021                                  MOV     R0,(R1)+        ;MOVE ADDR OF INTR SERV RTN TO VEC LOC
521 001472  113721  000012'                         MOVB    BR1,(R1)+       ;MOVE BR1 VALUE TO PSW VEC
522 001476  105721                                  TSTB    (R1)+           ;GET BACK TO AN EVEN ADDR
523 001500  062700  000010                          ADD     #10,R0          ;ADD 10 TO R0 FOR NEXT INTR SERV RTN
524 001504  000754                                  BR      CKMORE          ;SEE IF THERE ARE MORE DEVS
```

```
                                2$:
001506                                  MOV     DODI,DEVCNT     ;DEV XFER INDICATOR HAS # OF DEVS
001506  013737  000266' 000264'          DEC     DEVCNT          ;+ 1, SO DEC DEVCNT BY 1
001514  005337  000264'
001520                          GFIBUF:
001520  012700  000246'                  MOV     #BUFSZ-2,R0     ;SET R0 TO SIZE TABLE
001524  005001                          CLR     R1              ;CLEAR R1
001526                          1$:
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     2$              ;GO USE THE VALUE R0 POINTS TO

001566  020137  000304'                  CMP     R1,DOCNT        ;R1=# OF DATO DEVS ?
001572  001402                          BEQ     2$              ;YES, CONTINUE
001574  005301                          DEC     R1              ;NO, DEC R1
001576  005740                          TST     -(R0)           ;AND DEC POINTER
001600                          2$:
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
                                ;*************************************************************
                                ;*************************************************************
```

```
578 001646                              CLRREG:
579 001646  013737  000240' 000244'            MOV     DV,MORE         ;RESTORE MORE
580 001654  013701  000006'                     MOV     ADDR,R1         ;INIT R1 TO 1ST DEV ADDR
581 001660                              1$:
582 001660  006237  000244'                     ASR     MORE            ;IS THERE A DEVICE?
583 001664  103404                             BCS     2$              ;IF YES,GO SET IT UP
584 001666  001421                             BEQ     3$              ;IF NONE LEFT, - GET OUT
585 001670  062701  000020                     ADD     #20,R1          ;ELSE INC R1 TO NEXT DEV ADDR
586 001674  000771                             BR      1$              ;AND GO SEE IF IT'S THERE
587 001676                              2$:
588 001676  005011                             CLR     (R1)            ;CLEAR DATA BUFFER REG
589 001700  005061  000002                     CLR     2(R1)           ;CLEAR CYCLE COUNT REG
590 001704  005061  000004                     CLR     4(R1)           ;CLEAR BUFFER ADDR REG
591 001710  005061  000006                     CLR     6(R1)           ;CLEAR CR1 REG
592 001714  005061  000010                     CLR     10(R1)          ;CLEAR ERROR CLEAR REG
593 001720  005061  000016                     CLR     16(R1)          ;CLEAR CR2 REG
594 001724  062701  000020                     ADD     #20,R1          ;INC R1 TO NEXT DEV ADDR
595 001730  000753                             BR      1$              ;GO CHECK FOR ANOTHER DEV
596 001732                              3$:
597 001732  013737  000240' 000244'            MOV     DV,MORE         ;RESTORE MORE
598 001740  000207                             RTS     PC              ;RETURN
599                                     ;*******************************************************
600
601
602                                     ;*******************************************************
                                        ;*******************************************************
603                                     ;THIS ROUTINE WILL KEEP TRACK OF THE DATA PATTERNS AND TRANSFER
604                                     ;FUNCTIONS USED TO LOAD THE DEVICES
605                                     ;*******************************************************
                                        ;*******************************************************
606
607 001742                              CHEKPX:
608 001742  013701  000272'                     MOV     SAVR1,R1        ;RESTORE R1 TO PATTERN IN SAVR1
609 001746  013705  000274'                     MOV     SAVR5,R5        ;RESTORE R5 TO XFR FUNCTION IN SAVR5
610 001752  022705  000352'                     CMP     #XFR4,R5        ;DOES R5 POINT TO LAST XFER?
611 001756  100002                             BPL     1$              ;IF NOT, CONTINUE
612 001760  012705  000344'                     MOV     #XFR1,R5        ;ELSE INIT TO 1ST XFER
613 001764                              1$:
614 001764  022701  000232'                     CMP     #P4,R1          ;IF R1 DOES NOT EXCEED LAST PATTERN
615 001770  100011                             BPL     3$              ;ADDRESS, EXIT ROUTINE
616 001772  022737  000020  000234'            CMP     #16.,ROTCNT     ;HAVE P2&P4 BEEN ROTATED 16 TIMES?
617 002000  001003                             BNE     2$              ;IF NOT,INIT R1
618 002002  023701  000336'                     CMP     ENDRBF,R1       ;IS R1 AT LAST ADDR OF READ BUFF?
619 002006  100002                             BPL     3$              ;IF LESS,GO TO EXINIT
620 002010                              2$:
621 002010  012701  000224'                     MOV     #P1,R1          ;ELSE INIT R1 TO 1ST DATA PATTERN
622 002014                              3$:
623 002014  000207                             RTS     PC              ;RETURN
624                                     ;*******************************************************
625
626                                     ;*******************************************************
                                        ;*******************************************************
627                                     ;THIS ROUTINE WILL DETERMINE IF THE TRANSFER IS A DATO OR A DATOB
628                                     ;AND SET THE CYCLE COUNT APPROXIMATELY VIA R2 AND ALSO SET R3 TO
629                                     ;THE CORRECT WRITE BUFFER ADDRES
630                                     ;*******************************************************
                                        ;*******************************************************
```

