PDP11

UNIBUS SYSTEMS EXERCISER

MD-11-DZKUA-C

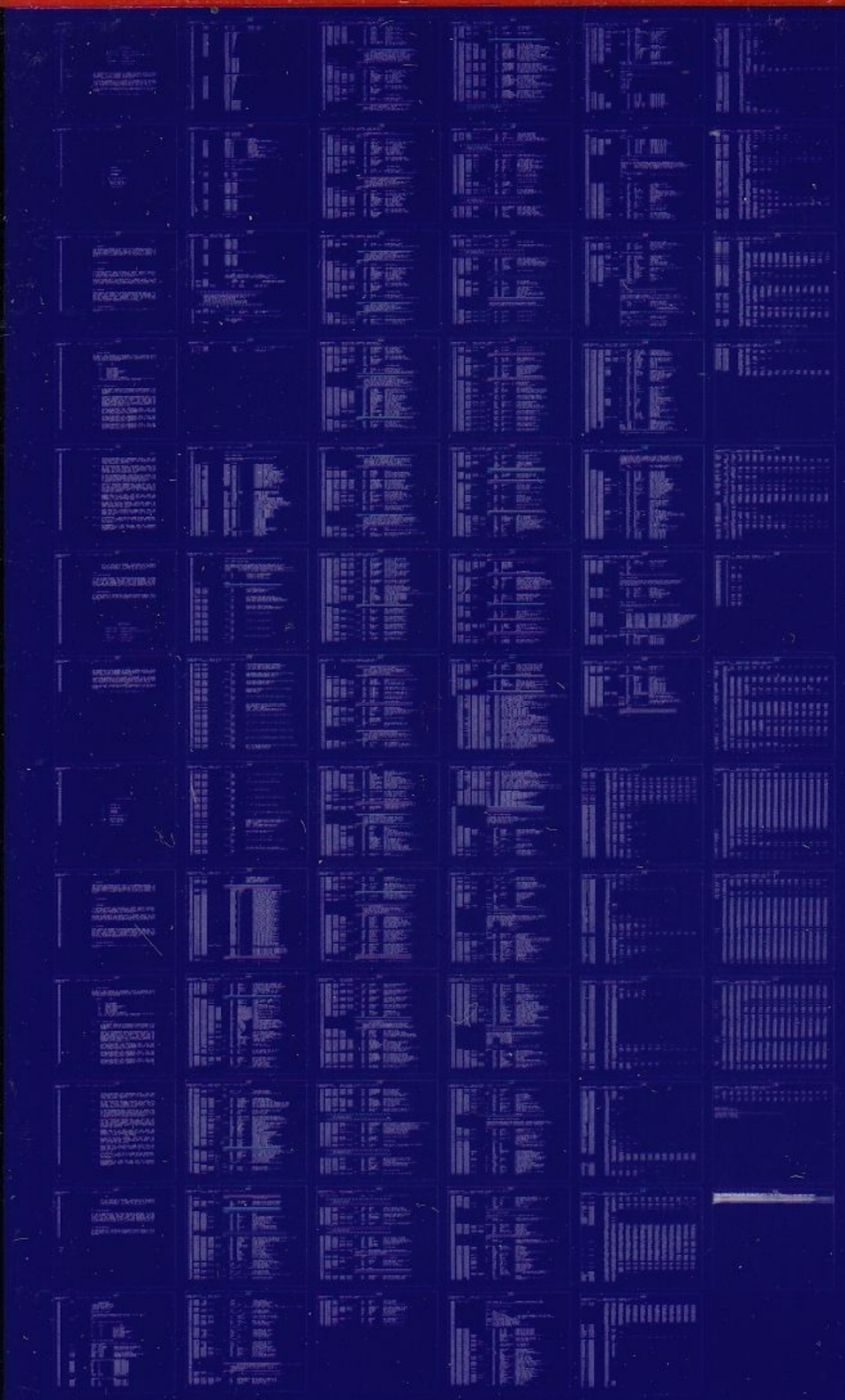
EP-DZKUA-C-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN USA



.REM %

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZKUA-C-D
PRODUCT NAME: UNIBUS SYSTEMS EXERCISER DIAGNOSTIC
DATE : NOVEMBER,75
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: MANUEL SOARES

TABLE OF CONTENTS

1.0 ABSTRACT

THIS PROGRAM WAS CREATED TO TEST PDP-11'S CPU INTERFACE CIRCUITRY. IT USES THE UNIBUS EXERCISER(S) (UBE) TO INSURE PROPER OPERATION BY SIMULATING PERIPHERALS WHICH WOULD REQUIRE THE 11-CPU TO PRODUCE THE NECESSARY SIGNALS. IT SHOULD BE NOTED THAT THE UBE IS A POWERFUL TOOL AND IF IT IS NOT PROGRAMMED CORRECTLY COULD CAUSE VARIOUS PROBLEMS ON THE BUS.

2.0 REQUIREMENTS

2.1 HARDWARE

THIS PROGRAM ASSUMES THE FOLLOWING IN PROPER WORKING CONDITION: 1. THE UNIBUS, 2. MEMORY (8K MINIMUM), AND 3. UBE(S) (4 MAXIMUM). IF A FOURTH UBE IS BEING USED, ITS TIME DELAY SHOULD BE SET AT 100US TO PREVENT LATENCY PROBLEMS IN ONE OF THE TESTS.

WITH TWO OR MORE UBE(S), ALL SHOULD HAVE W1 JUMPERS EXCEPT THE ONE FURTHEST ELECTRICALLY FROM THE CPU. IF THERE ARE MORE THAN 4 UBE(S) ON THE UNIBUS THE PROGRAM IS NOT RESPONSIBLE FOR ANY PROBLEMS WHICH MIGHT OCCUR , SINCE IT IS PROGRAMMED TO HANDLE A MAXIMUM OF ONLY 4.

2.2 SOFTWARE

AFTER LOADING THE PROGRAM THE STARTING ADDRESS MUST BE 200, SO THAT THE FIRST TIME THROUGH, THE AVAILABLE UBE(S) ARE DETERMINED. IN ADDITION IF ONE OR MORE UBE(S) ARE ADDED OR REMOVED, THE PROGRAM AGAIN MUST BE STARTED AT 200. OTHERWISE, TO AVOID DUPLICATING SOME PRINTOUTS, THE PROGRAM CAN BE RESTARTED AT ADDRESS 220. A SOFTWARE HALT CAN BE CAUSED BY DEPRESSING "↑H" ON THE CONSOLE.

3.0 PROGRAM DESCRIPTION

THIS PROGRAM WAS ASSEMBLED WITH MACY11 USING PDP-11 MAINDEC SYSMAC PACKAGE .

3.1 SWITCH OPTIONS

THE USE OF THES PROGRAM ON PROCESSORS HAVING A SOFTWARE SWITCH REGISTER NECESSITATES OPERATOR INTERACTION: THE OPERATOR MUST SET UP LOCATION 176 WITH THE SWITCH REGISTER VALUES DESIRED AND LOCATION 174 WITH THE DISPLAY VALUES DESIRED.

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST IN SWR<7:0>

NOTE: IF YOU WISH TO INHIBIT ALL TYPING EXCEPT "END OF PASS" YOU MUST PUT DOWN SWITCH 7, AFTER LOADING 200.

3.2 TEST 1 THROUGH TEST 16

TEST 1 - NO BUS GRANTS ISSUED WITH PROCESSOR AT HIGHER PRIORITY THAN BUS REQUEST. THIS TEST IS TO INSURE THAT ANY REQUEST IS NOT HONORED AS LONG AS THE PROCESSOR IS AT THE SAME OR HIGHER PRIORITY.

TEST 2 - ISSUING OF NON-PROCESSOR GRANTS AND ARBITRATION TESTS. THIS TEST WILL REQUEST ON NPR THROUGH BR4 LEVELS WITH THE PROCESSOR STATUS INITIALLY AT LEVEL 7 AND MAKE SURE THE DEVICE EXERCISES AN NPG TO DO A FUN 1-DATI, THEN THE REQUESTS WILL BE REPEATED WHILE SEQUENTIALLY LOWERING THE PROCESSOR STATUS FROM 7 TO 0 TO ALLOW ARBITRATION OF ALL REQUESTS AND THE ISSUING OF NPG.

TEST 3 - ISSUING OF BUS GRANT 7 AND ARBITRATION TESTS. THIS TEST WILL ARBITRATE FOR A BG7. THE REQUESTS WILL BE ON LEVELS BR7 THRU BR4, DOING FUN 1-DATI TRANSFERS, AND THE PROCESSOR STATUS LOWERED SEQUENTIALLY FROM 7 TO 0.

TEST 4 - ISSUING OF BUS GRANT 6 AND ARBITRATION TESTS. THIS TEST WILL ARBITRATE FOR A BG6, THE REQUESTS WILL BE ON LEVELS BR6 THRU BR4, DOING FUN 1-DATI TRANSFERS, AND THE PROCESSOR STATUS LOWERED SEQUENTIALLY FROM 6 TO 0.

TEST 5 - ISSUING OF BUS GRANT 5 AND ARBITRATION TESTS. THIS TEST WILL ARBITRATE FOR A BG5, THE REQUESTS WILL BE ON LEVELS BR5 THRU BR4, DOING FUN 1-DATI TRANSFERS, AND THE PROCESSOR STATUS LOWERED SEQUENTIALLY FROM 5 TO 0.

TEST 6 - ISSUING OF BUS GRANT 4 AND ARBITRATION TESTS. THIS TEST WILL ARBITRATE FOR A BG4, THE REQUESTS WILL BE ON LEVEL BR4, DOING FUNC 1-DATI TRANSFERS, AND THE PROCESSOR STATUS LOWERED SEQUENTIALLY FROM 4 TO 0.

TEST 7 - CPU TEST FOR NO SACK TIME OUT. THIS TEST WILL CHECK THAT THE CPU TIMES OUT AND DROPS A GRANT IF NO SACK SIGNAL IS RECEIVED. IF THE CPU TIME OUT IS INOPERATIVE, THE BUS EXERCISER WILL TIME OUT AND SEND THE SACK SIGNAL TO PREVENT A BUS HANG AND SET AN ERROR FLAG IN CR2.

TEST 10 - CPU TEST FOR RECEIVING SACK. THIS TEST IS TO INSURE THAT THE CPU CAN RECEIVE THE SACK SIGNAL; THE TIME DELAY WILL BE SET ON DEVICE 1 AND SEVERAL DATI TRANSFERS MADE. IF THERE IS NOT BUS LATE ERROR, THE CPU RECEIVED SACK CORRECTLY. IT IS ASSUMED THAT DEV 1 TIME DELAY IS SET FOR 10US.

TEST 11 - PASSING OF GRANTS AND INTERRUPT TEST. THIS TEST WILL SET OFF ALL AVAILABLE DEVICES SIMULTANEOUSLY WHOSE ONLY FUNCTIONS WILL BE TO INTERRUPT, THE REQUESTS WILL ALL BE AT LEVEL 7 SO THAT THE DEVICE CLOSEST TO THA CPU SHOULD RECEIVE BG7 FIRST AND INTERRUPT FIRST, THE NEXT CLOSEST SHOULD INTERRUPT NEXT AND SO ON.

TEST 12 - ADDRESS LINES (14 - 17) CHECK. THIS TEST WILL CHECK BUS ADDRESS LINES 14 THRU 17 BY DOING A FUN 1-DATI-NPR TO THOSE ADDRESSES. IF THE ADDRESSES DON'T EXIST THE INTERRUPT ROUTINE WILL IGNORE ANY NO SSYN ERROR.

TEST 13 - CPU TEST FOR ACLO/DCLO SEQUENCE. THIS TEST CHECKS THE ASSERTION OF ACLO AND DCLO AND THAT THE CPU TRAPS TO THE CORRECT SERVICE ROUTINE. IF THIS PROGRAM IS RUNNING UNDER ACT11 THIS TEST WILL BE SKIPPED.

TEST 14 - PARITY ERROR TEST. THIS TEST WILL CAUSE PARITY ERROR AND CHECKS THAT THE CPU TRAPS TO THE CORRECT VECTOR. IT FIRST CHECKS THAT THE CPU CAN HANDLE A PARITY TRAP OTHERWISE IT SKIPS THIS TEST.

TEST 15 - MULTITRANSFERS I. THIS TEST WILL CAUSE ANY BUS EXERCISERS, UP TO 4, TO CREATE A LOT OF TRAFFIC ON THE BUS AND CHECK THAT THE CPU CAN HANDLE IT; ALL DEVICES ARE SET OFF SIMULTANEOUSLY.

TEST 16 - MULTITRANSFERS II. THIS TEST WILL HAVE THE AVAILABLE EXERCISERS DOING VARIOUS TRANSFERS AND/OR INTERRUPTS AT DIFFERENT REQUEST LEVELS TO FURTHER CHECK CPU HANDLING CAPABILITIES.

TEST 17 - DUMMY END OF PROGRAM. THIS PORTION OF THE PROGRAM IS JUST TO SEE IF "↑H" HAS BEEN TYPED ON THE CONSOLE TO CAUSE A PROGRAM HALT. IF THERE IS NO "↑H" THE PROGRAM CONTINUES BY JUMPING TO $EOP (END-OF-PASS ROUTINE).

3.3 SYSMAC ROUTINES

THE 'END OF PASS ROUTINE' THRU 'POWER DOWN AND UP ROUTINES', AS LISTED IN THE PROGRAM LISTING, ARE THE SYSMAC PACKAGE MACROS. THEY ARE CALLED OUT IN THE SOURCE PROGRAM, SOME WITH ARGUMENTS AND SOME WITHOUT, AND ARE EXPANDED IN THE LISTING. SOME MACROS ARE NECESSARY FOR THE OPERATION OF OTHERS, SO FOR A COMPLETE EXPLANATION OF ALL AVAILABLE SYSMAC MACROS SEE PDP-11 MAINDEC SYSMAC PACKAGE (DZQAC-B-D).

4.0 ERROR REPORTING

THE MINIMUM AMOUNT OF INFORMATION GIVEN WHEN AN ERROR OCCURS IS THE PC OF THE ERROR CALL AND THE TEST NUMBER IN WHICH IT OCCURRED. OTHER PERTINENT DATA WILL BY TYPED OUT DEPENDING ON THE TEST BEING RUN AT THAT TIME.

```
        %
.REM    ?
```

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZKUA-C-D

PRODUCT NAME: UNIBUS SYSTEMS EXERCISER DIAGNOSTIC

DATE : DECEMBER, 75

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: MANUEL SOARES

```
290
291
292
293                     THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE  WITHOUT  NOTICE
294                     AND  SHOULD  NOT  BE  CONSTRUED  AS  A COMMITMENT BY DIGITAL EQUIPMENT
295                     CORPORATION.  DIGITAL EQUIPMENT CORPORATION ASSUMES NO  RESPONSIBILITY
296                     FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.
297
298                     THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE  PURCHASER
299                     UNDER  A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED
300                     (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR  USE  IN  SUCH
301                     SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.
302
303                     DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR
304                     RELIABILITY  OF  ITS  SOFTWARE  ON  EQUIPMENT  THAT IS NOT SUPPLIED BY
305                     DIGITAL.
306
307                          COPYRIGHT (C) 1975, DIGITAL EQUIPMENT CORPORATION
```

TABLE OF CONTENTS

1.0 ABSTRACT

THIS PROGRAM WAS CREATED TO TEST PDP-11'S CPU INTERFACE CIRCUITRY. IT USES THE UNIBUS EXERCISER(S) (UBE) TO INSURE PROPER OPERATION BY SIMULATING PERIPHERALS WHICH WOULD REQUIRE THE 11-CPU TO PRODUCE THE NECESSARY SIGNALS. IT SHOULD BE NOTED THAT THE UBE IS A POWERFUL TOOL AND IF IT IS NOT PROGRAMMED CORRECTLY COULD CAUSE VARIOUS PROBLEMS ON THE BUS.

2.0 REQUIREMENTS

2.1 HARDWARE

THIS PROGRAM ASSUMES THE FOLLOWING IN PROPER WORKING CONDITION: 1. THE UNIBUS, 2. MEMORY (8K MINIMUM), AND 3. UBE(S) (4 MAXIMUM). IF A FOURTH UBE IS BEING USED, ITS TIME DELAY SHOULD BE SET AT 100US TO PREVENT LATENCY PROBLEMS IN ONE OF THE TESTS.

WITH TWO OR MORE UBE(S), ALL SHOULD HAVE W1 JUMPERS EXCEPT THE ONE FURTHEST ELECTRICALLY FROM THE CPU. IF THERE ARE MORE THAN 4 UBE(S) ON THE UNIBUS THE PROGRAM IS NOT RESPONSIBLE FOR ANY PROBLEMS WHICH MIGHT OCCUR , SINCE IT IS PROGRAMMED TO HANDLE A MAXIMUM OF ONLY 4.

2.2 SOFTWARE

AFTER LOADING THE PROGRAM THE STARTING ADDRESS MUST BE 200, SO THAT THE FIRST TIME THROUGH, THE AVAILABLE UBE(S) ARE DETERMINED. IN ADDITION IF ONE OR MORE UBE(S) ARE ADDED OR REMOVED, THE PROGRAM AGAIN MUST BE STARTED AT 200. OTHERWISE, TO AVOID DUPLICATING SOME PRINTOUTS, THE PROGRAM CAN BE RESTARTED AT ADDRESS 220. A SOFTWARE HALT CAN BE CAUSED BY DEPRESSING "↑H" ON THE CONSOLE.

3.0 PROGRAM DESCRIPTION

THIS PROGRAM WAS ASSEMBLED WITH MACY11 USING PDP-11 MAINDEC SYSMAC PACKAGE .

3.1 SWITCH OPTIONS

THE USE OF THES PROGRAM ON PROCESSORS HAVING A SOFTWARE SWITCH REGISTER NECESSITATES OPERATOR INTERACTION: THE OPERATOR MUST SET UP LOCATION 176 WITH THE SWITCH REGISTER VALUES DESIRED AND LOCATION 174 WITH THE DISPLAY VALUES DESIRED.

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST IN SWR<7:0>

NOTE: IF YOU WISH TO INHIBIT ALL TYPING EXCEPT "END OF PASS" YOU MUST PUT DOWN SWITCH 7, AFTER LOADING 200.

3.2 TEST 1 THROUGH TEST 16

TEST 1 - NO BUS GRANTS ISSUED WITH PROCESSOR AT HIGHER PRIORITY THAN BUS REQUEST. THIS TEST IS TO INSURE THAT ANY REQUEST IS NOT HONORED AS LONG AS THE PROCESSOR IS AT THE SAME OR HIGHER PRIORITY.

TEST 2 - ISSUING OF NON-PROCESSOR GRANTS AND ARBITRATION TESTS. THIS TEST WILL REQUEST ON NPR THROUGH BR4 LEVELS WITH THE PROCESSOR STATUS INITIALLY AT LEVEL 7 AND MAKE SURE THE DEVICE EXERCISES AN NPG TO DO A FUN 1-DATI, THEN THE REQUESTS WILL BE REPEATED WHILE SEQUENTIALLY LOWERING THE PROCESSOR STATUS FROM 7 TO 0 TO ALLOW ARBITRATION OF ALL REQUESTS AND THE ISSUING OF NPG.

TEST 3 - ISSUING OF BUS GRANT 7 AND ARBITRATION TESTS. THIS TEST WILL ARBITRATE FOR A BG7. THE REQUESTS WILL BE ON LEVELS BR7 THRU BR4, DOING FUN 1-DATI TRANSFERS, AND THE PROCESSOR STATUS LOWERED SEQUENTIALLY FROM 7 TO 0.

TEST 4 - ISSUING OF BUS GRANT 6 AND ARBITRATION TESTS. THIS TEST WILL ARBITRATE FOR A BG6, THE REQUESTS WILL BE ON LEVELS BR6 THRU BR4, DOING FUN 1-DATI TRANSFERS, AND THE PROCESSOR STATUS LOWERED SEQUENTIALLY FROM 6 TO 0.