```
                                WREGS:
002016                                  MOV     #WCC,R2         ;SET R2 FOR A DATO CYCLE COUNT
002016  012702  000300'                 BIT     #BIT8,(R5)      ;IF BIT 8 OF XFER FUNC IS 0
002022  032715  000400                  BEQ     1$              ;ALL SET, FINISH UP
002026  001412                          BIT     BIT9,(R5)       ;IF BIT 9 OF XFER FUNCTION
002030  033715  001000                  BEQ     1$              ;ALL SET, FINISH UP
002034  001407                          MOV     #BYTCC,R2       ;ELSE SET R2=ADDR OF BYTCC
002036  012702  000276'                 MOV     WCC,BYTCC       ;MOVE VALUE OF WCC TO BYTCC AND
002042  013737  000300'  000276'        ASL     BYTCC           ;DOUBLE BY SHIFTING LEFT
002050  006337  000276'                1$:                     ;ELSE LEAVE R2 = WCC
002054                                  ADD     WBUF,WBFADR     ;INC WBFADR ADDR BY VALUE IN WBUF
002054  063737  000314'  000340'        MOV     #WBFADR,R3      ;SET R3 = TO ADDR OF WBFADR
002062  012703  000340'                  RTS     PC              ;RETURN
002066  000207                        ;***************************************************
                                      ;***************************************************
                                      ;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'  003174'        MSGN$,BEGIN,DIERR       ;ASCII MESSAGE CALL WITH COMMON HEADER
002136  005744                          TST     (R4)            ;RETURN R4 TO ERR LOCATION
002140  004737  002640'                  JSR     PC,BDDATA       ;AND TYPE OUT CONTENTS OF REGS
002144  000402                          BR      3$              ;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' 003200'       MSGN$,BEGIN,DOERR       ;ASCII MESSAGE CALL WITH COMMON HEADER
002210  005744                        TST     -(R4)           ;RETURN R4 TO ERR LOCATION
002212  004737  002640'               JSR     PC,BDDATA
002216  000402                        BR      3$
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
                                ;*************************************************************
                                ;*************************************************************
                                ;IF DATO (BITO EQ 1) SET EA BITS IN WBUFEA                       ;RS001
                                ;IF DATI (BITO EQ 0) SET EA BITS IN RBUFEA                       ;RS001
                                ;*************************************************************