TEST 5 - ISSUING OF BUS GRANT 5 AND ARBITRATION TESTS. THIS TEST WILL ARBITRATE FOR A BG5, THE REQUESTS WILL BE ON LEVELS BR5 THRU BR4, DOING FUN 1-DATI TRANSFERS, AND THE PROCESSOR STATUS LOWERED SEQUENTIALLY FROM 5 TO 0.

```
                        TEST  6 - ISSUING OF BUS GRANT 4 AND  ARBITRATION  TESTS.   THIS  TEST
                                    WILL ARBITRATE FOR A BG4, THE REQUESTS WILL BE ON LEVEL BR4,
                                    DOING  FUNC  1-DATI  TRANSFERS,  AND  THE  PROCESSOR  STATUS
                                    LOWERED SEQUENTIALLY FROM 4 TO 0.

                        TEST  7 - CPU TEST FOR NO SACK TIME OUT.  THIS TEST  WILL  CHECK  THAT
                                    THE  CPU  TIMES  OUT  AND DROPS A GRANT IF NO SACK SIGNAL IS
                                    RECEIVED.  IF THE CPU  TIME  OUT  IS  INOPERATIVE,  THE  BUS
                                    EXERCISER  WILL TIME OUT AND SEND THE SACK SIGNAL TO PREVENT
                                    A BUS HANG AND SET AN ERROR FLAG IN CR2.

                        TEST 10 - CPU TEST FOR RECEIVING SACK.  THIS TEST IS  TO  INSURE  THAT
                                    THE CPU CAN RECEIVE THE SACK SIGNAL;  THE TIME DELAY WILL BE
                                    SET ON DEVICE 1 AND SEVERAL DATI TRANSFERS MADE.   IF  THERE
                                    IS  NOT BUS LATE ERROR, THE CPU RECEIVED SACK CORRECTLY.  IT
                                    IS ASSUMED THAT DEV 1 TIME DELAY IS SET FOR 10US.

                        TEST 11 - PASSING OF GRANTS AND INTERRUPT TEST.  THIS  TEST  WILL  SET
                                    OFF   ALL   AVAILABLE   DEVICES  SIMULTANEOUSLY  WHOSE  ONLY
                                    FUNCTIONS WILL BE TO INTERRUPT, THE REQUESTS WILL ALL BE  AT
                                    LEVEL 7 SO THAT THE DEVICE CLOSEST TO THA CPU SHOULD RECEIVE
                                    BG7 FIRST AND  INTERRUPT  FIRST,  THE  NEXT  CLOSEST  SHOULD
                                    INTERRUPT NEXT AND SO ON.

                        TEST 12 - ADDRESS LINES (14 - 17) CHECK.  THIS  TEST  WILL  CHECK  BUS
                                    ADDRESS  LINES 14 THRU 17 BY DOING A FUN 1-DATI-NPR TO THOSE
                                    ADDRESSES.  IF  THE  ADDRESSES  DON'T  EXIST  THE  INTERRUPT
                                    ROUTINE WILL IGNORE ANY NO SSYN ERROR.

                        TEST 13 - CPU TEST FOR  ACLO/DCLO  SEQUENCE.   THIS  TEST  CHECKS  THE
                                    ASSERTION  OF  ACLO  AND  DCLO AND THAT THE CPU TRAPS TO THE
                                    CORRECT SERVICE ROUTINE.  IF THIS PROGRAM IS  RUNNING  UNDER
                                    ACT11 THIS TEST WILL BE SKIPPED.

                        TEST 14 - PARITY ERROR TEST.  THIS TEST WILL CAUSE  PARITY  ERROR  AND
                                    CHECKS  THAT  THE CPU TRAPS TO THE CORRECT VECTOR.  IT FIRST
                                    CHECKS THAT THE CPU CAN HANDLE A PARITY  TRAP  OTHERWISE  IT
                                    SKIPS THIS TEST.

                        TEST 15 - MULTITRANSFERS I.  THIS TEST WILL CAUSE ANY BUS  EXERCISERS,
                                    UP  TO  4,  TO  CREATE A LOT OF TRAFFIC ON THE BUS AND CHECK
                                    THAT THE CPU  CAN  HANDLE  IT;   ALL  DEVICES  ARE  SET  OFF
                                    SIMULTANEOUSLY.

                        TEST 16 - MULTITRANSFERS  II.  THIS  TEST  WILL  HAVE  THE   AVAILABLE
                                    EXERCISERS  DOING  VARIOUS  TRANSFERS  AND/OR  INTERRUPTS AT
                                    DIFFERENT REQUEST  LEVELS  TO  FURTHER  CHECK  CPU  HANDLING
                                    CAPABILITIES.
```

```
492
493
494                         TEST 17 - DUMMY END OF PROGRAM.  THIS PORTION OF THE PROGRAM
495                         IS  JUST  TO  SEE  IF  "↑H" HAS BEEN TYPED ON THE CONSOLE TO
496                         CAUSE A PROGRAM HALT.  IF  THERE  IS  NO  "↑H"  THE  PROGRAM
497                         CONTINUES BY JUMPING TO $EOP (END-OF-PASS ROUTINE).
498
499
500
501                 3.3  SYSMAC ROUTINES
502
503                 THE 'END OF PASS ROUTINE' THRU 'POWER DOWN AND UP ROUTINES', AS LISTED
504                 IN  THE  PROGRAM  LISTING,  ARE  THE  SYSMAC PACKAGE MACROS.  THEY ARE
505                 CALLED OUT IN  THE  SOURCE  PROGRAM,  SOME  WITH  ARGUMENTS  AND  SOME
506                 WITHOUT.  AND  ARE EXPANDED IN THE LISTING.  SOME MACROS ARE NECESSARY
507                 FOR THE OPERATION OF OTHERS, SO FOR  A  COMPLETE  EXPLANATION  OF  ALL
508                 AVAILABLE SYSMAC MACROS SEE PDP-11 MAINDEC SYSMAC PACKAGE (DZQAC-B-D).
509
510
511
512                 4.0  ERROR REPORTING
513
514                 THE MINIMUM AMOUNT OF INFORMATION GIVEN WHEN AN ERROR OCCURS IS THE PC
515                 OF  THE  ERROR  CALL  AND THE TEST NUMBER IN WHICH IT OCCURRED.  OTHER
516                 PERTINENT DATA WILL BY TYPED OUT DEPENDING ON THE TEST  BEING  RUN  AT
517                 THAT TIME.
518                 ?
519
```

```
                                $SWR=167400
                        .TITLE  UNIBUS EXERCISER
                        ;*COPYRIGHT (C) MARCH, 75
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY M.SOARES
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-B1),AUG 29,1975.
                        ;*
000001                  $TN=1

                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        11            INHIBIT ITERATIONS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            LOOP ON TEST IN SWR<7:0>

                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                  STACK=  1100
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0=     %0              ;;GENERAL REGISTER
000001                  R1=     %1              ;;GENERAL REGISTER
000002                  R2=     %2              ;;GENERAL REGISTER
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5=     %5              ;;GENERAL REGISTER
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
                        .EQUIV  R6,SP           ;;STACK POINTER
                        .EQUIV  R7,PC           ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
```

```
                        000240          PR5=    240             ;;PRIORITY LEVEL 5
                        000300          PR6=    300             ;;PRIORITY LEVEL 6
                        000340          PR7=    340             ;;PRIORITY LEVEL 7


                                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                        100000          SW15=   100000
                        040000          SW14=   40000
                        020000          SW13=   20000
                        010000          SW12=   10000
                        004000          SW11=   4000
                        002000          SW10=   2000
                        001000          SW09=   1000
                        000400          SW08=   400
                        000200          SW07=   200
                        000100          SW06=   100
                        000040          SW05=   40
                        000020          SW04=   20
                        000010          SW03=   10
                        000004          SW02=   4
                        000002          SW01=   2
                        000001          SW00=   1
                                        .EQUIV  SW09,SW9
                                        .EQUIV  SW08,SW8
                                        .EQUIV  SW07,SW7
                                        .EQUIV  SW06,SW6
                                        .EQUIV  SW05,SW5
                                        .EQUIV  SW04,SW4
                                        .EQUIV  SW03,SW3
                                        .EQUIV  SW02,SW2
                                        .EQUIV  SW01,SW1
                                        .EQUIV  SW00,SW0

                                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                        100000          BIT15=  100000
                        040000          BIT14=  40000
                        020000          BIT13=  20000
                        010000          BIT12=  10000
                        004000          BIT11=  4000
                        002000          BIT10=  2000
                        001000          BIT09=  1000
                        000400          BIT08=  400
                        000200          BIT07=  200
                        000100          BIT06=  100
                        000040          BIT05=  40
                        000020          BIT04=  20
                        000010          BIT03=  10
                        000004          BIT02=  4
                        000002          BIT01=  2
                        000001          BIT00=  1
                                        .EQUIV  BIT09,BIT9
                                        .EQUIV  BIT08,BIT8
                                        .EQUIV  BIT07,BIT7
                                        .EQUIV  BIT06,BIT6
                                        .EQUIV  BIT05,BIT5
                                        .EQUIV  BIT04,BIT4
                                        .EQUIV  BIT03,BIT3
```

```
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR

                        .SBTTL  MEMORY MANAGEMENT DEFINITIONS

                        ;*KT11 VECTOR ADDRESS

000250                  MMVEC=  250

                        ;*KT11 STATUS REGISTER ADDRESSES

177572                  SR0=    177572
177574                  SR1=    177574
177576                  SR2=    177576
172516                  SR3=    172516

                        ;*USER "I" PAGE DESCRIPTOR REGISTERS

177600                  UIPDR0= 177600
177602                  UIPDR1= 177602
177604                  UIPDR2= 177604
177606                  UIPDR3= 177606
177610                  UIPDR4= 177610
177612                  UIPDR5= 177612
177614                  UIPDR6= 177614
177616                  UIPDR7= 177616

                        ;*USER "I" PAGE ADDRESS REGISTERS

177640                  UIPAR0= 177640
177642                  UIPAR1= 177642
177644                  UIPAR2= 177644
177646                  UIPAR3= 177646
177650                  UIPAR4= 177650
177652                  UIPAR5= 177652
177654                  UIPAR6= 177654
177656                  UIPAR7= 177656

                        ;*KERNAL "I" PAGE DESCRIPTOR REGISTERS

172300                  KIPDR0= 172300
```

```
                172302                  KIPDR1= 172302
                172304                  KIPDR2= 172304
                172306                  KIPDR3= 172306
                172310                  KIPDR4= 172310
                172312                  KIPDR5= 172312
                172314                  KIPDR6= 172314
                172316                  KIPDR7= 172316

                                        ;*KERNAL "I" PAGE ADDRESS REGISTERS

                172340                  KIPAR0= 172340
                172342                  KIPAR1= 172342
                172344                  KIPAR2= 172344
                172346                  KIPAR3= 172346
                172350                  KIPAR4= 172350
                172352                  KIPAR5= 172352
                172354                  KIPAR6= 172354
                172356                  KIPAR7= 172356


                                        .SBTTL  TRAP CATCHER

                000000                          .=0
                                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                000174                          .=174
000174  000000                         DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                         SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                000200                  .=200
000200  005037  001174                          CLR     $TMP0           ;MAKE SURE TMP0=0
000204  000137  001756                          JMP     @#START         ;JUMP TO START


        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
                ;*WHEN LOADING THE PROGRAM FOR THE FIRST TIME, OR ANY TIME
                ;*AFTER ALTERING THE # OF EXERCISERS ON THE BUS,
                ;*YOU MUST START AT LOCATION 200 AND
                ;*RESTART AT LOCATION 220 ONLY IF YOU DO NOT WISH
                ;*TO SIZE MEMORY AND TYPE OUT DEV ADDRESSES AGAIN
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*


                000220                  .=220
000220  012737  000777  001174                MOV     #777,$TMP0      ;TMP0 IS INDICATOR FOR RESTART
000226  000137  001756                        JMP     @#START         ;JUMP TO START

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

                                        .SBTTL  ACT11 HOOKS
                                        ;HOOKS REQUIRED BY ACT11
                000232                          $SVPC=.                 ;SAVE PC
                000046                          .=46
```

```
744  000046  015626                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
745          000052                  .=52
746  000052  040000                  .WORD   40000           ;;2)SET LOC.52 TO 40000
747          000232                  .=$SVPC                 ;; RESTORE PC
748
749
```

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

                                .SBTTL  COMMON TAGS

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

                    001100                  .=1100
001100                          $CMTAG:                 ;;START OF COMMON TAGS
001100  000000      $PASS:      .WORD   0               ;;CONTAINS PASS COUNT
001102  000         $TSTNM:     .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000         $ERFLG:     .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000      $ICNT:      .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000      $LPADR:     .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000      $LPERR:     .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000      $ERTTL:     .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000         $ITEMB:     .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001         $ERMAX:     .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000      $ERRPC:     .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000      $GDADR:     .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000      $BDADR:     .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000      $GDDAT:     .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000      $BDDAT:     .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000000                  .WORD   0
001136  177570      SWR:        .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001140  177570      DISPLAY:    .WORD   DDISP           ;;ADDRESS OF DISPLAY REGISTER
001142  177560      $TKS:       177560                  ;;TTY KBD STATUS
001144  177562      $TKB:       177562                  ;;TTY KBD BUFFER
001146  177564      $TPS:       177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001150  177566      $TPB:       177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001152  000         $NULL:      .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001153  002         $FILLS:     .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001154  012         $FILLC:     .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001155  000         $TPFLG:     .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001156  000000      $REGAD:     .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REG0) WAS OBTAINED
001160  000000      $REG0:      .WORD   0               ;;CONTAINS (($REGAD)+0)
001162  000000      $REG1:      .WORD   0               ;;CONTAINS (($REGAD)+2)
001164  000000      $REG2:      .WORD   0               ;;CONTAINS (($REGAD)+4)
001166  000000      $REG3:      .WORD   0               ;;CONTAINS (($REGAD)+6)
001170  000000      $REG4:      .WORD   0               ;;CONTAINS (($REGAD)+10)
001172  000000      $REG5:      .WORD   0               ;;CONTAINS (($REGAD)+12)
001174  000000      $TMP0:      .WORD   0               ;;USER DEFINED
001176  000000      $TMP1:      .WORD   0               ;;USER DEFINED
001200  000000      $TMP2:      .WORD   0               ;;USER DEFINED
001202  000000      $TMP3:      .WORD   0               ;;USER DEFINED
001204  000000      $TMP4:      .WORD   0               ;;USER DEFINED
001206  000000      $TMP5:      .WORD   0               ;;USER DEFINED
001210  000000      $TIMES: 0                           ;;MAX. NUMBER OF ITERATIONS
001212  000000      $ESCAPE:0                           ;;ESCAPE ON ERROR ADDRESS
001214  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001220  077         $QUES:      .ASCII  /?/             ;;QUESTION MARK
001221  015         $CRLF:      .ASCII  <15>            ;;CARRIAGE RETURN
001222  000012      $LF:        .ASCIZ  <12>            ;;LINE FEED
```

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

                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:       EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                ;*      DH              ;;POINTS TO THE DATA HEADER
                                ;*      DT              ;;POINTS TO THE DATA
                                ;*      DF              ;;POINTS TO THE DATA FORMAT


001224                          $ERRTB:
                                ;*****************************************************************
                                ;*****************************************************************
                                ;ITEM 1
001224  011374                          EM1             ;CPU TRAPPED THRU LOC 4 -TIME OUT
001226  011442                          DH1             ; ADDR   $ERRPC #ERR/TST#
001230  015252                          DT1             ;$REG2,$ERRPC,$TSTNM,0
001232  000000                          0
                                ;ITEM 2
001234  011473                          EM2             ;CPU ISSUED A BUS GRANT WITH PSW = 7
                                                        ;DEV 1 SHOULD NOT HAVE BECOME BUS MASTER
001236  011606                          DH2             ;BE1DB BE1CC BE1BA BE1CR1 PSW $ERRPC #ERR/TST#
001240  015262                          DT2             ;$REG0,$REG1,$REG2,$REG3,$REG4,$ERRPC,$TSTNM,0
001242  000000                          0
                                ;ITEM  3
001244  011676                          EM3             ;CPU DID NOT ISSUE A BUS NPG
001246  011730                          DH3             ;BE1CR1 BE1CC FM-PS TO-PS $ERRPC #EPR/TST#
001250  015302                          DT3             ;$REG0,$REG1,$REG2,$REG3,$ERRPC,$TSTNM,0
001252  000000                          0
                                ;ITEM 4
001254  012011                          EM4             ;CPU DID NOT ISSUE BUS GRANT 7
001256  011730                          DH3
001260  015302                          DT3
001262  000000                          0
                                ;ITEM 5
001264  012047                          EM5             ;CPU DID NOT ISSUE BUS GRANT 6
001266  011730                          DH3
001270  015302                          DT3
001272  000000                          0
                                ;ITEM 6
001274  012105                          EM6             ;CPU DID NOT ISSUE BUS GRANT 5
001276  011730                          DH3
001300  015302                          DT3
001302  000000                          0
                                ;ITEM 7
001304  012143                          EM7             ;CPU DID NOT ISSUE BUS GRANT 4
001306  011730                          DH3
001310  015302                          DT3
001312  000000                          0
                                ;ITEM 10
```

```
 861  001314  012201                        EM10            ;ONE OR MORE DEVICES DID NOT INTERRUPT
 862  001316  012247                        DH10            ;THIS IS THE ORDER IN WHICH THEY INTERRUPTED
 863                                                        ;   1ST  2ND  3RD  4TH $ERRPC #ERR/TST#
 864  001320  015320                        DT10            ;$REG1,$REG2,$REG3,$REG4,$ERRPC,$TSTNM,0
 865  001322  000000                        0
 866                                  ;ITEM 11
 867  001324  012405                        EM11            ;BUS ADDRESS LINES (A17:A14) DID NOT FUNCTION PROPERLY
 868  001326  012473                        DH11            ;BE1CR1 BE1CR2 BE1BA $ERRPC #ERR/TST#
 869  001330  015336                        DT11            ;$REG1,$REG2,$REG3,$ERRPC,$TSTNM,0
 870  001332  000000                        0
 871                                  ;ITEM 12
 872  001334  012544                        EM12            ;CPU NO SACK TIME OUT LOGIC FAILED(TO NEGATE BUS GRANT)
 873  001336  012632                        DH12            ;BE1CR1 BE1CR2 $ERRPC #ERR/TST#
 874  001340  015352                        DT12            ;$REG0,$REG1,$ERRPC,$TSTNM,0
 875  001342  000000                        0
 876                                  ;ITEM 13
 877  001344  012673                        EM13            ;CPU DID NOT PROPERLY EXECUTE AN ACLO/DCLC SEQUENCE
 878  001346  012756                        DH13            ;$ERRPC #ERR/TST#
 879  001350  015364                        DT13            ;$ERRPC,$TSTNM,0
 880  001352  000000                        0
 881                                  ;ITEM 14
 882  001354  012777                        EM14            ;CPU DID NOT TRAP FROM BUS PARITY ERR PA/PB = 0/1
 883  001356  012756                        DH13
 884  001360  015364                        DT13
 885  001362  000000                        0
 886                                  ;ITEM 15
 887  001364  013062                        EM15            ;DEV 1 DID DATIP WITH ROL ON DATOB TO MEMORY
 888                                                        ;THE TRANSFER TO THE FOLLOWING LOC WAS INCORRECT
 889  001366  013225                        DH15            ;MEMORY  ACTUAL CORRECT
 890                                                        ; LOC    DATA   DATA  $ERRPC #ERR/TST# $ICNT #
 891  001370  015372                        DT15            ;$REG0,$REG1,$REG3,$ERRPC,$TSTNM,$ICNT,0
 892  001372  000000                        0
 893                                  ;ITEM 16
 894  001374  013337                        EM16            ;DEV 3'S DATO TO MEMORY DID NOT EQUAL PATTERN IN R3
 895  001376  013225                        DH15
 896  001400  015372                        DT15
 897  001402  000000                        0
 898                                  ;ITEM 17
 899  001404  013425                        EM17            ;DEV 4'S DATO TO MEMORY DID NOT EQUAL PATERN IN R4
 900  001406  013225                        DH15
 901  001410  015372                        DT15
 902  001412  000000                        0
 903                                  ;ITEM 20
 904  001414  013513                        EM20            ;DEV 1 DID FUN 1-NPR-DATIP;INCORRECT PATTERN IN MEMORY
 905  001416  013225                        DH15
 906  001420  015372                        DT15
 907  001422  000000                        0
 908                                  ;ITEM 21
 909  001424  013607                        EM21            ;DEV 2 DID FUN 2-NPR-DATOB;INCORRECT PATTERN IN MEMORY
 910  001426  013225                        DH15
 911  001430  015372                        DT15
 912  001432  000000                        0
 913                                  ;ITEM 22
 914  001434  013703                        EM22            ;BIT 7 OF CR2 SET-CPU DID NOT TIME OUT WITH SACK INHIBITED
 915  001436  013775                        DH22            ;DEV #  PC  $ERRPC #ERR/TST#
 916  001440  015410                        DT22            ;$TMP4,$REG5,$ERRPC,$TSTNM,0
```

```
 917  001442  000000                          0
 918                                  ;ITEM 23
 919  001444  014037                          EM23            ;BIT 11 OF CR2 SET-NO SSYN ON INTR SIGNAL
 920  001446  013775                          DH22
 921  001450  015410                          DT22
 922  001452  000000                          0
 923                                  ;ITEM 24
 924  001454  014110                          EM24            ;BIT 5 OF CR2 SET-RECEIVED WRONG GRANT
 925  001456  013775                          DH22
 926  001460  015410                          DT22
 927  001462  000000                          0
 928                                  ;ITEM 25
 929  001464  014156                          EM25            ;BIT 6 OF CR2 SET-BUS LATE
 930  001466  013775                          DH22
 931  001470  015410                          DT22
 932  001472  000000                          0
 933                                  ;ITEM 26
 934  001474  014210                          EM26            ;BIT 8 OF CR2 SET-DEV DID NOT RECEIVE SSYN
 935  001476  013775                          DH22
 936  001500  015410                          DT22
 937  001502  000000                          0
 938                                  ;ITEM 27
 939  001504  014252                          EM27            ;BIT 9 OF CR2 SET-WRONG ADDR ON BUS
 940  001506  013775                          DH22
 941  001510  015410                          DT22
 942  001512  000000                          0
 943                                  ;ITEM 30
 944  001514  014321                          EM30            ;BIT 10 OF CR2 SET-DEV RECEIVED OTHER THAN ONE GRANT
 945  001516  013775                          DH22
 946  001520  015410                          DT22
 947  001522  000000                          0
 948                                  ;ITEM31
 949  001524  014410                          EM31            ;BKGRND RTN INSTRUCTIONS OF NEGB'S WERE NOT DONE
 950                                                          ;CORRECTLY TO $REG1 DURING MULTITRANFERS II
 951  001526  014550                          DH31            ;ACTUAL CORRECT
 952                                                          ;DATA    DATA $ERRPC #ERR/TST# $ICNT #
 953  001530  015422                          DT31            ;$REG1,146463,$ERRPC.$TSTNM,$ICNT,0
 954  001532  000000                          0
 955                                  ;ITEM 32
 956  001534  014643                          EM32            ;DEV 3 DID DATI BUT HAS INCORRECT
 957                                                          ;VALUES IN DATA REGISTER
 958  001536  014550                          DH31
 959  001540  015422                          DT31
 960  001542  000000                          0
 961                                  ;ITEM 33
 962  001544  014727                          EM33            ;DEV 4 DID NOT INTR THE CORRECT # OF TIMES
 963  001546  014550                          DH31
 964  001550  015422                          DT31
 965  001552  000000                          0
 966                                  ;ITEM 34
 967  001554  015001                          EM34            ;LAST DATI XFER BY DEV 1 WAS INCORRECT-
 968                                                          ;EITHER DEV DID NOT WORK OR WRONG DATA WASSET UP
 969  001556  014550                          DH31
 970  001560  015422                          DT31
 971  001562  000000                          0
 972                                  ;ITEM 35
```

```
 973  001564  015155                          EM35            ;CPU TRAPPED THRU LOC 0 TO CATCH
 974                                                          ;IMPROPERLY LOADED VECTORS
 975  001566  011442                          DH1             ; ADDR   $ERRPC #ERR/TST#
 976  001570  015252                          DT1             ;$REG2,$ERRPC,$TSTNM,0
 977  001572  000000                          0
 978                                  ;*****************************************************************
 979                                  ;*****************************************************************
 980                                  ;*****************************************************************
 981  001574  007740                          ALLERR  :7740           ;ALL ERR BITS OF CR2
 982  001576  170014                          SIMLGO  :170014         ;ADDR TO SET OFF ALL DEVICES SIMOLTANEOUSLY
 983          000114                          PBVEC   =114            ;TRAP VEC FOR PARITY ERROR
 984          000116                          PBPSW   =116            ;PSW ADDR FOR TRAP ON PARITY ERR
 985  001600  000000                          BE1DB   :0              ;DATA REG ADDR FOR DEVICE 1
 986  001602  000000                          BE1CC   :0              ;CYCLE COUNT REG ADDR FOR DEV 1
 987  001604  000000                          BE1BA   :0              ;ADDR REG ADDR FOR DEV 1
 988  001606  000000                          BE1CR1  :0              ;CONTROL REG 1 ADDR FOR DEV 1
 989  001610  000000                          BE1CLR  :0              ;CLEAR ERRS REG ADDR FOR DEV 1
 990  001612  000000                          BE1CR2  :0              ;CONTROL REG 2 ADDR FOR DEV 1
 991  001614  000000                          BE2DB   :0              ;DATA REG ADDR FOR DEV 2
 992  001616  000000                          BE2CC   :0              ;CYCLE COUNT REG ADDR FOR DEV 2
 993  001620  000000                          BE2BA   :0              ;ADDR REG ADDR FOR DEV 2
 994  001622  000000                          BE2CR1  :0              ;CONTROL REG 1 ADDR FOR DEV 2
 995  001624  000000                          BE2CLR  :0              ;CLEAR ERRS REG ADDR FOR DEV 2
 996  001626  000000                          BE2CR2  :0              ;CONTROL REG 2 ADDR FOR DEV 2
 997  001630  000000                          BE3DB   :0              ;DATA REG ADDR FOR DEV 3
 998  001632  000000                          BE3CC   :0              ;CYCLE COUNT REG ADDR FOR DEV 3
 999  001634  000000                          BE3BA   :0              ;ADDR REG ADDR FOR DEV 3
1000  001636  000000                          BE3CR1  :0              ;CONTROL REG 1 ADDR FOR DEV 3
1001  001640  000000                          BE3CLR  :0              ;CLEAR ERRS REG ADDR FOR DEV 3
1002  001642  000000                          BE3CR2  :0              ;CONTROL REG 2 ADDR FOR DEV 3
1003  001644  000000                          BE4DB   :0              ;DATA REG ADDR FOR DEV 4
1004  001646  000000                          BE4CC   :0              ;CYCLE COUNT REG ADDR FOR DEV 4
1005  001650  000000                          BE4BA   :0              ;ADDR REG ADDR FOR DEV 4
1006  001652  000000                          BE4CR1  :0              ;CONTROL REG 1 ADDR FOR DEV 4
1007  001654  000000                          BE4CLR  :0              ;CLEAR ERRS REG ADDR FOR DEV 4
1008  001656  000000                          BE4CR2  :0              ;CONTROL REG 2 ADDR FOR DEV 4
1009  001660  000000                          BE1VEC  :0              ;TRAP VEC ADDR FOR DEV 1
1010  001662  000000                          BE1PSW  :0              ;PSW ADDR FOR TRAP THRU BE1VEC
1011  001664  000000                          BE2VEC  :0              ;TRAP VEC ADDR FOR DEV 2
1012  001666  000000                          BE2PSW  :0              ;PSW ADDR FOR TRAP THRU BE2VEC
1013  001670  000000                          BE3VEC  :0              ;TRAP VEC ADDR FOR DEV 3
1014  001672  000000                          BE3PSW  :0              ;PSW ADDR FOR TRAP THRU BE3VEC
1015  001674  000000                          BE4VEC  :0              ;TRAP VEC ADDR FOR DEV 4
1016  001676  000000                          BE4PSW  :0              ;PSW ADDR FOR TRAP THRU BE4VEC
1017  001700  000000                          DEVCNT  :0              ;CONTAINS # OF DEVICES ON BUS
1018  001702  000000  000000  000000          DEVS    :0,0,0,0        ;WILL CONTAIN ADDR(S) OF INTR'G DEVS
1019  001710  000000
1020  001712  000000                          DATA1   :0              ;MAX ADDR TO WHICH DATA XFERRED BY DEV 1
1021  001714  000000                          DATA2   :0              ;MAX ADDR TO WHICH DATA XFERRED BY DEV 2
1022  001716  000000                          DATA3   :0              ;MAX ADDR TO WHICH DATA XFERRED BY DEV 3
1023  001720  000000                          DATA4   :0              ;MAX ADDR TO WHICH DATA XFERRED BY DEV 4
1024  001722  000000                          ENDMEM  :0              ;TAG ENDING DEFINED LABELS
1025                                  ;*****************************************************************
1026                                  ;*****************************************************************
1027  001724                          CLRRTN:
1028  001724  012703  001600                  MOV     #BE1DB,R3       ;R3 IS POINTER TO BUFFER AREAS
```

```
 1029  001730  005023                    1$:     CLR     (R3)+           ;CLEAR BUFFER THEN INCREMENT ADDR
 1030  001732  022703  001722                    CMP     #ENDMEM, R3     ;IF POINTER AT LAST BUFFER, EXIT
 1031  001736  100374                            BPL     1$              ;IF PLUS, GO BACK AND CLEAR NEXT ADDR
 1032  001740  012703  001160                    MOV     #$REGD,R3       ;NOW START TO CLEAR TEMP REGISTERS
 1033  001744  005023                    2$:     CLR     (R3)+           ;CLEAR CURRENT ADDR
 1034  001746  022703  001206                    CMP     #$TMP5,R3       ;CHECK FOR LAST TEMP REG ADDR
 1035  001752  101374                            BHI     2$              ;IF NOT, CLEAR NEXT TEMP REG
 1036  001754  000207                            RTS     PC              ;EXIT
 1037                              ;*****************************************************************
 1038                              ;*****************************************************************
 1039  001756                    START:
 1040  001756  012706  001100                    MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
 1041  001762  005026                            CLR     (R6)+           ;;CLEAR MEMORY LOCATION
 1042  001764  022706  001126                    CMP     #$BDDAT,R6      ;;DONE?
 1043  001770  001374                            BNE     .-6             ;;LOOP BACK IF NO
 1044  001772  012706  001100                    MOV     #STACK,SP       ;;SETUP THE STACK POINTER
 1045  001776  012737  015646  000020            MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
 1046  002004  012737  000340  000022            MOV     #340,@#IOTVEC+2 ;;LEVEL 7
 1047  002012  012737  016116  000030            MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
 1048  002020  012737  000340  000032            MOV     #340,@#EMTVEC+2 ;;LEVEL 7
 1049  002026  012737  020104  000034            MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
 1050  002034  012737  000340  000036            MOV     #340,@#TRAPVEC+2;LEVEL 7
 1051  002042  012737  020150  000024            MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
 1052  002050  012737  000340  000026            MOV     #340,@#PWRVEC+2 ;;LEVEL 7
 1053  002056  013737  015472  015464            MOV     $ENDCT,$ECOCT   ;;SETUP END-OF-PROGRAM COUNTER
 1054  002064  005037  001210                    CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
 1055  002070  005037  001212                    CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 1056  002074  112737  000001  001115            MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
 1057  002102  012737  002102  001106            MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
 1058  002110  012737  002110  001110            MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
 1059  002116  013746  000004                    MOV     @#4,-(SP)       ;;SAVE ERROR VECTOR
 1060  002122  013746  000006                    MOV     @#6,-(SP)
 1061  002126  012737  002142  000004            MOV     #64$,4          ;;SET UP TIME OUT VECTOR
 1062  002134  005777  176776                    TST     @SWR            ;;TRY TO REFERENCE HARDWARE SWR
 1063  002140  000407                            BR      65$             ;;BRANCH IF NO TIMEOUT TRAP OCCURS
 1064  002142  012737  000176  001136  64$:     MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
 1065  002150  012737  000174  001140            MOV     #DISPREG,DISPLAY ;;POINT TO SOFTWARE DISPLAY REG
 1066  002156  022626                            CMP     (SP)+,(SP)+     ;;RESTORE STACK
 1067  002160  012637  000006          65$:     MOV     (SP)+,@#6       ;;RESTORE ERROR VECTOR