002234                          EABITS:                                                         ;RS001
002234  005037  000310'                CLR     EABIT           ;CLEAR EXTENDED MEMORY BITS     ;RS001
002240  032737  000001  000266'       BIT     #BIT0,DODI      ;DATO?                          ;RS001
002246  001003                        BNE     1$              ;YES, SET EA BITS IN WBUFEA     ;RS001
002250  004737  002264'               JSR     PC,REABITS      ;NO, SET EA BITS IN RBUFEA      ;RS001
002254  000402                        BR      2$              ;RETURN                         ;RS001
002256  004737  002322'       1$:     JSR     PC,WEABITS      ;SET EA BITS IN WBUFEA          ;RS001
002262  000207                2$:     RTS     PC              ;RETURN                         ;RS001

                                ;*****************************************************
                                ;*****************************************************
                                ;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.
                                ;*****************************************************
                                ;*****************************************************

002264                          REABITS:
002264  032737  000020  000330'        BIT     #BIT4,RBUFEA    ;IS BIT 4 OF RBUFEA SET?
002272  001403                        BEQ     1$              ;IF NOT, CONTINUE
002274  052737  000001  000310'       BIS     #BIT0,EABIT     ;ELSE SET BIT 0 OF EABIT
002302  032737  000040  000330'  1$:  BIT     #BIT5,RBUFEA    ;IS BIT 5 OF RBUFEA SET?
002310  001403                        BEQ     2$              ;IF NOT, RETURN
002312  052737  000002  000310'       BIS     #BIT1,EABIT     ;ELSE SET BIT 1 OF EABIT
```

```
002320  000207                  2$:     RTS     PC              ;RETURN

                                ;*************************************************************
                                ;THIS ROUTINE WILL PERFORM THE SAME FUNCTION AS THE REABITS      ;RS001
                                ;ROUTINE EXCEPT TO THE WBUFEA.                                  ;RS001
                                ;*************************************************************

002322                          WEABITS:                                                        ;RS001
002322  032737  000020  000322' BIT     @BIT4,WBUFEA    ;IS BIT 4 OF WBUFEA SET?                ;RS001
002330  001403                  BEQ     1$              ;IN NOT, CONTINUE                       ;RS001
002332  052737  000001  000310' BIS     @BIT0,EABIT     ;ELSE SET BIT 0 OF EABIT?               ;RS001
002340  032737  000040  000322' 1$:     BIT     @BIT5,WBUFEA    ;IS BIT 5 OF WBUFEA SET?        ;RS001
002346  001403                  BEQ     2$              ;IF NOT, RETURN                         ;RS001
002350  052737  000002  000310' BIS     @BIT1,EABIT     ;ELSE SET BIT 1 OF EABIT                ;RS001
002356  000207                  2$:     RTS     PC              ;RETURN                         ;RS001

                                ;*************************************************************

                                ;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

002360                          ROTRQS:
002360  022737  000060  000354' CMP     @60,RQLVL       ;IS REQUEST AT NPR WITH BR7 INTR?
002366  001004                  BNE     1$              ;IF NOT, CONTINUE
002370  012737  000020  000354' MOV     @20,RQLVL       ;ELSE SET REQUEST LEVEL TO BR7
002376  000411                  BR      2$              ;AND RETURN
002400                  1$:
002400  006237  000354'         ASR     RQLVL           ;SHIFT REQUEST LEVEL TO NEXT BR
002404  022737  000001  000354' CMP     @1,RQLVL        ;IF REQUEST LEVEL DOES NOT = 1
002412  001003                  BNE     2$              ;THEN RETURN
002414  012737  000060  000354' MOV     @60,RQLVL       ;ELSE RESET REQUEST TO NPR/BR7 INTR
002422                  2$:
002422  000207                  RTS     PC              ;RETURN
                                ;*************************************************************


                                ;*******************************************************
                                ;*******************************************************
                                ;THIS ROUTINE WILL ROTATE THE DATA PATTERNS TO TRY TO EXERCISE
                                ;THE DATA LINES.
                                ;*******************************************************
                                ;*******************************************************