 1068  002164  012637  000004                    MOV     (SP)+,@#4
 1069  002170  032777  000200  176740            BIT     #BIT07,@SWR     ;IS SWITCH 7 UP?
 1070  002176  001402                            BEQ     3$              ;IF NOT, SKIP TYPEOUT
 1071  002200  104400  011164                    TYPE    ,QNO
 1072  002204                            3$:
 1073  002204  022737  000777  001174            CMP     #777,$TMP0      ;IS THIS RESTART FROM LOC 220?
 1074  002212  001002                            BNE     5$              ;IF NOT,SKIP THE JMP INSTR
 1075  002214  000137  003506                    JMP     @#TST1          ;ELSE JUMP TO TEST 1
 1076
 1077  002220                            5$:
 1078  002220  012737  010552  000000            MOV     #THRU0,0        ;SET UP FOR TRAP THRU LOC 0
 1079  002226  012737  000340  000002            MOV     #PR7,2          ;SET UP PSW FOR TRAP THRU 0
 1080  002234  032777  000200  176674            BIT     #BIT07,@SWR     ;IS SWITCH 7 UP?
 1081  002242  001452                            BEQ     33$             ;IF NOT, SKIP TYPEOUT
 1082  002244  104400  002252                    TYPE    ,.+4            ;;TYPE ASCIZ STRING
 1083  002250  000422                            BR      66$             ;;GET OVER THE ASCIZ
 1084                                            ;;.ASCIZ      <15><12>/IF BUS HANGS WHILE SIZING MEMORY/
```

```
1085  002316                                  66$:
1086  002316  104400  002324                          TYPE    ,.+4            ;;TYPE ASCIZ STRING
1087  002322  000422                          BR      67$             ;;GET OVER THE ASCIZ
1088                                          ;;.ASCIZ        <15><12>/IT IS DUE TO NO CPU SSYN TIME OUT/
1089  002370                                  67$:
1090  002370                                  33$:
1091  002370  004737  001724                  JSR     PC,CLRRTN       ;CLEAR BUFFER AREAS
1092  002374  004737  016600                  JSR     PC,$SIZE        ;FIND AVAILABLE MEMORY
1093  002400  012737  000340  000006          MOV     #PR7,ERRVEC+2   ;PS=7 FOR TRAP THRU LOC 4
1094  002406  012700  170000                  MOV     #170000,R0      ;SET UP POINTER FOR 1ST POSSIBLE DEV ADDR
1095  002412  012702  000510                  MOV     #510,R2         ;SET UP POINTER FOR 1ST POSSIBLE VEC ADDR
1096  002416  012701  001600                  MOV     #BE1DB,R1       ;SET UP POINTER FOR DEVICE ADDR LOCATION
1097  002422  012703  001660                  MOV     #BE1VEC,R3      ;SET UP POINTER FOR INTR ADDR LOCATION
1098  002426                                  LODDEV:
1099  002426  022700  170060                  CMP     #170060,R0      ;IS R0 > LAST POSSIBLE DEV ADDR?
1100  002432  002002                          BGE     10$             ;IF NOT,GO TO 10$
1101  002434  000137  002736                  JMP     BGIN            ;ELSE GO TO BGIN
1102  002440                                  10$:
1103  002440  012737  002544  000004          MOV     #NODEV,ERRVEC   ;SET UP TRAP VECTOR FOR TIME OUT
1104  002446  005710                          TST     (R0)            ;SEE IF ACTUAL DEVICE ADDRESS EXISTS
1105  002450  012737  002662  000004          MOV     #TYMOUT,ERRVEC  ;CHANGE TRAP VECTOR FOR ERROR CONDITION
1106  002456  005237  001700                  INC     DEVCNT          ;COUNT DEVICES
1107  002462  010021                          MOVREG: MOV     R0,(R1)+        ;MOVE ACTUAL DEVICE ADDR TO DEVICE NAME
1108  002464  010037  001172                  MOV     R0,$REG5        ;REG5 CONTAINS LAST DEVICE ADDR
1109  002470  062700  000002                  ADD     #2,R0           ;INCREMENT POINTER BY 2
1110  002474  105237  001174                  INCB    $TMP0           ;COUNT # OF REGISTERS PER DEVICE
1111  002500  122737  000005  001174          CMPB    #5,$TMP0        ;AFTER 5 REGISTERS
1112  002506  001365                          BNE     MOVREG          ;ARE RECORDED
1113  002510  105037  001174                  CLRB    $TMP0           ;CLEAR THE COUNTING REGISTER
1114  002514  062700  000004                  ADD     #4,R0           ;ADD 4 TO THE POINTER THEN
1115  002520  010021                          MOV     R0,(R1)+        ;RECORD THE LAST REGISTER ADDRESS
1116  002522  062700  000002                  ADD     #2,R0           ;INCREMENT POINTER BY 2
1117  002526                                  MOVVEC:                 ;NOW START RECORDING
1118  002526  010223                          MOV     R2,(R3)+        ;THE INTR VECTORS
1119  002530  062702  000002                  ADD     #2,R2           ;INCREMENT POINTER BY 2
1120  002534  010223                          MOV     R2,(R3)+        ;THE INTR VECTORS
1121  002536  062702  000002                  ADD     #2,R2           ;INCREMENT POINTER BY 2
1122  002542  000731                          BR      LODDEV          ;AND GO SEE IF THER'S ANOTHER DEVICE
1123                                          ;*******************************************************
1124                                          ;*******************************************************
1125  002544                                  NODEV:                  ;TIME OUT ROUTINE FOR DEVICE CHECK
1126  002544  022700  170060                  CMP     #170060,R0      ;IF ALL POSSIBLE ADDR'S HAVE NOT BEEN CHECKED
1127  002550  003035                          BGT     ADD20           ;OUT-GO BACK AND CHECK FOR MORE.
1128  002552  012716  002736                  MOV     #BGIN,(SP)      ;ELSE CHANGE STACK POINTER
1129  002556  022737  000000  001700          CMP     #0,DEVCNT       ;CHECK FOR NO EXERCISERS
1130  002564  001035                          BNE     EXNO            ;IF ONE OR MORE EXERCISERS, EXIT
1131  002566  104400  002574                  TYPE    ,.+4            ;;TYPE ASCIZ STRING
1132  002572  000423                          BR      64$             ;;GET OVER THE ASCIZ
1133                                          ;;.ASCIZ        <15><12>/THERE ARE NO EXERCISERS ON THE BUS/
1134  002642                                  64$:
1135  002642  000000                          HALT
1136  002644  062700  000020                  ADD20:  ADD     #20,R0          ;ADD 20 TO POINTER
1137  002650  062702  000004                  ADD     #4,R2           ;POINTER=NEXT DEV'S VEC LOCATIONS
1138  002654  012716  002426                  MOV     #LODDEV,(SP)    ;GO BACK TO LODDEV
1139  002660                                  EXNO:
1140  002660  000002                          RTI                     ;EXIT
```

```
1141                                    ;*****************************************************************
1142                                    ;*****************************************************************
1143  002662                            TYMOUT:                         ;TIME OUT ROUTINE
1144
1145  002662  011637  001164                    MOV     (SP),$REG2      ;THE MOVE IS FOR TYPEOUT REASONS
1146  002666  162737  000002  001164            SUB     #2,$REG2        ;SUBTRACT 2 TO FIND ACTUAL ADDR
1147  002674  104001                            ERROR   1               ;ERR MESSG FOR ILLEGAL TIME OUT
1148  002676  000002                            RTI
1149                                    ;*****************************************************************
1150                                    ;*****************************************************************
1151                                    ;*****************************************************************
1152  002700                            CLRREG:
1153  002700  005000                            CLR     R0              ;R0=0
1154  002702  005002                            CLR     R2              ;R2=0
1155  002704  012701  001600                    MOV     #BE1DB,R1       ;R1 WILL POINT TO ADDR OF 1ST DEVICE
1156  002710  005200                    1$:     INC     R0              ;R0 IS REGISTER COUNTER
1157  002712  005031                            CLR     @(R1)+          ;CLEAR CONTENTS OF REG
1158  002714  022700  000006                    CMP     #6,R0           ;IF COUNT IS NOT 6
1159  002720  001373                            BNE     1$              ;GO BACK AND INCREMENT COUNT
1160  002722  005000                            CLR     R0              ;ELSE CLEAR R0
1161  002724  005202                            INC     R2              ;ADD 1 TO R2(DEVICE COUNTER)
1162  002726  020237  001700                    CMP     R2,DEVCNT       ;SEE IF IT = PREVIOUS COUNT
1163  002732  101766                            BLOS    1$              ;IF NOT, CLEAR NEXT DEV REGS
1164  002734  000207                            RTS     PC              ;EXIT
1165                                    ;//////////////////////////////////////////////////////////////////
1166                                    ;//////////////////////////////////////////////////////////////////
1167                                    ;//////////////////////////////////////////////////////////////////
1168  002736                            BGIN:
1169  002736  012737  010332  000024            MOV     #PWRFAL,PWRVEC  ;TAKE CARE OF BIT 4(S) BEING SET RANDOMLY IN CR2(S)
1170  002744  004737  010230                    JSR     PC,STVEC        ;SET UP VEC(S) FOR RANDOM ERRS
1171  002750  004737  002700                    JSR     PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
1172  002754  005037  001174                    CLR     $TMP0           ;CLEAR TEMPORARY REG
1173  002760  005037  001160                    CLR     $REG0           ;CLEAR COUNTER
1174  002764  013737  017120  001174            MOV     $LSTBK,$TMP0    ;************************
1175  002772  032777  000200  176136            BIT     #BIT07,@SWR     ;IS SWITCH 7 UP?
1176  003000  001002                            BNE     2$              ;IF UP, GO TO 2$
1177  003002  000137  003506                    JMP     5$              ;ELSE SKIP THE TYPEOUTS
1178  003006  005237  001160            2$:     INC     $REG0           ;ADD 1 TO REG0
1179  003012  006237  001174                    ASR     $TMP0           ;SHIFT TO DIVIDE
1180  003016  022737  000005  001160            CMP     #5,$REG0        ;IS TMP0 DIVIDED BY 40(5 ASR'S)
1181  003024  001370                            BNE     2$              ;IF NOT, SHIFT AGAIN
1182  003026  005237  001174                    INC     $TMP0           ;ADD 1 FOR FUDGE FACTOR
1183  003032  104400  001221                    TYPE    ,$CRLF
1184  003036  013746  001174                    MOV     $TMP0,-(SP)     ;;SAVE $TMP0 FOR TYPEOUT
1185                                                                    ;;THIS IS THE # OF K OF MEMORY
1186  003042  104404                            TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
1187  003044  104400  003052                    TYPE    ,.+4            ;;TYPE ASCIZ STRING
1188  003050  000424                            BR      64$             ;;GET OVER THE ASCIZ
1189                                            ;;.ASCIZ      /K OF MEMORY IS AVAILABLE IN THIS SYSTEM/
1190  003122                            64$:
1191  003122  104400  003130                    TYPE    ,.+4            ;;TYPE ASCIZ STRING
1192  003126  000431                            BR      65$             ;;GET OVER THE ASCIZ
1193                                            ;;.ASCIZ      <15><12>/THE FOLLOWING # OF EXERCISERS ARE ON THE BUS:  /
1194  003212                            65$:
1195  003212  013746  001700                    MOV     DEVCNT,-(SP)    ;;SAVE DEVCNT FOR TYPEOUT
1196  003216  104402                            TYPOS                   ;;GO TYPE--OCTAL ASCII
```

```
1197  003220  001
1198  003221  000
1199  003222  104400  003230
1200  003226  000436
1201
1202  003324
1203  003324  104400  003332
1204  003330  000415
1205
1206  003364
1207  003364  005037  001174
1208  003370  012700  001600
1209  003374
1210  003374  005237  001174
1211  003400  011037  001160
1212  003404  104400  003412
1213  003410  000403
1214
1215  003420
1216  003420  013746  001174
1217  003424  104402
1218  003426  002
1219  003427  000
1220  003430  104400  003436
1221  003434  000402
1222
1223  003442
1224  003442  013746  001160
1225  003446  104402
1226  003450  006
1227  003451  000
1228  003452  062700  000014
1229  003456  023737  001174  001700
1230  003464  001343
1231  003466  104400  001221
1232  003472  022737  000004  001700
1233  003500  001002
1234  003502  104400  011267
1235  003506
1236
1237
1238
1239
1240
1241
1242
1243
1244  003506  000004
1245  003510  004737  002700
1246  003514
1247
1248  003514  012777  004730  176136
1249  003522  012777  000340  176132
1250  003530  012737  002662  000004
1251  003536  012700  000340
1252  003542  012701  002021

                .BYTE   1               ;;TYPE 1 DIGIT(S)
                .BYTE   0               ;;SUPPRESS LEADING ZEROS
                TYPE    .+4             ;;TYPE ASCIZ STRING
                BR      66$             ;;GET OVER THE ASCIZ
                ;;.ASCIZ      <15><12>/THE LOWEST ELECT. PRIORITY UBE SHOULD NOT HAVE W1 JUMPER/
66$:
                TYPE    .+4             ;;TYPE ASCIZ STRING
                BR      67$             ;;GET OVER THE ASCIZ
                ;;.ASCIZ      <15><12>/DEVICE ADDRESS(ES): /<15><12>
67$:
                CLR     $TMP0           ;CLEAR TMP0(USED AS COUNTER)
                MOV     #BE1DB,R0       ;USE R0 AS POINTER TO ADDRESSES
4$:
                INC     $TMP0           ;ADD 1 TO TMP0
                MOV     (R0),$REG0      ;MOVE FOR TYPEOUT REASONS
                TYPE    .+4             ;;TYPE ASCIZ STRING
                BR      68$             ;;GET OVER THE ASCIZ
                ;;.ASCIZ      / DEV/
68$:
                MOV     $TMP0,-(SP)     ;;SAVE $TMP0 FOR TYPEOUT
                TYPOS                   ;;GO TYPE--OCTAL ASCII
                .BYTE   2               ;;TYPE 2 DIGIT(S)
                .BYTE   0               ;;SUPPRESS LEADING ZEROS
                TYPE    .+4             ;;TYPE ASCIZ STRING
                BR      69$             ;;GET OVER THE ASCIZ
                ;;.ASCIZ      / = /
69$:
                MOV     $REG0,-(SP)     ;;SAVE $REG0 FOR TYPEOUT
                TYPOS                   ;;GO TYPE--OCTAL ASCII
                .BYTE   6               ;;TYPE 6 DIGIT(S)
                .BYTE   0               ;;SUPPRESS LEADING ZEROS
                ADD     #14,R0          ;ADD 14 FOR NEXT ADDR
                CMP     $TMP0,DEVCNT    ;SEE IF TMP0 = # OF DEVICES
                BNE     4$              ;IF NOT, GO TYPE NEXT ADDR
                TYPE    ,$CRLF          ;TYPE <CR><LF>
                CMP     #4,DEVCNT       ;SEE IF THERE ARE 4 DEVICES
                BNE     5$              ;IF NOT,SKIP THE TYPE OUT
                TYPE    ,FOR4           ;ELSE TYPE MSSG FOR 4TH DEV
5$:


;*****************************************************************
;*TEST 1         NO BUS GRANTS ISSUED WITH PROCESSOR AT HIGHER PRIORITY THAN BUS REQUEST
                ;*THIS TEST IS TO INSURE THAT ANY REQUEST IS NOT
                ;*HONORED AS LONG AS THE PROCESSOR IS AT THE SAME OR
                ;*HIGHER PRIORITY
;*****************************************************************
TST1:   SCOPE
                JSR     PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
NG:

                MOV     #ERRCHK,@BE1VEC ;SET UP DEVICE 1 INTR VECTOR
                MOV     #PR7,@BE1PSW    ;SET UP DEVICE 1 PSW VECTOR
                MOV     #TYMOUT,ERRVEC  ;SET UP TRAP THRU LOC 4(TIME OUT VEC)
                MOV     #PR7,R0         ;MOVE PS=7 TO R0
                MOV     #2021,R1        ;MOVE FUN 1-DATI-BR7 TO R1
```

```
 1253  003546  004737  010626                  JSR     PC,NOG          ;DO  NOG
 1254  003552  012700  000300                  MOV     #PR6,R0         ;MOVE PS=6 TO R0
 1255  003556  012701  002011                  MOV     #2011,R1        ;MOVE FUN 1-DATI-BR6 TO R1
 1256  003562  004737  010626                  JSR     PC,NOG          ;DO NOG
 1257  003566  012700  000240                  MOV     #PR5,R0         ;MOVE PS=5 TO R0
 1258  003572  012701  002005                  MOV     #2005,R1        ;MOVE FUN 1-DATI-BR5 TO R1
 1259  003576  004737  010626                  JSR     PC,NOG          ;DO NOG
 1260  003602  012700  000200                  MOV     #PR4,R0         ;MOVE PS=4 TO R0
 1261  003606  012701  002003                  MOV     #2003,R1        ;MOVE FUN 1-DATI-BR4 TO R1
 1262  003612  004737  010626                  JSR     PC,NOG          ;DO NOG
 1263  003616  052737  004000  175762          BIS     #BIT11,@BE1CR1  ;SET BIT 11 TO DO FUN 3
 1264  003624  052737  000040  175754          BIS     #BIT05,@BE1CR1  ;SET OFF DEV AT NPR LEVEL
 1265  003632  000240                          NOP                     ;ALLOW TIME FOR XFER
 1266
 1267                                  ;*****************************************************************
 1268                                  ;*TEST 2        ISSUING OF NON-PROCESSOR GRANTS AND ARBITRATION TESTS
 1269                                          ;*THIS TEST WILL REQUEST ON NPR THRU BR4 LEVELS
 1270                                          ;*WITH THE PROCESSOR STATUS INITIALLY AT LEVEL 7 AND MAKE
 1271                                          ;*SURE THE DEVICE EXERCISES AN NPG TO DO A FUN 1-DATI.
 1272                                          ;*THEN THE REQUESTS WILL BE REPEATED WHILE SEQUENTIALLY
 1273                                          ;*LOWERING THE PROCESSOR STATUS FROM 7 TO 0 TO ALLOW
 1274                                          ;*ARBITRATION OF ALL REQUESTS AND THE ISSUING OF NPG
 1275                                  ;*****************************************************************
 1276  003634  000004                  TST2:   SCOPE
 1277
 1278  003636                          NPRTST:
 1279  003636  012700  000340                  MOV     #PR7,R0
 1280  003642                          2$:
 1281  003642  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;MAX ERRS FOR THIS TEST OCCURRED?
 1282  003650  100451                          BMI     TST3            ;;BR IF YES TO NEXT TEST
 1283  003652  012737  000340  177776          MOV     #PR7,PSW                ;INITIAL PS
 1284  003660  012777  004730  175772          MOV     #ERRCHK,@BE1VEC ;SET UP VECTOR LOCATION
 1285  003666  012777  000340  175766          MOV     #PR7,@BE1PSW    ;SET UP DEVICE INTR PSW
 1286  003674  012777  020310  175702          MOV     #ATEND,@BE1BA   ;SET UP ADDR REG
 1287  003702  012777  177777  175672          MOV     #-1,@BE1CC      ;SET CYCLE COUNT = 1
 1288  003710  012777  002077  175670          MOV     #2077,@BE1CR1   ;LOAD #2077 FUNTIONS
 1289  003716  010037  177776                  MOV     R0,PSW          ;LOWER PROC STATUS
 1290
 1291  003722  022777  177777  175652          CMP     #-1,@BE1CC      ;SEE IF DEVICE WENT OFF
 1292  003730  001014                          BNE     5$              ;IF IT DID,SKIP ERR TYPEOUT
 1293  003732  017737  175[illegible]  001160  MOV     @BE1CR1,$REG0   ;NEXT MOVES ARE FOR TYPEOUTS
 1294  003740  017737  175636  001162          MOV     @BE1CC,$REG1
 1295  003746  012737  000340  001164          MOV     #PR7,$REG2
 1296  003754  010037  001166                  MOV     R0,$REG3
 1297  003760  104003                          ERROR   3               ;TYPE ERROR MESSG
 1298  003762                          5$:
 1299  003762  162700  000040                  SUB     #40,R0          ;LOWER PS BY 1 LEVEL
 1300  003766  020027  000000                  CMP     R0,#PR0 ;SEE IF R0 IS LESS THAN 0
 1301  003772  100323                          BPL     2$              ;IF PLUS ,GO BACK AND DO ANOTHER CYCLE
 1302
 1303                                  ;*****************************************************************
 1304                                  ;*TEST 3        ISSUING OF BUS GRANT 7 AND ARBITRATION TESTS
 1305                                          ;*THIS TEST WILL ARBITRATE FOR A BG7,
 1306                                          ;*THE REQUESTS WILL BE ON LEVELS BR7 THRU BR4, DOING
 1307                                          ;*FUN 1-DATI TRANSFERS, AND THE PROCESSOR STATUS
```

```
1309                                        ;*LOWERED SEQUENTIALLY FROM 7 TO 0.
1310                                ;*****************************************************************
1311  003774  000004                TST3:   SCOPE
1312  003776                      BR7TST:
1313  003776  012700  000300               MOV     #PR6,R0         ;2ND PS WILL = 6
1314  004002                      2$:
1315  004002  123737  001115  001103       CMPB    $ERMAX,$ERFLG   ;MAX ERRS FOR THIS TEST OCCURRED?
1316  004010  100451               .       BMI     TST4            ;;BR IF YES TO NEXT TEST
1317  004012  012737  000340  177776       MOV     #PR7,PSW                ;INITIAL PS
1318  004020  012777  004730  175632       MOV     #ERRCHK,@BE1VEC ;SET UP VECTOR LOCATION
1319  004026  012777  000340  175626       MOV     #PR7,@BE1PSW    ;SET UP DEVICE INTR PSW
1320  004034  012777  020310  175542       MOV     #ATEND,@BE1BA   ;SET UP ADDR REG
1321  004042  012777  177777  175532       MOV     #-1,@BE1CC      ;SET CYCLE COUNT = 1
1322  004050  012777  002037  175530       MOV     #2037,@BE1CR1   ;LOAD #2037 FUNTIONS
1323  004056  010037  177776               MOV     R0,PSW          ;LOWER PROC STATUS
1324
1325  004062  022777  177777  175512       CMP     #-1,@BE1CC      ;SEE IF DEVICE WENT OFF
1326  004070  001014                       BNE     5$              ;IF IT DID,SKIP ERR TYPEOUT
1327  004072  017737  175510  001160       MOV     @BE1CR1,$REG0   ;NEXT MOVES ARE FOR TYPEOUTS
1328  004100  017737  175476  001162       MOV     @BE1CC,$REG1
1329  004106  012737  000340  001164       MOV     #PR7,$REG2
1330  004114  010037  001166               MOV     R0,$REG3
1331  004120  104004                       ERROR   4               ;TYPE ERROR MESSG
1332  004122                      5$:
1333  004122  162700  000040               SUB     #40,R0          ;LOWER PS BY 1 LEVEL
1334  004126  020027  000000               CMP     R0,#PR0 ;SEE IF R0 IS LESS THAN 0
1335  004132  100323                       BPL     2$              ;IF PLUS ,GO BACK AND DO ANOTHER CYCLE
1336
1337
1338                                ;*****************************************************************
1339                                ;*TEST 4         ISSUING OF BUS GRANT 6 AND ARBITRATION TESTS
1340                                        ;*THIS TEST WILL ARBITRATE FOR A BG6.
1341                                        ;*THE REQUESTS WILL BE ON LEVELS BR6 THRU BR4, DOING
1342                                        ;*FUN 1-DATI TRANSFERS. AND THE PROCESSOR STATUS
1343                                        ;*LOWERED SEQUENTIALLY FROM 6 TO 0.
1344                                ;*****************************************************************
1345  004134  000004                TST4:   SCOPE
1346  004136                      BR6TST:
1347  004136  012700  000240               MOV     #PR5,R0         ;2ND PS WILL = 5
1348  004142                      2$:
1349  004142  123737  001115  001103       CMPB    $ERMAX,$ERFLG   ;MAX ERRS FOR THIS TEST OCCURRED?
1350  004150  100451                       BMI     TST5            ;;BR IF YES TO NEXT TEST
1351  004152  012737  000300  177776       MOV     #PR6,PSW                ;INITIAL PS
1352  004160  012777  004730  175472       MOV     #ERRCHK,@BE1VEC ;SET UP VECTOR LOCATION
1353  004166  012777  000340  175466       MOV     #PR7,@BE1PSW    ;SET UP DEVICE INTR PSW
1354  004174  012777  020310  175402       MOV     #ATEND,@BE1BA   ;SET UP ADDR REG
1355  004202  012777  177777  175372       MOV     #-1,@BE1CC      ;SET CYCLE COUNT = 1
1356  004210  012777  002017  175370       MOV     #2017,@BE1CR1   ;LOAD #2017 FUNTIONS
1357  004216  010037  177776               MOV     R0,PSW          ;LOWER PROC STATUS
1358
1359  004222  022777  177777  175352       CMP     #-1,@BE1CC      ;SEE IF DEVICE WENT OFF
1360  004230  001014                       BNE     5$              ;IF IT DID,SKIP ERR TYPEOUT
1361  004232  017737  175350  001160       MOV     @BE1CR1,$REG0   ;NEXT MOVES ARE FOR TYPEOUTS
1362  004240  017737  175336  001162       MOV     @BE1CC,$REG1
1363  004246  012737  000300  001164       MOV     #PR6,$REG2
1364  004254  010037  001166               MOV     R0,$REG3
```

```
1365  004260  104005                          ERROR   5               ;TYPE ERROR MESSG
1366  004262                          5$:
1367  004262  162700  000040                SUB     #40,R0          ;LOWER PS BY 1 LEVEL
1368  004266  020027  000000                CMP     R0,#PR0 ;SEE IF R0 IS LESS THAN 0
1369  004272  100323                        BPL     2$              ;IF PLUS ,GO BACK AND DO ANOTHER CYCLE
1370
1371
1372                                  ;*****************************************************************
1373                                  ;*TEST 5        ISSUING OF BUS GRANT 5 AND ARBITRATION TESTS
1374                                          ;*THIS TEST WILL ARBITRATE FOR A BG5.
1375                                          ;*THE REQUESTS WILL BE ON LEVELS BR5 THRU BR4, DOING
1376                                          ;*FUN 1-DATI TRANSFERS, AND THE PROCESSOR STATUS
1377                                          ;*LOWERED SEQUENTIALLY FROM 5 TO 0.
1378                                  ;*****************************************************************
1379  004274  000004                  TST5:   SCOPE
1380  004276                          BR5TST:
1381  004276  012700  000200                MOV     #PR4,R0         ;2ND PS WILL = 4
1382  004302                          2$:
1383  004302  123737  001115  001103        CMPB    $ERMAX,$ERFLG   ;MAX ERRS FOR THIS TEST OCCURRED?
1384  004310  100451                        BMI     TST6            ;;BR IF YES TO NEXT TEST
1385  004312  012737  000240  177776        MOV     #PR5,PSW                ;INITIAL PS
1386  004320  012777  004730  175332        MOV     #ERRCHK,@BE1VEC ;SET UP VECTOR LOCATION
1387  004326  012777  000340  175326        MOV     #PR7,@BE1PSW    ;SET UP DEVICE INTR PSW
1388  004334  012777  020310  175242        MOV     #ATEND,@BE1BA   ;SET UP ADDR REG
1389  004342  012777  177777  175232        MOV     #-1,@BE1CC      ;SET CYCLE COUNT = 1
1390  004350  012777  002007  175230        MOV     #2007,@BE1CR1   ;LOAD #2007 FUNTIONS
1391  004356  010037  177776                MOV     R0,PSW          ;LOWER PROC STATUS
1392
1393  004362  022777  177777  175212        CMP     #-1,@BE1CC      ;SEE IF DEVICE WENT OFF
1394  004370  001014                        BNE     5$              ;IF IT DID,SKIP ERR TYPEOUT
1395  004372  017737  175210  001160        MOV     @BE1CR1,$REG0   ;NEXT MOVES ARE FOR TYPEOUTS
1396  004400  017737  175176  001162        MOV     @BE1CC,$REG1
1397  004406  012737  000240  001164        MOV     #PR5,$REG2
1398  004414  010037  001166                MOV     R0,$REG3
1399  004420  104006                        ERROR   6               ;TYPE ERROR MESSG
1400  004422                          5$:
1401  004422  162700  000040                SUB     #40,R0          ;LOWER PS BY 1 LEVEL
1402  004426  020027  000000                CMP     R0,#PR0 ;SEE IF R0 IS LESS THAN 0
1403  004432  100323                        BPL     2$              ;IF PLUS ,GO BACK AND DO ANOTHER CYCLE
1404
1405
1406                                  ;*****************************************************************
1407                                  ;*TEST 6        ISSUING OF BUS GRANT 4 AND ARBITRATION TESTS
1408                                          ;*THIS TEST WILL ARBITRATE FOR A BG4.
1409                                          ;*THE REQUESTS WILL BE ON LEVEL BR4, DOING
1410                                          ;*FUNC 1-DATI TRANSFERS, AND THE PROCESSOR STATUS
1411                                          ;*LOWERED SEQUENTIALLY FROM 4 TO 0.
1412                                  ;*****************************************************************
1413  004434  000004                  TST6:   SCOPE
1414  004436                          BR4TST:
1415  004436  012700  000140                MOV     #PR3,R0         ;2ND PS WILL = 3
1416  004442                          2$:
1417  004442  123737  001115  001103        CMPB    $ERMAX,$ERFLG   ;MAX ERRS FOR THIS TEST OCCURRED?
1418  004450  100451                        BMI     TST7            ;;BR IF YES TO NEXT TEST
1419  004452  012737  000200  177776        MOV     #PR4,PSW                ;INITIAL PS
1420  004460  012777  004730  175172        MOV     #ERRCHK,@BE1VEC ;SET UP VECTOR LOCATION
```

```
1421  004466  012777  000340  175166          MOV     #PR7,@BE1PSW    ;SET UP DEVICE INTR PSW
1422  004474  012777  020310  175102          MOV     #ATEND,@BE1BA   ;SET UP ADDR REG
1423  004502  012777  177777  175072          MOV     #-1,@BE1CC      ;SET CYCLE COUNT = 1
1424  004510  012777  002003  175070          MOV     #2003,@BE1CR1   ;LOAD #2003 FUNTIONS
1425  004516  010037  177776                  MOV     R0,PSW          ;LOWER PROC STATUS
1426
1427  004522  022777  177777  175052          CMP     #-1,@BE1CC      ;SEE IF DEVICE WENT OFF
1428  004530  001014                          BNE     5$              ;IF IT DID,SKIP ERR TYPEOUT
1429  004532  017737  175050  001160          MOV     @BE1CR1,$REG0   ;NEXT MOVES ARE FOR TYPEOUTS
1430  004540  017737  175036  001162          MOV     @BE1CC,$REG1
1431  004546  012737  000200  001164          MOV     #PR4,$REG2
1432  004554  010037  001166                  MOV     R0,$REG3
1433  004560  104007                          ERROR   7               ;TYPE ERROR MESSG
1434  004562                          5$:
1435  004562  162700  000040                  SUB     #40,R0          ;LOWER PS BY 1 LEVEL
1436  004566  020027  000000                  CMP     R0,#PR0 :SEE IF R0 IS LESS THAN 0
1437  004572  100323                          BPL     2$              ;IF PLUS ,GO BACK AND DO ANOTHER CYCLE
1438                                  ;*****************************************************************
1439
1440                                  ;*****************************************************************
1441                                  ;*TEST 7        CPU TEST FOR NO SACK TIME OUT
1442                                          ;*THIS TEST WILL CHECK THAT THE CPU TIMES OUT AND
1443                                          ;*DROPS A GRANT IF NO SACK SIGNAL IS RECEIVED
1444                                          ;*IF THE CPU TIME OUT IS INOPERATIVE, THE BUS EXERCISER
1445                                          ;*WILL TIME OUT AND SEND THE SACK SIGNAL TO PREVENT
1446                                          ;*A BUS HANG AND SET AN ERROR FLAG IN CR2
1447                                  ;*****************************************************************
1448  004574  000004                  TST7:   SCOPE
1449  004576  004737  002700                  JSR     PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
1450  004602  012777  177777  174772          MOV     #-1,@BE1CC      ;SET CYCLE COUNT = 1
1451  004610  012777  020310  174766          MOV     #ATEND,@BE1BA   ;SET UP DEVICE REG ADDR
1452  004616  012737  000340  177776          MOV     #PR7,PSW        ;SET PS=7
1453  004624  012737  002662  000004          MOV     #TYMOUT,ERRVEC  ;SET UP TIME OUT VECTOR
1454  004632  012777  004730  175020          MOV     #ERRCHK,@BE1VEC ;SET UP DEVICE INTR VECTOR
1455  004640  012777  000340  175014          MOV     #PR7,@BE1PSW    ;SET UP DEVICE INTR PSW
1456  004646  052777  000010  174736          BIS     #BIT03,@BE1CR2  ;INHIBIT SACK RETURN
1457  004654  012777  006003  174724          MOV     #6003,@BE1CR1   ;DO FUN 3--BR4
1458  004662  012737  000140  177776          MOV     #PR3,PSW        ;LOWER PROC. STATUS TO 3
1459  004670  004737  011036                  JSR     PC,CNTR         ;DELAY FOR TIMEOUT
1460  004674  042777  000010  174710          BIC     #BIT03,@BE1CR2  ;ALLOW FUTURE SACKS
1461  004702  105777  174704                  TSTB    @BE1CR2         ;CHECK IF NO-NO SACK BIT IS SET
1462  004706  100024                          BPL     TST10           ;;IF NOT SET, GO TO NEXT TEST
1463  004710  017737  174672  001160          MOV     @BE1CR1,$REG0   ;MOVE FOR TYPEOUT REASONS
1464  004716  017737  174670  001162          MOV     @BE1CR2,$REG1   ;MOVE FOR TYPEOUT
1465  004724  104012                          ERROR   12              ;ERROR IF SET-DEVICE FORCED TO SEND SACK
1466  004726  000414                          BR      TST10           ;;GO TO NEXT TEST
1467                                  ;*****************************************************************
1468                                  ;*****************************************************************
1469  004730                          ERRCHK:
1470  004730  033777  001574  174654          BIT     ALLERR,@BE1CR2 ;CHECK FOR ANY ERRS IN CR2
1471  004736  001407                          BEQ     5$              ;IF NONE, EXIT
1472  004740  011637  001172                  MOV     (SP),$REG5      ;FOR TYPEOUT OF PC
1473  004744  012737  000001  001204          MOV     #1,$TMP4        ;INDICATOR FOR DEVICE 1
1474  004752  004737  010376                  JSR     PC,ERRTN        ;CHECK TO SEE IF ANY ERRORS OCCURED
1475  004756                          5$:
1476  004756  000002                          RTI                     ;EXIT TRAP
```

```
                                ;*****************************************************************
                                ;*TEST 10       CPU TEST FOR RECEIVING SACK
                                        ;*THIS TEST IS TO INSURE THAT THE CPU CAN RECEIVE THE
                                        ;*SACK SIGNAL; THE TIME DELAY WILL BE SET ON DEVICE 1
                                        ;*AND SEVERAL DATI TRANSFERS MADE, IF THERE IS NO BUS
                                        ;*LATE ERROR, THE CPU RECEIVED SACK CORRECTLY
                                        ;*IT IS ASSUMED THAT DEV 1 TIME DELAY IS SET FOR 10 US
                                ;*****************************************************************
004760  000004                TST10:  SCOPE

004762  012737  000340  177776          MOV     #PR7,PSW        ;PS = 7
004770  004737  002700                  JSR     PC,CLRREG       ;CLEAR ALL DEVICE REGISTERS
004774  012702  020310                  MOV     #ATEND,R2       ;R2 WILL POINT TO END OF PROG
005000  012705  000010                  MOV     #10,R5          ;R5 = # OF TEST WORDS TO CREATE
005004  004737  011020                  JSR     PC,DOUP         ;CREATE THOSE TEST WORDS

005010  012777  004730  174642          MOV     #ERRCHK,@BE1VEC ;SET UP VECTOR LOCATION
005016  012777  000340  174636          MOV     #PR7,@BE1PSW    ;SET UP DEVICE INTR PSW
005024  012777  177770  174550          MOV     #-10,@BE1CC     ;SET UP CYCLE COUNT
005032  012777  020310  174544          MOV     #ATEND,@BE1BA   ;SET UP ADDR REGISTER
005040  052777  040000  174544          BIS     #BIT14,@BE1CR2  ;SET BIT 14 OF CR2 FOR TIME DELAY
005046  012777  024441  174532          MOV     #24441,@BE1CR1  ;DO FUN 2-DATIP/NO ROL-NPR
005054  012737  000000  177776          MOV     #PR0,PSW        ;LOWER PS TO ALLOW INTERRUPTS
005062                          5$:
005062  105777  174520                  TSTB    @BE1CR1         ;SEE IF DONE BIT SET
005066  100375                          BPL     5$              ;IF NOT, GO BACK AND WAIT
005070  042777  040000  174514          BIC     #BIT14,@BE1CR2  ;ELSE CLEAR BIT 14 OF CR2
005076  022777  000010  174474          CMP     #10,@BE1DB      ;DID LAST XFER MOVE 10 INTO DB
005104  001407                          BEQ     10$             ;IF IT DID,GO TO 10$
005106  017737  174466  001162          MOV     @BE1DB,$REG1    ;ELSE MOVE FOR ERR TYPE OUT
005114  012737  000010  001164          MOV     #10,$REG2
005122  104034                          ERROR   34              ;TYPE ERR MSSG
005124                          10$:
005124  032777  004000  174460          BIT     #BIT11,@BE1CR2  ;SEE IF NO SSYN ON INTR ERR SET
005132  001402                          BEQ     TST11           ;;IF NOT SET, GO TO NEXT TEST
005134  104023                          ERROR   23              ;ELSE TYPE ERR MSSG
005136  000400                          BR      TST11           ;;THEN GO TO NEXT TEST


                                ;*****************************************************************
                                ;*TEST 11       PASSING OF GRANTS AND INTERRUPT TEST
                                        ;*THIS TEST WILL SET OFF ALL AVAILABLE DEVICES SIMULTANEOUSLY
                                        ;*WHOSE ONLY FUNCTIONS WILL BE TO INTERRUPT. THE REQUESTS
                                        ;*WILL ALL BE AT LEVEL 7 SO THAT THE DEVICE CLOSEST TO THE CPU
                                        ;*SHOULD RECEIVE BG7 FIRST AND INTERRUPT FIRST, THE NEXT
                                        ;*CLOSEST SHOULD INTERRUPT NEXT AND SO ON.
                                ;*****************************************************************
005140  000004                TST11:  SCOPE
005142  012737  000340  177776          MOV     #PR7,PSW        ;PS=7
005150  004737  002700                  JSR     PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
005154                          LOAD1:
005154  012704  001702                  MOV     #DEVS,R4        ;DEVS CONTAINS SEQUENCE OF INTR'G DEVICE ADDRS
005160  012777  005402  174472          MOV     #INTR1,@BE1VEC  ;SET UP DEVICE 1 INTR VECTOR
005166  012777  000340  174466          MOV     #PR7,@BE1PSW    ;SET UP INTR PSW
```

```
1533  005174  012777  000036  174404          MOV     #36,@BE1CR1     ;DO FUN 0 - BR7 THRU BR4
1534  005202  122737  000001  001700          CMPB    #1,DEVCNT       ;IF ONLY 1 DEVICE ON BUS
1535  005210  001443                          BEQ     GO              ;BRANCH TO GO
1536  005212  012777  005420  174444  LOAD2:  MOV     #INTR2,@BE2VEC  ;SET UP DEVICE 2 INTR VECTOR
1537  005220  012777  000340  174440          MOV     #PR7,@BE2PSW    ;SET UP DEVICE 2 PSW VECTOR
1538  005226  012777  000036  174366          MOV     #36,@BE2CR1     ;DO FUN 0 - BR7 THRU BR4
1539  005234  122737  000002  001700          CMPB    #2,DEVCNT       ;IF ONLY 2 DEVICES ON BUS
1540  005242  001426                          BEQ     GO              ;BRANCH TO GO
1541  005244  012777  005436  174416  LOAD3:  MOV     #INTR3,@BE3VEC  ;SET UP DEVICE 3 INTR VECTOR
1542  005252  012777  000340  174412          MOV     #PR7,@BE3PSW    ;SET UP DEVICE 3 PSW VECTOR
1543  005260  012777  000036  174350          MOV     #36,@BE3CR1     ;DO FUN 0 - BR7 THRU BR4
1544  005266  122737  000003  001700          CMPB    #3,DEVCNT       ;IF ONLY 3 DEVICES ON BUS
1545  005274  001411                          BEQ     GO              ;BRANCH TO GO
1546  005276                          LOAD4:
1547  005276  012777  005454  174370          MOV     #INTR4,@BE4VEC  ;SET UP DEVICE 4 INTR VECTOR
1548  005304  012777  000340  174364          MOV     #PR7,@BE4PSW    ;SET UP DEVICE 4 PSW VECTOR
1549  005312  012777  000036  174332          MOV     #36,@BE4CR1     ;DO FUN 0 - BR7 THRU BR4
1550  005320                          GO:
1551  005320  005001                          CLR     R1              ;CLEAR R1 FOR COUNTING
1552  005322  005277  174250                  INC     @SIMLGO         ;SET SIMULTANEOUS GO REGISTER
1553  005326  012737  000000  177776          MOV     #PR0,PSW        ;LOWER PS TO ALLOW INTERRUPTS
1554  005334  004737  011036                  JSR     PC,CNTR         ;ALLOW TIME FOR INTERRUPTS BY COUNTING
1555  005340  020137  001700          CMPARE: CMP     R1,DEVCNT       ;COMPARE THE TWO
1556  005344  001456                          BEQ     TST12           ;;IF BUFFERS INCREMENTED IN CORRECT SEQUENCE, GO TO NEXT
1557  005346  013737  001702  001162          MOV     DEVS,$REG1      ;MOVE FOR TYPEOUT REASONS
1558  005354  013737  001704  001164          MOV     DEVS+2,$REG2    ;MOVE FOR TYPEOUT REASONS
1559  005362  013737  001706  001166          MOV     DEVS+4,$REG3    ;MOVE FOR TYPEOUT REASONS
1560  005370  013737  001710  001170          MOV     DEVS+6,$REG4    ;MOVE FOR TYPEOUT REASONS
1561  005376  104010                          ERROR   10              ;TYPE ERR MSSG
1562  005400  000440                          BR      TST12           ;;GO TO NEXT TEST
1563                          ;*****************************************************************
1564                          ;*****************************************************************
1565  005402                  INTR1:
1566  005402  005201                          INC     R1              ;ADD 1 TO COUNTER ON INTR
1567  005404  013724  001600                  MOV     BE1DB,(R4)+     ;MOVE ADDR FOR TYPEOUT
1568  005410  012737  000001  001204          MOV     #1,$TMP4        ;INDICATOR FOR DEVICE 1
1569  005416  000424                          BR      INTRTN          ;BRANCH TO REST OF INTR RTN
1570  005420                  INTR2:
1571  005420  005201                          INC     R1              ;ADD 1 TO COUNTER ON INTR
1572  005422  013724  001614                  MOV     BE2DB,(R4)+     ;MOVE ADDR FOR TYPEOUT
1573  005426  012737  000002  001204          MOV     #2,$TMP4        ;INDICATOR FOR DEVICE 2
1574  005434  000415                          BR      INTRTN          ;BRANCH TO REST OF INTR RTN
1575  005436                  INTR3:
1576  005436  005201                          INC     R1              ;ADD 1 TO COUNTER ON INTR
1577  005440  013724  001630                  MOV     BE3DB,(R4)+     ;MOVE ADDR FOR TYPEOUT
1578  005444  012737  000003  001204          MOV     #3,$TMP4        ;INDICATOR FOR DEVICE 3
1579  005452  000406                          BR      INTRTN          ;BRANCH TO REST OF INTR RTN
1580  005454                  INTR4:
1581  005454  005201                          INC     R1              ;ADD 1 TO COUNTER ON INTR
1582  005456  013724  001644                  MOV     BE4DB,(R4)+     ;MOVE ADDR FOR TYPEOUT
1583  005462  012737  000004  001204          MOV     #4,$TMP4        ;INDICATOR FOR DEVICE 4
1584  005470                  INTRTN:
1585  005470  011637  001172                  MOV     (SP),$REG5      ;FOR TYPEOUT OF PC
1586  005474  004737  010376                  JSR     PC,ERRTN        ;SEE IF ERROR CAUSED INTR
1587  005500  000002                          RTI                     ;EXIT
1588
```

```
                                        ;*****************************************************************
                                        ;*TEST 12       ADDRESS LINES (14 - 17) CHECK
                                                ;*THIS TEST WILL CHECK BUS ADDRESS LINES 14 THRU 17
                                                ;*BY DOING A FUN 1-DATI-NPR TO THOSE ADDRESSES
                                                ;*IF THE ADDRESSES DON'T EXIST THE INTERRUPT ROUTINE
                                                ;*WILL IGNORE ANY NO SSYN ERROR.
                                        ;*****************************************************************
005502  000004                          TST12:  SCOPE

005504  004737  002700                          JSR     PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
005510  012737  000140  177776                  MOV     #PR3,PSW        ;PS=3
005516  012777  005604  174134                  MOV     #BRK,@BE1VEC    ;SET UP DEVICE INTR VEC
005524  012777  000340  174130                  MOV     #PR7,@BE1PSW    ;SET UP DEVICE PSW VEC
005532                                  D014:
005532  004737  011064                          JSR     PC,ADLI         ;TEST ADDR LINES 14 &15
005536  152777  000001  174046                  BISB    #1,@BE1CR2      ;ELSE SET BIT 0 OF CR2(ADDR LINE 16)
005544  004737  011064                          JSR     PC,ADLI
005550  042777  000001  174034                  BIC     #1,@BE1CR2      ;CLEAR BIT 0(ADDR LINE 16)
005556  152777  000002  174026                  BISB    #2,@BE1CR2      ;SET BIT 1 OF CR2(ADDR LINE 17)
005564  004737  011064                          JSR     PC,ADLI
005570  152777  000003  174014                  BISB    #3,@BE1CR2      ;ELSE SET BITS 0 AND 1 OF CR2
                                                                        ;SETS ADDR LINES 16 & 17
005576  004737  011064                          JSR     PC,ADLI
005602  000431                                  BR      TST13           ;;GO TO NEXT TEST
                                        ;*****************************************************************
                                        ;*****************************************************************
005604                                  BRK:
005604  011637  001172                          MOV     (SP),$REGS      ;FOR TYPEOUT OF PC
005610  012737  000001  001204                  MOV     #1,$TMP4        ;INDICATOR FOR DEVICE 1
005616  032777  007340  173766                  BIT     #7340,@BE1CR2   ;CHECK FOR ALL ERRS EXCEPT NO SSYN ERR
005624  001003                                  BNE     1$              ;IF ANY ARE SET,SEE WHICH ONES
005626  005077  173756                          CLR     @BE1CLR         ;ELSE CLEAR THE NO SSYN ERR
005632  000414                                  BR      EXBRK           ;AND EXIT
005634                                  1$:
005634  017737  173746  001162                  MOV     @BE1CR1,$REG1   ;MOVES ARE FOR TYPEOUTS
005642  017737  173744  001164                  MOV     @BE1CR2,$REG2
005650  017737  173730  001166                  MOV     @BE1BA,$REG3
005656  104011                                  ERROR   11              ;ERR ON ACCESSING A14 - A17
005660  004737  010376                          JSR     PC,ERRTN        ;DO ERR CHECK SUB-ROUTINE
005664  000002                          EXBRK:  RTI                     ;EXIT


                                        ;*****************************************************************
                                        ;*TEST 13       CPU TEST FOR ACLO/DCLO SEQUENCE
                                                ;*THIS TEST CHECKS THE ASSERTION OF ACLO AND DCLO
                                                ;*AND THAT THE CPU TRAPS TO THE CORRECT SERVICE ROUTINE.
                                                ;*IF THIS PROGRAM IS RUNNING UNDER ACT11 THIS TEST
                                                ;*WILL BE SKIPPED.