002424                          ROTPAT:
002424  012701  000226'         MOV     @P2,R1          ;INIT R1 TO ADDR OF P2
002430                  1$:
002430  032711  000001          BIT     @BIT0,(R1)      ;IF BIT 0 OF PATTERN IS 0
002434  001402                  BEQ     2$              ;GO CLEAR C-BIT
002436  000261                  SEC                     ;ELSE SET C-BIT
002440  000401                  BR      3$              ;AND CONTINUE
002442                  2$:
002442  000241                  CLC                     ;CLEAR C-BIT
002444                  3$:
002444  006011                  ROR     (R1)            ;ROTATE DATA TO THE RIGHT
002446  022701  000232'         CMP     @P4,R1          ;DOES R1 POINT TO LAST PATTERN?
```

```
 792 002452  001403                        BEQ     4$              ;IF IT DOES,EXIT
 793 002454  012701  000232'                MOV     #P4,R1          ;ELSE SET R1=LAST PATTERN ADDR
 794 002460  000763                        BR      1$              ;AND GO BACK FOR MORE
 795 002462                          4$:
 796 002462  000207                        RTS     PC              ;RETURN
 797                                 ;*************************************************************
 798
 799                                 ;THIS ROUTINE WILL SET UP THE READ BUFFER WITH ONE OR BOTH DATA
 800                                 ;PATTERNS DEPENDING ON THE NUMBER OF DEVICES AND CLEAR THE WRITE
 801                                 ;BUFFER SO THE FOLLOWING TRANSFERS CAN BE COMPARED CORRECTLY.
 802                                 ;*****************************************************
                                    ;*****************************************************

 803
 804 002464                          RESDAT:
 805 002464  012702  004406'                MOV     #REEDIN,R2      ;SET R2=1ST ADDR OF READ BUFF
 806
 807 002470  013703  000226'                MOV     P2,R3           ;SET R3= VALUE IN P2
 808 002474  012704  004406'                MOV     #REEDIN,R4      ;SET R4= ADDR OF READ BUFFER
 809 002500  063702  000312'                ADD     RBUF,R2         ;INC R2 TO LAST XFER ADDR
 810 002504                          1$:
 811 002504  010324                        MOV     R3,(R4)+        ;MOVE PATTERN INTO READ BUFF ADDR
 812 002506  022704  005344'                CMP     #REEDIN+736,R4  ;IS R4 LAST ADDR IN READ BUFF?
 813 002512  103415                        BLO     3$              ;IF IT IS,GO TAKE CARE OF IT
 814 002514  020204                        CMP     R2,R4           ;IS R4 AT LAST XFER ADDR?
 815 002516  101372                        BHI     1$              ;IF NOT,GO TO NEXT ADDR
 816 002520  063702  000312'                ADD     RBUF,R2         ;INC R2 TO LAST ADDR OF NEXT XFER
 817 002524  023703  000232'                CMP     P4,R3           ;IS R3 = VALUE IN P4?
 818 002530  001403                        BEQ     2$              ;IF IT IS,GO TAKE CARE OF IT
 819 002532  013703  000232'                MOV     P4,R3           ;ELSE SET R3 = P4
 820 002536  000762                        BR      1$              ;AND GO AGAIN
 821 002540                          2$:
 822 002540  013703  000226'                MOV     P2,R3           ;RESET R3 = P2
 823 002544  000757                        BR      1$              ;AND GO AGAIN
 824 002546                          3$:
 825 002546  012704  003444'                MOV     #WRTOUT,R4      ;SET R4 TO WRITE BUFFER
 826 002552                          4$:
 827 002552  005024                        CLR     (R4)+           ;CLEAR BUFFER
 828 002554  022704  004402'                CMP     #WRTOUT+736,R4  ;END OF BUFFER?
 829 002560  101374                        BHI     4$              ;IF NOT,CLEAR NEXT WORD OF BUFF
 830 002562  000207                        RTS     PC              ;RETURN
 831                                 ;*************************************************************
 832
 833                                 ;*****************************************************
                                    ;*****************************************************
 834                                 ;THIS ROUTINE WILL COMPARE BITS IN DV WITH THOSE IN MASK TO
 835                                 ;DETERMINE WHICH DEVICES DID NOT INTERRUPT, THEN PRINT THEM.
 836                                 ;*****************************************************
                                    ;*****************************************************