                                        ;*****************************************************************
005666  000004                          TST13:  SCOPE
005670  012737  000001  001210                  MOV     #1,$TIMES       ;;DO 1 ITERATION
005676  005737  000042                          TST     42              ;SEE IF PROGRAM IS UNDER ACT11
005702  001061                                  BNE     TST14           ;;IF UNDER ACT, DO NOT PERFORM THIS TEST
005704  012705  000001                          MOV     #1,R5           ;INIT R5 WITH A VALUE OF 1
005710                                  6$:
```

```
1645  005710  005205                          INC     R5              ;ADD 1 TO R5
1646  005712  100376                          BPL     6$              ;KEEP ADDING AS LONG AS R5 POS
1647  005714  012737  000001  001204          MOV     #1,$TMP4        ;INDICATOR FOR DEVICE 1
1648  005722  012737  000340  177776          MOV     #PR7,PSW        ;SET PS=7
1649  005730  012777  004730  173722          MOV     #ERRCHK,@BE1VEC ;SET UP INTR VECTOR
1650  005736  012777  000340  173716          MOV     #PR7,@BE1PSW    ;SET UP DEVICE INTR PSW
1651  005744  005037  001176                  CLR     $TMP1           ;CLEAR TEMPORARY REGISTER(TMP1)
1652  005750  012737  006026  000024          MOV     #TMPPWR,PWRVEC  ;SET UP SPECIAL POWER RTN
1653  005756  052777  000020  173626          BIS     #BIT04,@BE1CR2  ;INDICATE PWR FAILURE BY SETTING BIT 4
1654  005764  004737  011036                  JSR     PC,CNTR         ;PAUSE FOR TIME
1655  005770  012737  010332  000024          MOV     #PWRFAL,PWRVEC  ;RESTORE PWRFAL SEQ FOR A PWR FAIL
1656  005776  042777  000020  173606          BIC     #BIT04,@BE1CR2  ;MAKE SURE BIT 4 IS CLEARED
1657  006004                          FAILCK:
1658  006004  022737  001221  020266          CMP     #$CRLF,$PWRMG   ;IF THIS TEST IS CAUSE OF
1659                                                                  ;PWR FAIL --TYPE <CR><LF>
1660  006012  001401                          BEQ     XTST            ;IF EQUAL, EXIT TEST
1661  006014  104013                          ERROR   13              ;TYPE ERR'MSSG IF FAILURE
1662  006016                          XTST:
1663  006016  012737  020300  020266          MOV     #$POWER,$PWRMG  ;RESTORE TYPE OUT OF 'POWER'
1664  006024  000410                          BR      TST14           ;;GO TO NEXT TEST
1665                          ;*****************************************************************
1666                          ;*****************************************************************
1667  006026                  TMPPWR:                                 ;SPECIAL PWR RTN; OTHER THAN SYSMAC'S
1668  006026  012737  001221  020266          MOV     #$CRLF,$PWRMG   ;CHANGE PWR MSSG TO <CR><LF>
1669  006034  042777  000020  173550          BIC     #BIT04,@BE1CR2  ;CLEAR POWER DOWN/UP BIT
1670  006042  000137  020150                  JMP     $PWRDN          ;GO TO THAT RTN
1671
1672
1673                          ;*****************************************************************
1674                          ;*TEST 14       PARITY ERROR TEST
1675                                  ;*THIS TEST WILL CAUSE PARITY ERROR AND CHECKS
1676                                  ;*THAT THE CPU TRAPS TO THE CORRECT VECTOR, IT
1677                                  ;*FIRST CHECKS THAT THE CPU CAN HANDLE A PARITY
1678                                  ;*TRAP OTHERWISE IT SKIPS THIS TEST
1679                          ;*****************************************************************
1680  006046  000004          TST14:  SCOPE
1681  006050  012737  006260  000010          MOV     #NODO,10        ;SET UP RESERVED INSTR VECTOR
1682  006056  012737  000340  000012          MOV     #PR7,12         ;PSW=7
1683  006064  005037  001202                  CLR     $TMP3           ;SET $TMP3 = 0
1684  006070  000270                          SEN                     ;SET N BIT OF CC
1685  006072  006737  001202                  SXT     $TMP3           ;IF VALID INSTR, $TMP3 WILL = -1
1686  006076  005737  001202                  TST     $TMP3           ;IF INVALID, $TMP3 WILL REMAIN 0
1687  006102  100033                          BPL     NXT             ;IF CP NOT= 35,40,45,OR 70,GO TO NEXT TEST
1688  006104  012737  000140  177776          MOV     #PR3,PSW        ;PS=3
1689  006112  012777  006224  173540          MOV     #PBERR,@BE1VEC  ;SET UP DEVICE INTR
1690  006120  012737  006246  000114          MOV     #PBRTN,PBVEC    ;SET UP PARITY BIT VECTOR
1691  006126  012737  000340  000116          MOV     #340,PBPSW      ;SET UP PARITY BIT PSW
1692  006134  012777  020310  173442          MOV     #ATEND,@BE1BA   ;SET UP ADDR REG
1693  006142  012777  177777  173432          MOV     #-1,@BE1CC      ;SET UP CYCLE COUNT
1694  006150  052777  010000  173434          BIS     #BIT12,@BE1CR2  ;SET BIT 12 FOR PARITY ERROR
1695  006156  005777  173430                  TST     @BE1CR2         ;SET OFF PARITY ERR SEQUENCE
1696  006162  012777  013161  173416          MOV     #13161,@BE1CR1  ;TRY FUN 1-DATO FROM CC-NPR-INTR ON DONE(7)
1697  006170  000240                          NOP                     ;ALLOW TIME FOR ATTEMPTED XFER
1698  006172                          NXT:
1699  006172  012737  000116  000114          MOV     #PBPSW,PBVEC    ;RESTORE
1700  006200  012737  000000  000116          MOV     #0,PBPSW        ;TRAP CATCHER HERE AND
```

```
1701  006206  012737  000012  000010          MOV     #12,10          ;AT RESERVED
1702  006214  012737  000000  000012          MOV     #0,12           ;INSTRUCTION VECTOR
1703  006222  000417                          BR      TST15           ;;BRANCH TO NEXT TEST IF PARITY TRAP OCCURRED
1704  006224                          PBERR:
1705  006224  011637  001172                  MOV     (SP),$REGS      ;FOR TYPEOUT OF PC
1706  006230  104014                          ERROR   14              ;TYPE ERR MSSG IF DEVICE INTERRUPTED
1707  006232  012737  000001  001204          MOV     #1,$TMP4        ;INDICATOR FOR DEVICE 1
1708  006240  004737  010376                  JSR     PC,ERRTN        ;CHECK TO SEE IF ANY ERRORS OCCURED
1709  006244  000002                          RTI                     ;EXIT TRAP
1710                                  ;***************************************************************
1711                                  ;***************************************************************
1712  006246                          PBRTN:                          ;PARITY BIT TRAP RTN
1713  006246  012777  000000  173336          MOV     #0,@BE1CR2      ;CLEAR PARITY BIT ERROR-MUST BE DONE
1714                                                                  ;BY MOVING 0(S) TO BE1CR2
1715  006254  012716  006172                  MOV     #NXT,(SP)       ;SET STACK FOR NEXT TEST
1716  006260  000002                  NODO:   RTI
1717                  .
1718
1719                                  ;***************************************************************
1720                                  ;*TEST 15       MULTITRANSFERS I
1721                                          ;*THIS TEST WILL CAUSE ANY BUS EXERCISERS, UP TO 4,
1722                                          ;*TO CREATE A LOT OF TRAFFIC ON THE BUS AND
1723                                          ;*CHECK THAT THE CPU CAN HANDLE IT; ALL DEVICES
1724                                          ;*ARE SET OFF SIMULTANEOUSLY
1725                                  ;***************************************************************
1726  006262  000004                  TST15:  SCOPE
1727  006264  004737  002700                  JSR     PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
1728  006270  012703  000000                  MOV     #0,R3           ;SET DATA PATTERN = 0
1729  006274  012704  177777                  MOV     #177777,R4      ;SET DATA PATTERN = ALL 1'S
1730  006300  004737  007470                  JSR     PC,MULT1        ;LOAD & EXECUTE ALL DEVICES
1731  006304  022737  000002  001700          CMP     #2,DEVCNT       ;ARE THERE MORE THAN 2 DEVICES?
1732  006312  100115                          BPL     TST16           ;;IF 2 OR LESS, GO TO NEXT TEST
1733  006314  012703  161610                  MOV     #161610,R3      ;ELSE LOAD R3 AND R4 WITH
1734  006320  012704  016161                  MOV     #016161,R4      ;ANOTHER PATTERN
1735  006324                          5$:
1736  006324  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;MAX ERRS FOR THIS TEST OCCURRED?
1737  006332  100505                          BMI     TST16           ;;BR IF YES TO NEXT TEST
1738  006334  004737  010570                  JSR     PC,ROTATE       ;ROTATE DATA PATTERNS
1739  006340  004737  007470                  JSR     PC,MULT1        ;LOAD & EXECUTE ALL DEVICES
1740  006344  022703  107070                  CMP     #107070,R3      ;IS R3 = 107070?
1741  006350  001365                          BNE     5$              ;IF NOT,ROTATE AND DO AGAIN
1742  006352  012703  167777                  MOV     #167777,R3      ;ELSE MOVE NEW PATTERNS
1743  006356  012704  010000                  MOV     #010000,R4      ;INTO R3 AND R4
1744  006362                          10$:
1745  006362  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;MAX ERRS FOR THIS TEST OCCURRED?
1746  006370  100466                          BMI     TST16           ;;BR IF YES TO NEXT TEST
1747  006372  004737  010570                  JSR     PC,ROTATE       ;ROTATE DATA PATTERNS
1748  006376  004737  007470                  JSR     PC,MULT1        ;LOAD & EXECUTE ALL DEVICES
1749  006402  022703  167777                  CMP     #167777,R3      ;IS R3 = 167777 AGAIN?
1750  006406  001365                          BNE     10$             ;IF NOT,ROTATE AND DO AGAIN
1751  006410  000456                          BR      TST16           ;;GO TO NEXT TEST
1752                                  ;***************************************************************
1753                                  ;***************************************************************
1754                                  ;***************************************************************
1755  006412                          $ERV1:
1756  006412  017737  173166  001712          MOV     @BE1BA,DATA1    ;MOVE ADDR IN BE1BA TO DATA1 AND
```

```
1757  006420  162737  000001  001712          SUB     #1,DATA1        ;SUB 1 TO GET ACTUAL ADDR
1758  006426  012737  000001  001204          MOV     #1,$TMP4        ;INDICATOR FOR DEVICE 1
1759  006434  000435                          BR      INK             ;BRANCH TO INK
1760  006436                          SERV2:
1761  006436  017737  173156  001714          MOV     @BE2BA,DATA2    ;MOVE ADDR IN BE2BA TO DATA2 AND
1762  006444  162737  000002  001714          SUB     #2,DATA2        ;SUB 2 TO GET ACTUAL ADDR
1763  006452  012737  000002  001204          MOV     #2,$TMP4        ;INDICATOR FOR DEVICE 2
1764  006460  000423                          BR      INK             ;BRANCH TO INK
1765  006462                          SERV3:
1766  006462  017737  173146  001716          MOV     @BE3BA,DATA3    ;MOVE ADDR IN BE3BA TO DATA3 AND
1767  006470  162737  000002  001716          SUB     #2,DATA3        ;SUB 2 TO GET ACTUAL ADDR
1768  006476  012737  000003  001204          MOV     #3,$TMP4        ;INDICATOR FOR DEVICE 3
1769  006504  000411                          BR      INK             ;BRANCH TO INK
1770  006506                          SERV4:
1771  006506  017737  173136  001720          MOV     @BE4BA,DATA4    ;MOVE ADDR IN BE4BA TO DATA4 AND
1772  006514  162737  000002  001720          SUB     #2,DATA4        ;SUB 2 TO GET ACTUAL ADDR
1773  006522  012737  000004  001204          MOV     #4,$TMP4        ;INDICATOR FOR DEVICE 4
1774  006530                          INK:
1775  006530  005237  001164                  INC     $REG2           ;INCREMENT REG
1776  006534  011637  001172                  MOV     (SP),$REG5      ;FOR TYPEOUT OF PC
1777  006540  004737  010376                  JSR     PC,ERRTN        ;CHECK FOR ANY ERRS
1778  006544  000002                          RTI                     ;EXIT
1779                          ;*****************************************************************
1780
1781                          ;*****************************************************************
1782                          ;*TEST 16       MULTITRANSFERS II
1783                                  ;*THIS TEST WILL HAVE THE AVAILABLE EXERCISERS DOING
1784                                  ;*VARIOUS TRANSFERS AND/OR INTERRUPTS AT DIFFERENT
1785                                  ;*REQUEST LEVELS TO FURTHER CHECK CPU HANDLING CAPABILITIES
1786                          ;*****************************************************************
1787  006546  000004          TST16:  SCOPE
1788  006550  012702  020310          MOV     #ATEND,R2       ;R2 = END OF PROG
1789  006554  012705  005000          MOV     #5000,R5        ;R5 = THE # OF DATA WORDS
1790  006560  004737  011020          JSR     PC,DOUP         ;CREATE THOSE WORDS IN BUFFER MEMORY
1791  006564  004737  011132          JSR     PC,TSTCVR       ;SET UP PATTERN IN MEMORY BUFFER AREA
1792  006570  004737  002700          JSR     PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
1793
1794  006574  012737  000000  177776          MOV     #PR0,PSW        ;PS=0
1795  006602  012777  007334  173050          MOV     #S1,@BE1VEC     ;SET UP DEVICE 1 INTR VECTOR
1796  006610  012777  000340  173044          MOV     #PR7,@BE1PSW    ;SET UP DEVICE 1 PSW VECTOR
1797  006616  012777  022310  172760          MOV     #ATEND+2000,@BE1BA  ;SET UP ADDR REG
1798  006624  012777  176000  172750          MOV     #-2000,@BE1CC   ;SET UP CYCLE COUNT
1799  006632  012777  015551  172746          MOV     #15551,@BE1CR1  ;DO FUN 2-DATOB FROM CC-NPR-INTR ON DONE(6)
1800  006640  022737  000001  001700          CMP     #1,DEVCNT       ;CHECK FOR MORE THAN 1 DEVICE
1801  006646  001467                          BEQ     1$              ;IF NOT, GO CHECK RESULTS
1802
1803  006650  012777  007366  173006          MOV     #S2,@BE2VEC     ;SET UP DEVICE 2 INTR VECTOR
1804  006656  012777  000340  173002          MOV     #PR7,@BE2PSW    ;SET UP DEVICE 2 PSW VECTOR
1805  006664  012777  177000  172724          MOV     #-1000,@BE2CC   ;SET UP CYCLE COUNT FOR 1000 XFERS
1806  006672  012777  020310  172720          MOV     #ATEND,@BE2BA   ;SET UP ADDR REG=1ST LOCATION AFTER PROG
1807  006700  012777  002561  172714          MOV     #2561,@BE2CR1   ;DO FUN 1-DATIP-NPR-INTR ON DONE(7)
1808  006706  022737  000002  001700          CMP     #2,DEVCNT       ;CHECK FOR MORE THAN 2 DEVICES
1809  006714  001444                          BEQ     1$              ;IF NOT, GO CHECK RESULTS
1810
1811  006716  012777  007420  172744          MOV     #S3,@BE3VEC     ;SET UP DEVICE 3 INTR VECTOR
1812  006724  012777  000340  172740          MOV     #PR7,@BE3PSW    ;SET UP PSW VECTOR
```

```
1813  006732  012777  176776  172672          MOV     #-1002,@BE3CC   ;SET UP CYCLE COUNT
1814  006740  012777  020310  172666          MOV     #ATEND,@BE3BA   ;SET UP ADDR REG
1815  006746  012777  004005  172662          MOV     #4005,@BE3CR1   ;DO FUN 2-DATI-BR5
1816  006754  022737  000003  001700          CMP     #3,DEVCNT       ;CHECK FOR MORE THAN 3 DEVICES
1817  006762  001421                          BEQ     1$              ;IF NOT, GO CHECK RESULTS
1819
1819  006764  005037  001170                  CLR     $REG4           ;USE REG4 TO COUNT INTRS
1820  006770  012777  007436  172676          MOV     #S4,@BE4VEC     ;SET UP DEVICE 4 INTR VECTOR
1821  006776  012777  000340  172672          MOV     #PR7,@BE4PSW    ;SET UP PSW VECTOR
1822  007004  012777  175000  172634          MOV     #-3000,@BE4CC   ;SET UP CYCLE COUNT
1823  007012  052777  040000  172636          BIS     #BIT14,@BE4CR2  ;SET TIME DELAY BIT OF DEVICE 4
1824  007020  012777  000021  172624          MOV     #21,@BE4CR1     ;DO FUN 0-BR7
1825  007026                          1$:
1826  007026  012700  001606                  MOV     #BE1CR1,R0      ;USE R0 TO POINT TO DEVICE 1'S CR1
1827  007032  004737  010744                  JSR     PC,$KGD         ;PERFORM BACKGROUND ROUTINE

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
                ;BE1 TRANSFER CHECKS
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1835  007036  012702  022310                  MOV     #ATEND+2000,R2  ;USE R2 TO POINT TO 2000 BYTES AFTER END OF PROGRAM
1836  007042  012737  176000  001164          MOV     #-2000,$REG2    ;SET UP $REG2 WITH EXPECTED RESULTS
1837  007050  023702  001712          10$:    CMP     DATA1,R2        ;CHECK FOR 4000 BYTES PAST END OF PROGRAM
1838  007054  001417                          BEQ     14$             ;IF EQUAL, GO CHECK FOR ANOTHER DEVICE
1839  007056  023722  001164                  CMP     $REG2,(R2)+     ;ELSE COMPARE EXPECTED RESULTS WITH THAT IN R2
1840  007062  001004                          BNE     12$             ;IF NOT EQUAL, GO TO ERR MSSG
1841  007064  062737  000002  001164          ADD     #2,$REG2        ;INCREMENT REG2 AS CC WOULD
1842  007072  000766                          BR      10$             ;AND GO BACK TO CHECK NEXT LOC
1843  007074                          12$:
1844  007074  014237  001162                  MOV     -(R2),$REG1     ;MOVE R2 TO REG 5 FOR TYPEOUT
1845  007100  010237  001160                  MOV     R2,$REG0
1846  007104  013737  001164  001166          MOV     $REG2,$REG3
1847  007112  104021                          ERROR   21              ;TYPE ERR MSSG
1848  007114                          14$:
1849  007114  022737  000001  001700          CMP     #1,DEVCNT       ;CHECK IF ONLY 1 DEVICE SHOULD HAVE OPERATED
1850  007122  001470                          BEQ     TST17           ;;<IF EQUAL, GO TO NEXT TEST>

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
                ;BE2 TRANSFER CHECKS
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1858  007124  012701  020310                  MOV     #ATEND,R1       ;USE R1 TO POINT TO END OF PROG
1859  007130  023701  001714          5$:     CMP     DATA2,R1        ;CHECK FOR 2000 BYTES PAST END
1860  007134  001413                          BEQ     6$              ;IF EQUAL, EXIT
1861  007136  022721  052525                  CMP     #052525,(R1)+   ;ELSE COMPARE EXPECTED RESULTS WITH R1
1862  007142  001772                          BEQ     5$              ;IF EQUAL, GO CHECK NEXT LOC
1863  007144  014137  001162                  MOV     -(R1),$REG1     ;MOVE R1 TO REG1 FOR TYPEOUT
1864  007150  010137  001160                  MOV     R1,$REG0
1865  007154  012737  052525  001166          MOV     #052525,$REG3
1866  007162  104020                          ERROR   20              ;ELSE TYPE ERR MSSG
1867  007164                          6$:
1868  007164  022737  000002  001700          CMP     #2,DEVCNT       ;CHECK IF ONLY 2 DEVICES OPERATED
```

```
1869  007172  001444                          BEQ     TST17           ;;<IF EQUAL, GO TO NEXT TEST>

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
                ;BE3 CHECK ROUTINE
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1877  007174  105777  172436        16$:    TSTB    @BE3CR1         ;CHECK IF DEVICE 3 DONE
1878  007200  100401                        BMI     20$             ;IF YES, CHECK NEXT DEVICE
1879  007202  000774                        BR      16$             ;AND GO BACK TO SEE IF DONE YET
1880  007204                        20$:
1881  007204  022777  176002  172416        CMP     #-1776,@BE3DB   ;CHECK FOR CORR VALUE IN BE3DB
1882  007212  001407                        BEQ     25$             ;IF EQUAL, SKIP ERR TYPE OUT
1883  007214  017737  172410  001162        MOV     @BE3DB,$REG1    ;MOVE FOR ERR TYPE OUT
1884  007222  012737  176002  001164        MOV     #-1776,$REG2
1885  007230  104032                        ERROR   32              ;TYPE ERR MSSG
1886  007232                        25$:
1887  007232  022737  000003  001700        CMP     #3,DEVCNT       ;CHECK IF ONLY 3 DEVICES OPERATED
1888  007240  001421                        BEQ     TST17           ;;<IF EQUAL, GO TO NEXT TEST>

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
                ;BE4 CHECK ROUTINE
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1896  007242                        22$:
1897  007242  105777  172404                TSTB    @BE4CR1         ;TEST IF DEVICE 4 IS DONE
1898  007246  100375                        BPL     22$             ;IF NOT, GO BACK TO DELAY RTN
1899  007250  022737  003000  001170        CMP     #3000,$REG4     ;CHECK IF REG4 COUNTED 3000 INTRS
1900  007256  001407                        BEQ     26$             ;IF EQUAL SKIP ERR TYPE OUT
1901  007260  013737  001170  001162        MOV     $REG4,$REG1     ;MOVE FOR TYPE OUT
1902  007266  012737  003000  001164        MOV     #3000,$REG2
1903  007274  104033                        ERROR   33              ;TYPE ERR MSSG
1904  007276                        26$:
1905  007276  042777  040000  172352        BIC     #BIT14,@BE4CR2  ;ELSE CLEAR TIME DELAY BIT
1906                          ;*************************************************************
1907                          ;*TEST 17       DUMMY END OF PROGRAM
1908                          ;*************************************************************
1909  007304  000004          TST17:  SCOPE
1910  007306  012737  000001  001210        MOV     #1,$TIMES       ;;DO 1 ITERATION
1911
1912  007314  017700  171624                MOV     @$TKB,R0        ;MOVE READ BUFF CONTENTS TO R0
1913  007320  022700  000210                CMP     #210,R0         ;DOES THE VALUE = "^H" ?
1914  007324  001001                        BNE     10$             ;IF NOT GO TO 10$
1915  007326  000000                        HALT                    ;ELSE HALT THE PROGRAM
1916  007330                        10$:
1917  007330  000137  015436                JMP     $EOP
1918                          ;*************************************************************
1919                          ;*************************************************************
1920  007334                  $1:
1921
1922  007334  017737  172244  001712        MOV     @BE1BA,DATA1    ;MOVE ADDR IN BE1BA TO DATA1 AND
1923  007342  162737  000002  001712        SUB     #2,DATA1        ;SUB 2 TO GET ACTUAL ADDR
1924  007350  012737  000001  001204        MOV     #1,$TMP4        ;SET INDICATOR FOR DEVICE 1
```

```
1925  007356  005777  172224                    TST     @BE1CR1         ;TEST FOR ERROR
1926  007362  100041                            BPL     EXS             ;IF PLUS, EXIT
1927  007364  000434                            BR      CHEX            ;ELSE FIND CAUSE OF INTR
1928  007366                            S2:
1929  007366  017737  172226  001714            MOV     @BE2BA,DATA2    ;MOVE ADDR IN BE2BA TO DATA2 AND
1930  007374  162737  000002  001714            SUB     #2,DATA2        ;SUB 2 TO GET ACTUAL ADDR
1931  007402  012737  000002  001204            MOV     #2,$TMP4        ;SET INDICATOR FOR DEVICE 2
1932  007410  005777  172206                    TST     @BE2CR1         ;TEST FOR ERROR
1933  007414  100024                            BPL     EXS             ;IF PLUS EXIT