 837 002564                          GETDEV:
 838
 839 002564  012701  000001                MOV     #1,R1           ;SET BIT 0 IN R1
 840 002570  013700  000006'                MOV     ADDR,R0         ;INIT R0 TO 1ST DEV ADDR
 841 002574  005004                        CLR     R4              ;CLEAR R4 AS NOT USED IN BDDATA RTN
 842 002576                          1$:
 843 002576  030137  000240'                BIT     R1,DV           ;BIT TEST DV TO SEE IF DEV THERE
 844 002602  001007                        BNE     3$              ;IF THERE, CHECK MASK REG.
 845 002604                          2$:
```

```
                                        ASL     R1              ;ELSE SHIFT TO NEXT BIT
                                        BIT     #10000,R1       ;AT BIT 12?
                                        BNE     GETOUT          ;IF YES, EXIT
                                        ADD     #20,R0          ;ELSE GET NEXT DEV ADDR
                                        BR      1$              ;AND SEE IF ITS THERE
                                3$:
                                        BIT     R1,MASK         ;DID DEV INTR?
                                        BNE     2$              ;IF YES, GO CHECK NEXT ONE
                                        JSR     PC,BDDATA       ;ELSE REPORT ALL ITS REGS.
                                        BR      2$              ;THEN GO CHECK NEXT BIT
                                GETOUT:
                                        RTS     PC              ;EXIT
                                ;*************************************************************

                                ;*****************************************************
                                ;*****************************************************
                                ;THIS ROUTINE WILL LIST ALL THE CONTENTS OF THE DEVICE REGISTERS
                                ;ON REGISTERING AN INCORRECT DATA COMPARISON OR FOR AN INTERRUPT
                                ;ON ERROR
                                ;*****************************************************
                                ;*****************************************************
                                BDDATA:
                                        MOV     R4,BADR         ;SET UP BAD MEM FOR ERR TYPEOUT
                                        MOV     R0,DB           ;DB HAS DEV DATA REG ADDR
                                        MOV     DB,CC           ;MOVE DB TO CC AND ADD 2
                                        ADD     #2,CC           ;TO GET DEV CYCLE COUNT REG ADDR
                                        MOV     CC,BA           ;MOVE CC TO BA AND ADD 2
                                        ADD     #2,BA           ;TO GET DEV BUFF ADDR REG ADDR
                                        MOV     BA,CR1          ;MOVE BA TO CR1 AND ADD 2
                                        ADD     #2,CR1          ;TO GET DEV CR1 REG ADDR
                                        MOV     CR1,CR2         ;MOVE CR1 TO CR2 AND ADD 10
                                        ADD     #10,CR2         ;TO GET DEV CR2 REG ADDR
                                        MOV     CR1,CSRA        ;GET BECR1 ADDR
                                        MOV     @CR1,ACSR       ;GET CONTENTS OF BECR1

                                        MOV     #1,ERRTYP       ;DATA ERROR
                                        ;*************************************************************
                                        HRDER$,BEGIN,DVREGS     ;
                                        ;*************************************************************

                                        CLR     10(R0)          ;CLEAR ALL ERRORS
                                        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.
                                ;*****************************************************
                                ;*****************************************************
                                SWAPCC:
```

```
897 002766  013702  000302'                 MOV     RCC,R2          ;SAVE RCC
898 002772  013737  000300' 000302'         MOV     WCC,RCC         ;GET NEW RCC FROM WCC
899 003000  010237  000300'                 MOV     R2,WCC          ;GET NEW WCC FROM OLD RCC
900 003004  013702  000312'                 MOV     RBUF,R2         ;SAVE RBUF
901 003010  013737  000314' 000312'         MOV     WBUF,RBUF       ;GET NEW RBUF FROM WBUF
902 003016  010237  000314'                 MOV     R2,WBUF         ;GET NEW WBUF FROM OLD RBUF
903 003022  000207                          RTS     PC              ;RETURN
904
905
906 003024                           ISR1:
    003024  052737  000001  000242'         BIS     #BIT0,MASK      ;SET BIT0 TO INDICATE DEV 1 INTERR'D
    003032  000002                          RTI                     ;RETURN
907 003034                           ISR2:
    003034  052737  000002  000242'         BIS     #BIT1,MASK      ;SET BIT1 TO INDICATE DEV 2 INTERR'D
    003042  000002                          RTI                     ;RETURN
908 003044                           ISR3:
    003044  052737  000004  000242'         BIS     #BIT2,MASK      ;SET BIT2 TO INDICATE DEV 3 INTERR'D
    003052  000002                          RTI                     ;RETURN
909 003054                           ISR4:
    003054  052737  000010  000242'         BIS     #BIT3,MASK      ;SET BIT3 TO INDICATE DEV 4 INTERR'D
    003062  000002                          RTI                     ;RETURN
910 003064                           ISR5:
    003064  052737  000020  000242'         BIS     #BIT4,MASK      ;SET BIT4 TO INDICATE DEV 5 INTERR'D
    003072  000002                          RTI                     ;RETURN
911 003074                           ISR6:
    003074  052737  000040  000242'         BIS     #BIT5,MASK      ;SET BIT5 TO INDICATE DEV 6 INTERR'D
    003102  000002                          RTI                     ;RETURN
912 003104                           ISR7:
    003104  052737  000100  000242'         BIS     #BIT6,MASK      ;SET BIT6 TO INDICATE DEV 7 INTERR'D
    003112  000002                          RTI                     ;RETURN
913 003114                           ISR8:
    003114  052737  000200  000242'         BIS     #BIT7,MASK      ;SET BIT7 TO INDICATE DEV 8 INTERR'D
    003122  000002                          RTI                     ;RETURN
914 003124                           ISR9:
    003124  052737  000400  000242'         BIS     #BIT8,MASK      ;SET BIT8 TO INDICATE DEV 9 INTERR'D
    003132  000002                          RTI                     ;RETURN
915 003134                           ISR10:
    003134  052737  001000  000242'         BIS     #BIT9,MASK      ;SET BIT9 TO INDICATE DEV 10 INTERR'D
    003142  000002                          RTI                     ;RETURN
916 003144                           ISR11:
    003144  052737  002000  000242'         BIS     #BIT10,MASK     ;SET BIT10 TO INDICATE DEV 11 INTERR'D
    003152  000002                          RTI                     ;RETURN
917 003154                           ISR12:
    003154  052737  004000  000242'         BIS     #BIT11,MASK     ;SET BIT11 TO INDICATE DEV 12 INTERR'D
    003162  000002                          RTI                     ;RETURN
918
919 003164  003262'                  NOSET:  MSSG3
920 003166  177777                          -1
921 003170  003314'                  ERRSET: MSSG4
922 003172  177777                           1
923 003174  003210'                  DIERR:  MSSG1
924 003176  177777                          -1
925 003200  003235'                  DOERR:  MSSG2
926 003202  177777                          -1
927 003204  003366'                  NOINTR: MSSG5
928 003206  177777                          -1
929 003210     045     104     101  MSSG1:  .ASCIZ  '%DATI OR DATIP ERROR'
```

```
        003213      124      111      040
        003216      117      122      040
        003221      104      101      124
        003224      111      120      040
        003227      105      122      122
        003232      117      122      000
930     003235      045      104      101  MSSG2:  .ASCIZ  '%DATO OR DATOB ERROR'
        003240      124      117      040
        003243      117      122      040
        003246      104      101      124
        003251      117      102      040
        003254      105      122      122
        003257      117      122      000
931     003262      045      104      105  MSSG3:  .ASCIZ  '%DEVICE READY BIT NOT SET'
        003265      126      111      103
        003270      105      040      122
        003273      105      101      104
        003276      131      040      102
        003301      111      124      040
        003304      116      117      124
        003307      040      123      105
        003312      124      000
932     003314      045      104      105  MSSG4:  .ASCIZ  '%DEVICE INTERRUPTED ON ERROR--NOT ON DONE'
        003317      126      111      103
        003322      105      040      111
        003325      116      124      105
        003330      122      122      125
        003333      120      124      105
        003336      104      040      117
        003341      116      040      105
        003344      122      122      117
        003347      122      055      055
        003352      116      117      124
        003355      040      117      116
        003360      040      104      117
        003363      116      105      000
933     003366      045      124      110  MSSG5:  .ASCIZ  '%THE FOLLOWING DEVICE(S) DID NOT INTERRUPT:%'
        003371      105      040      106
        003374      117      114      114
        003377      117      127      111
        003402      116      107      040
        003405      104      105      126
        003410      111      103      105
        003413      050      123      051
        003416      040      104      111
        003421      104      040      116
        003424      117      124      040
        003427      111      116      124
        003432      105      122      122
        003435      125      120      124
        003440      072      045      000
934
935     003444                            WRTOUT: .BLKW   361
936     004406                            REEDIN: .BLKW   361
937
938             005350'                   .=.
939
```