1934  007416  000417                            BR      CHEX            ;ELSE FIND CAUSE OF INTR
1935  007420                            S3:
1936  007420  012737  000003  001204            MOV     #3,$TMP4        ;SET INDICATOR FOR DEVICE 3
1937  007426  005777  172204                    TST     @BE3CR1         ;TEST FOR ERROR
1938  007432  100015                            BPL     EXS             ;IF PLUS, EXIT
1939  007434  000410                            BR      CHEX            ;ELSE FIND CAUSE OF INTR
1940  007436                            S4:
1941  007436  005237  001170                    INC     $REG4           ;COUNT DEVICE 4'S INTRS
```

```
 1942  007442  012737  000004  001204          MOV     #4,$TMP4        ;SET INDICATOR FOR DEVICE 4
 1943  007450  005777  172176                  TST     @BE4CR1         ;TEST FOR ERROR
 1944  007454  100004                          BPL     EXS             ;IF PLUS, EXIT
 1945  007456                          CHEX:
 1946  007456  011637  001172                  MOV     (SP),$REGS      ;FOR TYPEOUT OF PC
 1947  007462  004737  010376                  JSR     PC,ERRTN        ;ELSE FIND CAUSE OF INTR
 1948  007466                          EXS:
 1949  007466  000002                          RTI
 1950                                  ;*****************************************************************
 1951                                  ;*****************************************************************
 1952  007470                          MULTI:
 1953  007470  012702  020310                  MOV     #ATEND,R2       ;R2 = END OF PROG
 1954  007474  012705  005000                  MOV     #5000,R5        ;R5 = THE # OF DATA WORDS
 1955  007500  004737  011020                  JSR     PC,DOUP         ;CREATE THOSE WORDS IN BUFFER MEMORY
 1956  007504  004737  011132                  JSR     PC,TSTOVR       ;SET UP PATTERN IN MEMORY BUFFER AREA
 1957  007510  012777  020310  172066          MOV     #ATEND,@BE1BA   ;SET REG ADDR= 1ST LOCATION AFTER END OF PROGRAM
 1958  007516  012777  176000  172056          MOV     #-2000,@BE1CC   ;SET CYCLE COUNT FOR 2000 XFERS
 1959  007524  012777  006412  172126          MOV     #SERV1,@BE1VEC  ;SET UP DEVICE INTR VECTOR
 1960  007532  012777  000340  172122          MOV     #PR7,@BE1PSW    ;SET UP DEVICE PSW VECTOR
 1961  007540  052777  040000  172044          BIS     #BIT14,@BE1CR2  ;SET BIT 14 FOR TIME DELAY ENABLE
 1962  007546  012777  042560  172032          MOV     #42560,@BE1CR1  ;DO DATIP/DATOB-FUN 1-NPR-INTR ON DONE(7)
 1963  007554  122737  000001  001700          CMPB    #1,DEVCNT       ;IF MORE THAN 1 DEVICE, LOAD THEIR REGISTERS
 1964  007562  001474                          BEQ     6$              ;OTHERWISE BEGIN TESTING
 1965  007564                          3$:
 1966  007564  012777  006436  172072          MOV     #SERV2,@BE2VEC  ;SET UP DEVICE 2 INTR VECTOR
 1967  007572  012777  000340  172066          MOV     #PR7,@BE2PSW    ;SET UP DEVICE 2 PSW VECTOR
 1968  007600  012777  020310  172012          MOV     #ATEND,@BE2BA   ;SET UP ADDR REG FOR SAME LOCATIONS AS DEVICE 1
 1969  007606  012777  177000  172002          MOV     #-1000,@BE2CC   ;SET CYCLE COUNT FOR A 1000 XFERS
 1970  007614  012777  024510  172000          MOV     #24510,@BE2CR1  ;DO DATIP/NO ROTATE-FUN 2-BR6-INTR ON DONE(6)
 1971  007622  122737  000002  001700          CMPB    #2,DEVCNT       ;IF MORE THAN 2 DEVICES, LOAD THER REGISTERS
 1972  007630  001451                          BEQ     6$              ;OTHERWISE BEGIN TESTING
 1973  007632                          4$:
 1974  007632  012777  000340  172032          MOV     #PR7,@BE3PSW    ;SET UP DEVICE 3 PSW VECTOR
 1975  007640  010377  171764                  MOV     R3,@BE3DB       ;MOVE PATTERN IN R3 TO DEVICE DATA REG
 1976  007644  012777  006462  172016          MOV     #SERV3,@BE3VEC  ;SET UP DEVICE INTR VECTOR
 1977  007652  012777  022310  171754          MOV     #ATEND+2000,@BE3BA  ;SET UP ADDR REG
 1978  007660  012777  177000  171744          MOV     #-1000,@BE3CC   ;SET UP FOR 1000 XFERS
 1979  007666  052777  040000  171746          BIS     #BIT14,@BE3CR2  ;SET BIT 14 FOR TIME DELAY ENABLE
 1980  007674  012777  003160  171734          MOV     #3160,@BE3CR1   ;DO DATO-FUN 1-FROM DB-NPR-INTR ON DONE(7)
 1981  007702  122737  000003  001700          CMPB    #3,DEVCNT       ;IF A 4TH DEVICE, GO AND LOAD REGISTERS
 1982  007710  001421                          BEQ     6$              ;OTHERWISE BEGIN TESTING
 1983  007712                          5$:
 1984  007712  010477  171726                  MOV     R4,@BE4DB       ;MOVE PATTERN IN R4 TO DEVICE DATA REG
 1985  007716  012777  000340  171752          MOV     #PR7,@BE4PSW    ;SET UP DEVICE 4 PSW VECTOR
 1986  007724  012777  006506  171742          MOV     #SERV4,@BE4VEC  ;SET UP DEVICE 4 INTR VECTOR
 1987  007732  012777  024310  171710          MOV     #ATEND+4000,@BE4BA  ;SET UP ADDR REG
 1988  007740  012777  177000  171700          MOV     #-1000,@BE4CC   ;SET CYCLE COUNT FOR 1000 XFERS
 1989  007746  012777  003104  171676          MOV     #3104,@BE4CR1   ;DO DATO-FUN 1-BR5-INTR ON DONE
 1990  007754                          6$:
 1991  007754  005277  171616                  INC     @SIMLGO         ;START DEVICES SIMULTANEOUSLY

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
                ;BACKGROUND ROUTINE FOR MULTITRANSFERS I

 1997  007760  012737  000001  001162          MOV     #1,$REG1        ;MOVE 1 TO TEMPORARY REG
```

```
 1998  007766  012701  001162                    MOV     #$REG1,R1       ;SET UP R1 AS POINTER
 1999  007772  005121                    7$:     COM     (R1)+           ;COMPLEMENT TEMP REG
 2000  007774  006041                            ROR     -(R1)           ;ROTATE CONTENTS RIGHT
 2001  007776  123737  001700  001164            CMPB    DEVCNT,$REG2    ;CHECK IF ALL DEVICES ARE DONE
 2002  010004  101372                            BHI     7$              ;IF NOT, CONTINUE WITH BACKGROUND RTN
 2003                                    ;*****************************************************************

       ; *\:/*\: *\: *\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\: *\:/*\:/*\:/*\:/*\:
       ; *\: *\: *\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
                ;DEVICE 1 TRANSFER CHECKS
                ;THERE ARE NO CHECKS FOR BE2
       ; :\* :\* :\*/:\* :\*/:\*/:\*/:\*/:\*/:\* :\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
       ; :\* :\* :\*/:\*/:\*/:\*/:\*/:\* :\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

 2012  010006  042777  040000  171576            BIC     #BIT14,@BE1CR2  ;CLEAR TIME DELAY BIT
 2013  010014  012700  020310                    MOV     #ATEND,R0       ;START CHECKING FOR CORRECT XFERS
 2014  010020                            9$:
 2015  010020  122720  000125                    CMPB    #125,(R0)+      ;COMPARE LOWER BYTE
 2016  010024  001012                            BNE     20$             ;IF NOT EQUAL, BR TO ERR MSSG
 2017  010026  023700  001712                    CMP     DATA1,R0        ;IS THIS LAST BYTE COMPARE?
 2018  010032  001422                            BEQ     9$              ;IF SO, BR TO 9$
 2019  010034  122720  000124                    CMPB    #124,(R0)+      ;ELSE COMPARE UPPER BYTE
 2020  010040  001006                            BNE     22$             ;IF NOT EQUAL, BR TO ERR MSSG
 2021  010042  023700  001712                    CMP     DATA1,R0        ;IS THIS LAST BYTE COMPARE?
 2022  010046  001414                            BEQ     9$              ;IF SO, BR TO 9$
 2023  010050  000763                            BR      8$              ;ELSE CHECK NEXT ADDR
 2024  010052                            20$:
 2025  010052  105740                            TSTB    -(R0)           ;SUB 1 TO GET ERR ADDR
 2026  010054  000401                            BR      24$             ;GO DO MOVES FOR ERR MSSG
 2027  010056                            22$:
 2028  010056  005740                            TST     -(R0)           ;SUB 2 TO GET ERR ADDR
 2029  010060                            24$:
 2030  010060  011037  001162                    MOV     (R0),$REG1      ;MOVES ARE FOR ERR MSSG
 2031  010064  010037  001160                    MOV     R0,$REG0
 2032  010070  012737  052125  001166            MOV     #052125,$REG3
 2033  010076  104015                            ERROR   15              ;ELSE TYPE ERR MSSG
 2034  010100                            9$:
 2035  010100  022737  000001  001700            CMP     #1,DEVCNT       ;IF ONLY ONE DEVICE
 2036  010106  001447                    25$:    BEQ     13$             ;IF NO MORE DEVICES,EXIT RTN
 2037  010110  122737  000002  001700            CMPB    #2,DEVCNT       ;CHECK FOR MORE THAN 2 DEVICES
 2038  010116  001773                            BEQ     25$             ;IF NOT, EXIT TEST

       ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
       ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
                ;BE3 TRANSFER CHECKS
       ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
       ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

 2046  010120  042777  040000  171514            BIC     #BIT14,@BE3CR2  ;CLEAR TIME DELAY BIT OF DEVICE 3
 2047  010126  012700  022310                    MOV     #ATEND+2000,R0  ;CHECK NEXT 2000 LOCATIONS
 2048  010132  023700  001716            10$:    CMP     DATA3,R0        ;CHECK FOR 2000 XFERS
 2049  010136  001411                            BEQ     11$             ;IF SO, CHECK NEXT BLOCK.
 2050  010140  020320                            CMP     R3,(R0)+        ;TEST FOR CORRECT PATTERNS
 2051  010142  001773                            BEQ     10$             ;IF NO ERR, CHECK ANOTHER LOC
 2052  010144  010337  001166                    MOV     R3,$REG3        ;THE MOVE IS FOR TYPEOUT REASONS
 2053  010150  014037  001162                    MOV     -(R0),$REG1
```

```
 2054  010154  010037  001160                    MOV     R0,$REG0
 2055  010160  104016                            ERROR   16              ;ELSE TYPE ERR MSSG
 2056  010162                            11$:
 2057  010162  122737  000003  001700            CMPB    #3,DEVCNT       ;CHECK FOR MORE THAN 3 DEVICES
 2058  010170  001416                            BEQ     13$             ;IF NO MORE DEVICES GO DO NEXT PATTERN

       ; #\:/#\:/#\: #\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:
       ; #\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:/#\:
                 ;BE4 TRANSFER CHECKS
       ; :\#/:\#/:\#:/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#
       ; :\# :\#/:\#/:\#/:\#/:\# :\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#

 2066  010172  012700  024310                    MOV     #ATEND+4000,R0  ;CHECK NEXT BLOCK OF 2000 LOCATIONS
 2067  010176  023700  001720            12$:    CMP     DATA4,R0        ;CHECK FOR 1000 XFERS
 2068  010202  001411                            BEQ     13$             ;IF SO, CHANGE DATA PATTERNS & START OVER AGAIN
 2069  010204  020420                            CMP     R4,(R0)+        ;ELSE CHECK FOR CORRECT PATTERNS
 2070  010206  001773                            BEQ     12$             ;IF NO ERR, CHECK ANOTHER LOCATION
 2071  010210  010437  001166                    MOV     R4,$REG3        ;THE MOVE IS FOR TYPEOUT REASONS
 2072  010214  014037  001162                    MOV     -(R0),$REG1
 2073  010220  010037  001160                    MOV     R0,$REG0
 2074  010224  104017                            ERROR   17              ;ELSE TYPE ERR MSSG
 2075  010226                            13$:
 2076  010226  000207                            RTS     PC
 2077                                      ;////////////////////////////////////////////////////////////////
 2078                                      ;////////////////////////////////////////////////////////////////
 2079                                      ;////////////////////////////////////////////////////////////////
 2080  010230                            STVEC:
 2081  010230  005000                            CLR     R0              ;R0 IS COUNTER
 2082  010232  012701  010266                    MOV     #Q1,R1          ;R1 IS ADDR OF INTR RTN
 2083  010236  012702  000510                    MOV     #510,R2         ;R2 IS DEV INTR VEC ADDR
 2084  010242                            5$:
 2085  010242  010122                            MOV     R1,(R2)+        ;LOAD INTR RTN INTO INTR VEC
 2086  010244  012722  000340                    MOV     #PR7,(R2)+      ;LOAD INTR PSW
 2087  010250  062701  000006                    ADD     #6,R1           ;GET NEXT INTR RTN
 2088  010254  005200                            INC     R0              ;INC COUNTER
 2089  010256  020037  001700                    CMP     R0,DEVCNT       ;ARE ALL AVAILABLE VECS LOADED?
 2090  010262  001367                            BNE     5$              ;IF NOT, GO AND LOAD NEXT ONE
 2091  010264  000207                            RTS     PC              ;ELSE EXIT
 2092                                      ;****************************************************************
 2093                                      ;****************************************************************
 2094
 2095                                      ; THE FOLLOWING INTR ROUTINES SHOULD HANDLE ANY ERRORS
 2096                                      ; WHICH MIGHT OCCUR BEFORE THE PROGRAM HAS A CHANCE TO
 2097                                      ; PROPERLY CHECK OUT ALL DEVICES AND SET UP THEIR INTR VECTORS
 2098
 2099
 2100  010266                            Q1:
 2101  010266  012737  000001  001204            MOV     #1,$TMP4        ;INDICATOR FOR DEV 1
 2102  010274  000413                            BR      XQ
 2103  010276                            Q2:
 2104  010276  012737  000002  001204            MOV     #2,$TMP4        ;INDICATOR FOR DEV 2
 2105  010304  000407                            BR      XQ
 2106  010306                            Q3:
 2107  010306  012737  000003  001204            MOV     #3,$TMP4        ;INDICATOR FOR DEV 3
 2108  010314  000403                            BR      XQ
 2109  010316                            Q4:
```

```
2110  010316  012737  000004  001204          MOV     #4,$TMP4        ;INDICATOR FOR DEV 4
2111  010324                          XQ:
2112  010324  004737  010376                  JSR     PC,ERRTN        ;GO TO ERR ROUTINE
2113  010330  000002                          RTI
2114                                  ;*****************************************************************
2115                                  ;*****************************************************************
2116  010332                          PWRFAL:
2117  010332  010146                  MOV     R1,-(SP)        ;SAVE CONTENTS OF R1
2118  010334  010046                  MOV     R0,-(SP)        ;SAVE CONTENTS OF R0
2119  010336  012700  001612          MOV     #BE1CR2,R0      ;R0 POINTS TO DEV 1 CR2 ADDR
2120  010342  005001                  CLR     R1              ;CLEAR R1
2121  010344                          5$:
2122  010344  042770  000020  000000  BIC     #BIT04,@(R0)    ;CLR BIT 4 OF CURRENT CR2
2123  010352  062700  000014          ADD     #14,R0          ;ADD 14 TO POINT TO NEXT CR2
2124  010356  005201                  INC     R1              ;COUNT THE NUMBER OF DEVS
2125  010360  020137  001700          CMP     R1,DEVCNT       ;REACHED MAX # ON BUS?
2126  010364  103767                  BLO     5$              ;IF NOT, CLR NEXT CR2
2127  010366  012600                  MOV     (SP)+,R0        ;ELSE RESTORE R0
2128  010370  012601                  MOV     (SP)+,R1        ;AND R1
2129  010372  000137  020150          JMP     $PWRDN          ;THEN DO REGULAR PWR DOWN RTN
2130                                  ;*****************************************************************
2131                                  ;*****************************************************************
2132  010376                          ERRTN:
2133  010376  104407                  SAVREG                  ;SAVE REGISTERS
2134  010400  012700  001576          MOV     #BE1CR2-14,R0 ;INITIALIZE R0
2135  010404  105005                  CLRB    R5              ;CLEAR DEVICE COUNTER
2136  010406                          1$:
2137  010406  105205                  INCB    R5              ;ADD 1 TO COUNTER
2138  010410  062700  000014          ADD     #14,R0          ;SET R0=ADDR OF CR2 OF NEXT DEVICE
2139  010414  120537  001204          CMPB    R5,$TMP4        ;IF COUNTER NOT EQUAL TO INDICATOR
2140  010420  001372                  BNE     1$              ;ADD 1 TO COUNTER & CHECK AGAIN
2141  010422                          CHKERR:
2142  010422  105770  000000          TSTB    @(R0)           ;CHECK FOR NO NOSACK TIMEOUT
2143  010426  100001                  BPL     1$              ;IF NOT, SEE IF THERE ARE ANY ERRS
2144  010430  104022                  ERROR   22              ;TYPE ERR MESSG FOR NO NOSACK
2145  010432                          1$:
2146  010432  032770  007540  000000  BIT     #7540,@(R0)     ;CHECK FOR OTHER ERRORS
2147  010440  001436                  BEQ     LEEV            ;IF NO ERRORS,EXIT
2148  010442  032770  004000  000000  BIT     #BIT11,@(R0)    ;CHECK FOR NO SSYN ON INTR
2149  010450  001401                  BEQ     10$             ;IF NOT SET, CHECK FOR NEXT ERR
2150  010452  104023                  ERROR   23              ;TYPE ERR MSSG FOR NO SSYN ON INTR
2151  010454                          10$:
2152  010454  132770  000040  000000  BITB    #BIT05,@(R0)    ;CHECK FOR WRONG GRANT ERR
2153  010462  001401                  BEQ     2$              ;IF NOT, CHECK BIT 6
2154  010464  104024                  ERROR   24              ;ELSE TYPE ERR MESSG FOR WRONG GRANT
2155  010466                          2$:
2156  010466  132770  000100  000000  BITB    #BIT06,@(R0)    ;CHECK FOR BUS LATE ERR
2157  010474  001401                  BEQ     3$              ;IF NOT, CHECK BIT 8
2158  010476  104025                  ERROR   25              ;TYPE ERR MSSG FOR BUS LATE
2159  010500                          3$:
2160  010500  032770  000400  000000  BIT     #BIT08,@(R0)    ;CHECK FOR NO SSYN ERR
2161  010506  001401                  BEQ     4$              ;IF NOT, CHECK BIT 9
2162  010510  104026                  ERROR   26              ;TYPE ERR MSSG FOR NO SSYN
2163  010512                          4$:
2164  010512  032770  001000  000000  BIT     #BIT09,@(R0)    ;CHECK FOR WRONG ADDR ERR
2165  010520  001401                  BEQ     5$              ;IF NOT, CHECK BIT 10
```

```
2166  010522  104027                        ERROR   27              ;TYPE ERR MSSG FOR WRONG ADDR
2167  010524                          5$:
2168  010524  032770  002000  000000        BIT     #BIT10,@(R0)    ;CHECK FOR NO GRANT ERR
2169  010532  001401                        BEQ     LEEV            ;IF NOT, EXIT
2170  010534  104030                        ERROR   30              ;TYPE ERR MSSG FOR NO GRANT
2171  010536                          LEEV:
2172  010536  162700  000002                SUB     #2,R0           ;POINT TO DEVICE CLEAR REG
2173  010542  005070  000000                CLR     @(R0)           ;CLEAR ALL ERRORS
2174  010546  104410                        RESREG                  ;RESTORE REGISTERS
2175  010550  000207                        RTS     PC              ;EXIT
2176                            ;*************************************************************
2177                            ;*************************************************************
2178                            ;*************************************************************
2179  010552                          THRU0:
2180  010552  011637  001164                MOV     (SP),$REG2      ;MOVE FOR ERR TYPE OUT
2181  010556  162737  000002  001164        SUB     #2,$REG2        ;SUB 2 FOR ACTUAL ADDR
2182  010564  104035                        ERROR   35              ;TYPE ERR MSSG
2183  010566  000002                        RTI
2184                            ;*************************************************************
2185                            ;*************************************************************
2186  010570                          ROTATE:
2187  010570  032703  000001                BIT     #BIT00,R3       ;IS LSB A 1 OR 0 ?
2188  010574  001402                        BEQ     5$              ;IF 0, GO TO 5$
2189  010576  000261                        SEC                     ;ELSE SET C BIT OF COND CODES
2190  010600  000401                        BR      10$             ;AND GO ROTATE
2191  010602                          5$:
2192  010602  000241                        CLC                     ;CLEAR C BIT OF COND CODES
2193  010604                          10$:
2194  010604  006003                        ROR     R3              ;ROTATE R3
2195  010606  032704  000001                BIT     #BIT00,R4       ;IS LSB A 1 OR 0 ?
2196  010612  001402                        BEQ     15$             ;IF 0, GO TO 15$
2197  010614  000261                        SEC                     ;ELSE SET C BIT OF COND CODES
2198  010616  000401                        BR      20$             ;AND GO ROTATE
2199  010620                          15$:
2200  010620  000241                        CLC                     ;CLEAR C BIT OF COND CODES
2201  010622                          20$:
2202  010622  006004                        ROR     R4              ;ROTATE R4
2203  010624  000207                        RTS     PC
2204                            ;*************************************************************
2205                            ;*************************************************************
2206  010626                          NOG:
2207  010626                          2$:
2208  010626  010037  177776                MOV     R0,PSW          ;SET UP PROCESSOR STATUS
2209  010632  012777  020310  170744        MOV     #ATEND,@BE1BA   ;SET UP ADDR REG
2210  010640  012777  177777  170734        MOV     #-1,@BE1CC      ;SET UP CYCLE COUNT FOR 1 CYCLE
2211  010646  010177  170734                MOV     R1,@BE1CR1      ;DO FUN 1 ON BR LEVELS IN R1
2212  010652  000240                        NOP                     ;WAIT FOR DEVICE TO ATTEMPT TO  DO XFER
2213  010654  022777  177777  170720        CMP     #-1,@BE1CC      ;SEE IF DEVICE OPERATED
2214  010662  001005                        BNE     4$              ;IF IT DID,GO TYPE ERR MSSG
2215  010664  106201                        ASRB    R1              ;SHIFT BYTE RIGHT TO LOWER BRRQ
2216  010666  122701  000001                CMPB    #1,R1           ;IF BYTE IS NOT EQUAL TO 1
2217  010672  001355                        BNE     2$              ;GO TO 2$
2218  010674  000402                        BR      EXNOG           ;EXIT
2219  010676                          4$:
2220  010676  004737  010704                JSR     PC,ERRS
2221  010702  000207                EXNOG:  RTS     PC              ;EXIT SUB RTN
```

```
2222                                        ;*****************************************************************
2223  010704                                ERRS:
2224  010704  017737  170670  001160              MOV     @BE1DB,$REG0    ;MOVES ARF FOR TYPEOUTS