```
    940         000001                      .END
```

SYMBOL TABLE

```
ACSR     000102R    CR2      000370R    ISR1     003024R    PRHMS$= 000002    SAVR1    000272R
ADDR     000006R    CSRA     000100R    ISR10    003134R    PRTY   = 000000    SAVR5    000274R
ADDR22= 001000    DATACK   001132R    ISR11    003144R    PRTY0  = 000000    SBADR    000102R
APTPRE= 000200    DATCK$=  104411    ISR12    003154R    PRTY1  = 000040    SETUP    001410R
ASB      000106R    DATER$=  104404    ISR2     003034R    PRTY2  = 000100    SIMLGO   000236R
ASTAT    000104R    DB       000360R    ISR3     003044R    PRTY3  = 000140    SOFCNT   000042R
AUTO   = 000010    DEVCNT   000264R    ISR4     003054R    PRTY4  = 000200    SOFER$=  104406
AWAS     000110R    DICNT    000306R    ISR5     003064R    PRTY5  = 000240    SOFPAS   000045R
BA       000364R    DIDATA   002110R    ISR6     003074R    PRTY6  = 000300    SPOINT   000032R
BADMEM   000372R    DIERR    003174R    ISR7     003104R    PRTY7  = 000340    SPSIZ  = 000040
BADR     000374R    DOCNT    000304R    ISR8     003114R    PS     = 177776    SR1      000016R
BDDATA   002640R    DODATA   002162R    ISR9     003124R    PSW    = 177776    SR2      000020R
BEGIN    000000R    DODI     000266R    KTPRES= 000400    PTR1     000400R    SR3      000022R
BIT0   = 000001    DOERR    003200R    KTXTND= 040000    PTR2     000402R    SR4      000024R
BIT1   = 000002    DROP     001404R    MAP22$= 104416    PUSH   = 005746    START    000410R
BIT10  = 002000    DV       000240R    MASK     000242R    PUSH2  = 024646    STAT     000026R
BIT11  = 004000    DVID1    000014R    MODNAM   000000R    PWRFLG= 000002    SVDODI   000270R
BIT12  = 010000    DVMASK   000400R    MODSP    000224R    P1       000224R    SVR0     000062R
BIT13  = 020000    DVREGS   000360R    MORDEV   001170R    P2       000226R    SVR1     000064R
BIT14  = 040000    EABIT    000310R    MORE     000244R    P3       000230R    SVR2     000066R
BIT15  = 100000    EABITS   002234R    MSGN$  = 104403    P4       000232R    SVR3     000070R
BIT2   = 000004    ECCMEM=  000100    MSGS$  = 104402    QMON22= 000010    SVR4     000072R
BIT3   = 000010    ENDIT$=  104413    MSG$   = 104401    RAND$ = 104417    SVR5     000074R
BIT4   = 000020    ENDRBF   000336R    MSSG1    003210R    RANNUM   000054R    SVR6     000076R
BIT5   = 000040    ENDWBF   000342R    MSSG2    003235R    REFADR   000334R    SWAPCC   002766R
BIT6   = 000100    END$   = 104410    MSSG3    003262R    RBUF     000312R    SYSCNT   000052R
BIT7   = 000200    ERBITS   000356R    MSSG4    003314R    RBUFEA   000330R    TOCNT    000040R
BIT8   = 000400    ERRSET   003170R    MSSG5    003366R    RBUFPA   000326R    TRPDFD=  000000
BIT9   = 001000    ERRTYP   000106R    NCPUOP= 000020    RBUFSZ   000332R    USTACK=  000000
BREAK$= 104407    EXINC    001644R    NOAPTY= 000002    RBUFVA   000324R    VECTOR   000000
BRK      001070R    EXIT$  = 104400    NOINTR   003204R    RCC      000302R    WASADR   000000
BR1      000012R    GETBUF   001520R    NOSET    003164R    REABIT   002264R    WBFADR   000000
BR2      000013R    GETDEV   002564R    NPASS    001320R    REEDIN   004406R    WBUF     000000
BTOD$  = 104421    GETOUT   002636R    NULL   = 000000    REREST   000634R    WBUFEA   000000
BUFSZ    000250R    GETPA$=  104415    NXT      001310R    RESDAT   002464R    WBUFPA   000000
BYTCC    000276R    GO       001054R    OPEN   = 000000    RESTRT   000534R    WBUFVA   000316R
B12T15   000246R    GWBUF$=  104414    OTOA$  = 104420    RES1     000056R    WCC      000300R
CAPRES= 000004    HRDCNT   000044R    PARPRE= 002000    RES2     000060R    WDFR     000116R
CC       000362R    HRDER$=  104405    PASCNT   000034R    RH70   = 001000    WDTO     000114R
CDATA$= 104412    HRDPAS   000050R    PDPF11= 000002    ROTCNT   000234R    WEABIT   002322R
CHEKPX   001742R    ICONT    000036R    PDPLSI= 020000    ROTPAT   002424R    WREGS    002016R
CKHNG$= 000001    ICOUNT   000040R    PDP44  = 100000    ROTRQS   002360R    WRTOUT   003044R
CKMORE   001436R    IDNUM    000122R    PDP60  = 004000    RQLVL    000354R    XFLAG    000000R
CLKPRE= 000001    INCCNT   001622R    PDP70  = 010000    RREGS    002070R    XFR1     000344R
CLKSP$= 104422    INCDO    001640R    PIRQ$  = 000004    RSTRT    000112R    XFR2     000346R
CLRREG   001646R    INDPAR=  000040    POPSP  = 005726    R6     =%000006    XFR3     000350R
CONFIG   000056R    INIT     000030R    POPSP2= 022626    R7     =%000007    XFR4     000352R
CR1      000366R    INTR     000120R
```

```
. ABS.   000000      000
         005350      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  13417 WORDS  ( 53 PAGES)
DYNAMIC MEMORY:  20060 WORDS  ( 77 PAGES)
ELAPSED TIME:  00:01:02
BEA,BEA/-SP=DDXCOM,BEA
```

```
DEC/X11 SYSTEM EXERS....B1
DEC/X11 SYSTEM EXERS....C1
DEC/X11 SYSTEM EXERS....D1
DEC/X11 SYSTEM EXERS....E1
DEC/X11 SYSTEM EXERS....F1
DEC/X11 SYSTEM EXERS....G1
DEC/X11 SYSTEM EXERS....H1
DEC/X11 SYSTEM EXERS....I1
DEC/X11 SYSTEM EXERS....J1
DEC/X11 SYSTEM EXERS....K1
DEC/X11 SYSTEM EXERS....L1
DEC/X11 SYSTEM EXERS....M1
DEC/X11 SYSTEM EXERS....N1


DEC/X11 SYSTEM EXERS....B2
DEC/X11 SYSTEM EXERS....C2
DEC/X11 SYSTEM EXERS....D2
DEC/X11 SYSTEM EXERS....E2
DEC/X11 SYSTEM EXERS....F2
DEC/X11 SYSTEM EXERS....G2
DEC/X11 SYSTEM EXERS....H2
DEC/X11 SYSTEM EXERS....I2
DEC/X11 SYSTEM EXERS....J2
SYMBOL TABLE           ....K2
```