2225  010712  017737  170664  001162              MOV     @BE1CC,$REG1
2226  010720  017737  170660  001164              MOV     @BE1BA,$REG2
2227  010726  017737  170654  001166              MOV     @BE1CR1,$REG3
2228  010734  010037  001170                    MOV     R0,$REG4
2229  010740  104002                             ERROR   2
2230  010742  000207                             RTS     PC              ;EXIT ERROR RTN
2231                                        ;*****************************************************************
2232                                        ;*****************************************************************
2233  010744                                BKGD:
2234  010744  012737  031463  001162              MOV     #031463,$REG1   ;START OF BACKGROUND ROUTINE
2235  010752  012701  001163                    MOV     #$REG1+1,R1     ;USE R1 TO POINT TO TEST PATTERN
2236  010756  105441                       1$:   NEGB    -(R1)           ;DECREMENT LOC AND NEGATE BYTE=(031715)
2237  010760  105421                             NEGB    (R1)+           ;NEGATE BYTE THEN INCREMENT LOC=(031463)
2238  010762  105421                             NEGB    (R1)+           ;NEGATE BYTE THEN INCREMENT LOC=(146463)
2239  010764  105770  000000                    TSTB    @(R0)           ;TEST FOR DONE BIT OF DEVICE IN R0
2240  010770  100402                             BMI     2$              ;IF DONE, GO CHECK RESULTS
2241  010772  105441                             NEGB    -(R1)           ;ELSE DECREMENT LOC AND NEGATE BYTE=(031463)
2242  010774  000770                             BR      1$              ;CONTINUE WITH BACKGROUND
2243  010776                                2$:
2244
2245  010776  005741                             TST     -(R1)           ;BRING POINTER DOWN TO REG1
2246  011000  022711  146463                    CMP     #146463,(R1)    ;COMPARE EXPECTED PATTERN WITH THAT IN R1
2247  011004  001404                             BEQ     BKEX            ;IF EQUAL, EXIT THIS RTN
2248  011006  012737  146463  001164              MOV     #146463,$REG2   ;MOVE FOR TYPE OUT
2249  011014  104031                             ERROR   31              ;ELSE TYPE ERR MSSG
2250  011016  000207                       BKEX: RTS     PC
2251                                        ;*****************************************************************
2252                                        ;*****************************************************************
2253  011020                                DOUP:
2254  011020  012701  000001                    MOV     #1,R1           ;INIT R1 TO 1
2255  011024                                5$:
2256  011024  010122                             MOV     R1,(R2)+        ;MOVE CONTENTS OF R1 TO AREA IN R2
2257  011026  005201                             INC     R1              ;ADD 1 TO R1
2258  011030  020105                             CMP     R1,R5           ;IS # OF MOVES = TO # IN R5?
2259  011032  101774                             BLOS    5$              ;IF NOT, DO ANOTHER MOVE
2260  011034  000207                             RTS     PC              ;ELSE EXIT
2261                                        ;*****************************************************************
2262                                        ;*****************************************************************
2263
2264  011036                                CNTR:
2265  011036  012737  000001  001170              MOV     #1,$REG4        ;INITIALIZE COUNTER REG
2266  011044  062737  000001  001170       1$:   ADD     #1,$REG4        ;ADD 1 TO IT
2267  011052  022737  000106  001170              CMP     #70.,$REG4      ;COUNT OF 70 IS APPROX 20 US
2268  011060  001371                             BNE     1$              ;IF NOT, GO BACK AND ADD 1 TO REG4
2269  011062  000207                             RTS     PC              ;EXIT
2270                                        ;*****************************************************************
2271                                        ;*****************************************************************
2272  011064                                ADLI:
2273  011064  012700  040000                    MOV     #40000,R0       ;USE R0 TO SET BIT 14
2274  011070                                1$:
2275  011070  012777  177777  170504              MOV     #-1,@BE1CC      ;SET CYCLE COUNT = 1 XFER
2276  011076  010077  170502                    MOV     R0,@BE1BA       ;SET ADDR AS SPECIFIED IN R0
2277  011102  012777  002041  170476              MOV     #2041,@BE1CR1   ;DO DATI-FUN 1-NPR
```

```
                                        JSR     PC,CNTR         ;ALLOW TIME FOR RDY BIT TO SET
                                        CMP     #100000,R0      ;CHECK IF BIT 15 OF R0 IS SET
                                        BEQ     EXAD            ;IF SET, GO SET NEXT ADDR LINE
                                        MOV     #100000,R0      ;ELSE, NOW SET BIT 15 OF R0
                                        BR      1$              ;GO BACK AND CHECK THAT ADDR
                                EXAD:
                                        RTS     PC              ;EXIT SUB ROUTINE
                                ;***************************************************************
                                ;***************************************************************
                                TSTOVR:
                                        MOV     #PR3,PSW        ;PS=3
                                        CLR     $REG2           ;CLEAR REG FOR INTR ON DONE COUNTER
                                        MOV     #ATEND,R0       ;SET UP R0 AS POINTER
                                1$:     MOV     #125252,(R0)+   ;MOVE DATA PATTERN TO AVAILABLE MEMORY
                                        CMP     #ATEND+2000,R0  ;CHECK FOR A 2000 MOVES
                                        BNE     1$              ;IF NOT, GO BACK AND MOVE AGAIN
                                        RTS     PC              ;EXIT
                                ;***************************************************************
                                ;***************************************************************
011164  005015  005015  047125  QNO:    .ASCIZ  <15><12><15><12>\UNIBUS SYSTEMS EXERCISER DIAGNOSTIC - DZKUA-C    BY:M.S
011267     015  042012  053105  FOR4:   .ASCIZ  <15><12>\DEV 4 MUST HAVE TIME DELAY SET @ 100 US OR LATENCY ERR MAY OCCU
011374  050103  020125  051124  EM1:    .ASCIZ \CPU TRAPPED THRU LOCATION 4 -TIMEOUT \
011442  040440  042104  020122  DH1:    .ASCIZ \ ADDR   $ERRPC #ERR/TST#\
011473     103  052520  044440  EM2:    .ASCII  \CPU ISSUED A BUS GRANT WITH PSW = 7\
011536  042504  020126  020061          .ASCIZ  \DEV 1 SHOULD NOT HAVE BECOME BUS MASTER\
011606  042502  042061  020102  DH2:    .ASCIZ \BE1DB   BE1CC   BE1BA   BE1CR1   PSW   $ERRPC #ERR/TST#\
011676  050103  020125  044504  EM3:    .ASCIZ \CPU DID NOT ISSUE BUS NPG\
011730  042502  041461  030522  DH3:    .ASCIZ  \BE1CR1   BE1CC  FM-PS   TO-PS   $ERRPC #ERR/TST#\
012011     103  052520  042040  EM4:    .ASCIZ  \CPU DID NOT ISSUE BUS GRANT 7\
012047     103  052520  042040  EM5:    .ASCIZ  \CPU DID NOT ISSUE BUS GRANT 6\
012105     103  052520  042040  EM6:    .ASCIZ  \CPU DID NOT ISSUE BUS GRANT 5\
012143     103  052520  042040  EM7:    .ASCIZ  \CPU DID NOT ISSUE BUS GRANT 4\
012201     117  042516  047440  EM10:   .ASCIZ  \ONE OR MORE DEVICES DID NOT INTERRUPT\
012247     124  044510  020123  DH10:   .ASCII  \THIS IS THE ORDER IN WHICH THEY INTERRUPTED\<15><12>
012324  020040  051461  020124          .ASCIZ  \  1ST     2ND     3RD     4TH   $ERRPC #ERR/TST#\
012405     102  051525  040440  EM11:   .ASCIZ  \BUS ADDRESS LINES <A17:A14> DID NOT FUNCTION PROPERLY\
012473     103  030505  051103  DH11:   .ASCIZ  \BE1CR1  BE1CR2   BE1BA  $ERRPC #ERR/TST#\
012544  050103  020125  047516  EM12:   .ASCIZ  \CPU NO SACK TIMEOUT LOGIC FAILED(TO NEGATE BUS GRANT)\
012632  042502  041461  030522  DH12:   .ASCIZ  \BE1CR1  BE1CR2  $ERRPC #ERR/TST#\
012673     103  052520  042040  EM13:   .ASCIZ  \CPU DID NOT PROPERLY EXECUTE AN ACLO/DCLO SEQUENCE\
012756  042444  051122  041520  DH13:   .ASCIZ  \$ERRPC #ERR/TST#\
012777     103  052520  042040  EM14:   .ASCIZ  \CPU DID NOT TRAP FROM BUS PARITY ERROR PA/PB = 0/1\
013062  042504  020126  020061  EM15:   .ASCII  \DEV 1 DID DATIP WITH ROL ON DATOB TO MEMORY\<15><12>
013137     124  042510  052040          .ASCIZ  \THE TRANSFER TO THE FOLLOWING LOCATION WAS  INCORRECT\
013225     115  046505  051117  DH15:   .ASCII  \MEMORY  ACTUAL  CORRECT\<15><12>
013256  046040  041517  020040          .ASCIZ  \ LOC     DATA    DATA   $ERRPC #ERR/TST# $ICNT #\
013337     104  053105  041511  EM16:   .ASCIZ  \DEVICE 3'S DATO TO MEMORY DID NOT EQUAL PATTERN IN R3\
013425     104  053105  041511  EM17:   .ASCIZ  \DEVICE 4'S DATO TO MEMORY DID NOT EQUAL PATTERN IN R4\
013513     104  053105  041511  EM20:   .ASCIZ \DEVICE 1 DOING FUN 1-NPR-DATIP; INCORRECT PATTERN IN MEMORY\
013607     104  053105  041511  EM21:   .ASCIZ \DEVICE 2 DOING FUN 2-NPR-DATOB; INCORRECT PATTERN IN MEMORY\
013703     102  052111  033440  EM22:   .ASCIZ \BIT 7 OF CR2 SET-CPU DID NOT TIME OUT WITH SACK INHIBITED\
013775     104  053105  021440  DH22:   .ASCIZ \DEV #     PC    $ERRPC  #ERR/TST#\
014037     102  052111  030440  EM23:   .ASCIZ \BIT 11 OF CR2 SET-NO SSYN ON INTR SIGNAL\
014110  044502  020124  020065  EM24:   .ASCIZ \BIT 5 OF CR2 SET-RECEIVED WRONG GRANT\
014156  044502  020124  020066  EM25:   .ASCIZ  \BIT 6 OF CR2 SET-BUS LATE\
```

```
014210  044502  020124  020070  EM26:   .ASCIZ  \BIT 8 OF CR2 SET-NO SSYN OCCURRED\
014252  044502  020124  020071  EM27:   .ASCIZ  \BIT 9 OF  CR2 SET-WRONG ADDRESS ON BUS\
014321     102  052111  030440  EM30:   .ASCIZ  \BIT 10 OF CR2 SET-DEVICE RECEIVED OTHER THAN ONE GRANT
014410  045502  047107  020104  EM31:   .ASCII  \BKGND ROUTINE INSTRUCTIONS OF NEGB'S WERE NOT DONE\<15><12>
014474  047503  051122  041505          .ASCIZ  \CORRECTLY TO $REG1 DURING MULTITRANSFERS II\
014550  041501  052524  046101  DH31:   .ASCII  \ACTUAL  CORR'T  \<15><12>
014572  040504  040524  020040          .ASCIZ  \DATA    DATA    $ERRPC #ERR/TST# $ICNT #\
014643     104  053105  031440  EM32:   .ASCIZ  \DEV 3 DID DATI BUT HAS INCORRECT VALUES IN DATA REG\
014727     104  053105  032040  EM33:   .ASCIZ  \DEV 4 DID NOT INTR THE CORRECT # OF TIMES\
015001     114  051501  020124  EM34:   .ASCII  \LAST DATIP TO DEVICE 1 DB WAS INCORRECT- EITHER DEVICE DID\<15><12>
015075     116  052117  053440          .ASCIZ  \NOT WORK OR BUFFER AREA WAS NOT SET UP PROPERLY\
015155     015  041412  052520  EM35:   .ASCIZ  <15><12>\CPU TRAPPED THRU LOC 0 TO CATCH IMPROPERLY LOADED VECTORS\
           015252                          .EVEN
015252  001164  001116  001102  DT1:    .WORD   $REG2,$ERRPC,$TSTNM,0
015262  001160  001162  001164  DT2:    .WORD   $REG0,$REG1,$REG2,$REG3,$REG4,$ERRPC,$TSTNM,0
015302  001160  001162  001164  DT3:    .WORD   $REG0,$REG1,$REG2,$REG3,$ERRPC,$TSTNM,0
015320  001162  001164  001166  DT10:   .WORD   $REG1,$REG2,$REG3,$REG4,$ERRPC,$TSTNM,0
015336  001162  001164  001166  DT11:   .WORD   $REG1,$REG2,$REG3,$ERRPC,$TSTNM,0
015352  001160  001162  001116  DT12:   .WORD   $REG0,$REG1,$ERRPC,$TSTNM,0
015364  001116  001102  000000  DT13:   .WORD   $ERRPC,$TSTNM,0
015372  001160  001162  001166  DT15:   .WORD   $REG0,$REG1,$REG3,$ERRPC,$TSTNM,$ICNT,0
015410  001204  001172  001116  DT22:   .WORD   $TMP4,$REG5,$ERRPC,$TSTNM,0
015422  001162  001164  001116  DT31:   .WORD   $REG1,$REG2,$ERRPC,$TSTNM,$ICNT,0
(2)                             ;*****************************************************************
2299                            ;*****************************************************************
2300                            ;*****************************************************************
2301                            ;*****************************************************************
2302
2303                            .SBTTL  END OF PASS ROUTINE
2304
2305                            ;*INCREMENT THE PASS NUMBER ($PASS)
2306                            ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
2307                            ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
2308                            ;*IF THERES A MONITOR GO TO IT
2309                            ;*IF THERE ISN'T JUMP TO TST1
2310
2311  015436                    $EOP:
2312  015436  000004                   SCOPE
2313  015440  005037  001102           CLR     $TSTNM          ;;ZERO THE TEST NUMBER
2314  015444  005037  001210           CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
2315  015450  005237  001100           INC     $PASS           ;;INCREMENT THE PASS NUMBER
2316  015454  042737  100000  001100   BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
2317  015462  005327                   DEC     (PC)+           ;;LOOP?
2318  015464  000001           $EOPCT: .WORD   1
2319  015466  003063                   BGT     $DOAGN          ;;YES
2320  015470  012737                   MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
2321  015472  000001           $ENDCT: .WORD   1
2322  015474  015464                   $EOPCT
2323  015476  104400  015504           TYPE    ..+4            ;;TYPE ASCIZ STRING
2324  015502  000407                   BR      64$             ;;GET OVER THE ASCIZ
2325                                   ;;.ASCIZ      <12><15>/END PASS #/
2326  015522                   64$:
2327  015522  013746  001100           MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
2328                                                           ;;TYPE PASS NUMBER
2329  015526  104404                   TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
2330  015530  104400  015536           TYPE    ..+4            ;;TYPE ASCIZ STRING
```

```
2331  015534  000421                        BR      65$             ;;GET OVER THE ASCIZ
2332                                        ;;.ASCIZ      /  TOTAL ERRORS SINCE LAST REPORT /
2333  015600                          65$:
2334  015600  013746  001112                MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
2335                                                                ;;TOTAL NUMBER OF ERRORS
2336  015604  104404                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
2337  015606  104400  001221                TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
2338  015612  005037  001112                CLR     $ERTTL          ;;CLEAR ERROR TOTAL
2339  015616                          $GET42:
2340
2341  015616  013700  000042                MOV     @#42,R0         ;;GET MONITOR ADDRESS
2342  015622  001405                        BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
2343  015624  000005                        RESET                   ;;CLEAR THE WORLD
2344  015626  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
2345  015630  000240                        NOP                     ;;SAVE ROOM
2346  015632  000240                        NOP                     ;;FOR
2347  015634  000240                        NOP                     ;;ACT11
2348  015636                          $DOAGN:
2349  015636  000137  003506                JMP     @#TST1          ;;RETURN
2350  015642     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
2351           015646                        .EVEN
2352                                  ;*****************************************************************
2353
2354                                  .SBTTL  SCOPE HANDLER ROUTINE
2355
2356                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
2357                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
2358                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
2359                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2360                                  ;*SW14=1        LOOP ON TEST
2361                                  ;*SW11=1        INHIBIT ITERATIONS
2362                                  ;*SW09=1        LOOP ON ERROR
2363                                  ;*SW08=1        LOOP ON TEST IN SWR<7:0>
2364                                  ;*CALL
2365                                  ;*      SCOPE           ;;SCOPE=IOT
2366
2367  015646                          $SCOPE:
2368  015646  032777  040000  163262  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
2369  015654  001111                        BNE     $OVER           ;;YES IF SW14=1
2370                                  ;*****START OF CODE FOR THE XOR TESTER*****
2371  015656  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
2372                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
2373  015660  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
2374  015664  012737  015704  000004        MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
2375  015672  005737  177060                TST     @#177060        ;;TIME OUT ON XOR?
2376  015676  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
2377  015702  000463                        BR      $SVLAD          ;;GO TO THE NEXT TEST
2378  015704  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
2379  015706  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
2380  015712  000423                        BR      7$              ;;LOOP ON THE PRESENT TEST
2381  015714                          6$:;*****END OF CODE FOR THE XOR TESTER*****
2382  015714  032777  000400  163214        BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
2383  015722  001404                        BEQ     2$              ;;BR IF NO
2384  015724  127737  163206  001102        CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
2385  015732  001462                        BEQ     $OVER           ;;BR IF YES
2386  015734  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
```

```
2387  015740  001421                          BEQ     3$                ;;BR IF NO
2388  015742  123737  001115  001103          CMPB    $ERMAX,$ERFLG     ;;MAX. ERRORS FOR THIS TEST OCCURRED?
2389  015750  101015                          BHI     3$                ;;BR IF NO
2390  015752  032777  001000  163156          BIT     #BIT09,@SWR       ;;LOOP ON ERROR?
2391  015760  001404                          BEQ     4$                ;;BR IF NO
2392  015762  013737  001110  001106  7$:     MOV     $LPERR,$LPADR     ;;SET LOOP ADDRESS TO LAST SCOPE
2393  015770  000443                          BR      $OVER
2394  015772  105037  001103          4$:     CLRB    $ERFLG            ;;ZERO THE ERROR FLAG
2395  015776  005037  001210                  CLR     $TIMES            ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
2396  016002  000415                          BR      1$                ;;ESCAPE TO THE NEXT TEST
2397  016004  032777  004000  163124  3$:     BIT     #BIT11,@SWR       ;;INHIBIT ITERATIONS?
2398  016012  001011                          BNE     1$                ;;BR IF YES
2399  016014  005737  001100                  TST     $PASS             ;;IF FIRST PASS OF PROGRAM
2400  016020  001406                          BEQ     1$                ;;      INHIBIT ITERATIONS
2401  016022  005237  001104                  INC     $ICNT             ;;INCREMENT ITERATION COUNT
2402  016026  023737  001210  001104          CMP     $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
2403  016034  002021                          BGE     $OVER             ;;BR IF MORE ITERATION REQUIRED
2404  016036  012737  000001  001104  1$:     MOV     #1,$ICNT          ;;REINITIALIZE THE ITERATION COUNTER
2405  016044  013737  016114  001210          MOV     $MXCNT,$TIMES     ;;SET NUMBER OF ITERATIONS TO DO
2406  016052  105237  001102          $SVLAD: INCB    $TSTNM            ;;COUNT TEST NUMBERS
2407  016056  011637  001106                  MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
2408  016062  011637  001110                  MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
2409  016066  005037  001212                  CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
2410  016072  112737  000001  001115          MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
2411  016100  013777  001102  163032  $OVER:  MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
2412  016106  013716  001106                  MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
2413  016112  000002                          RTI                       ;;FIXES PS
2414  016114  000040                  $MXCNT: 40                        ;;MAX. NUMBER OF ITERATIONS
2415                                  ;*****************************************************************
2416
2417                                  .SBTTL  'ERROR HANDLER ROUTINE
2418
2419                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2420                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2421                                  ;*AND GO TO $ERRTYP ON ERROR
2422                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2423                                  ;*SW15=1         HALT ON ERROR
2424                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
2425                                  ;*SW10=1         BELL ON ERROR
2426                                  ;*SW09=1         LOOP ON ERROR
2427                                  ;*CALL
2428                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
2429
2430  016116                          $ERROR:
2431  016116  105237  001103          7$:     INCB    $ERFLG            ;;SET THE ERROR FLAG
2432  016122  001775                          BEQ     7$                ;;DON'T LET THE FLAG GO TO ZERO
2433  016124  013777  001102  163006          MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER AND ERROR FLAG
2434  016132  032777  002000  162776          BIT     #BIT10,@SWR       ;;BELL ON ERROR?
2435  016140  001402                          BEQ     1$                ;;NO - SKIP
2436  016142  104400  001214                  TYPE    ,$BELL            ;;RING BELL
2437  016146  005237  001112          1$:     INC     $ERTTL            ;;COUNT THE NUMBER OF ERRORS
2438  016152  011637  001116                  MOV     (SP),$ERRPC       ;;GET ADDRESS OF ERROR INSTRUCTION
2439  016156  162737  000002  001116          SUB     #2,$ERRPC
2440  016164  117737  162726  001114          MOVB    @$ERRPC,$ITEMB    ;;STRIP AND SAVE THE ERROR ITEM CODE
2441  016172  032777  020000  162736          BIT     #BIT13,@SWR       ;;SKIP TYPEOUT IF SET
2442  016200  001004                          BNE     20$               ;;SKIP TYPEOUTS
```

```
2443  016202  004737  016264                JSR   PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
2444  016206  104400  001221                TYPE  ,$CRLF
2445  016212                        20$:
2446  016212  005777  162720          2$:   TST   @SWR              ;;HALT ON ERROR
2447  016216  100006                BPL   3$                ;;SKIP IF CONTINUE
2448  016220  000000                HALT                    ;;HALT ON ERROR!
2449  016222  022737  015626  000042      CMP   #$ENDAD,@#42      ;;ACT-11 AUTO-ACCEPT?
2450  016230  001001                BNE   3$                ;;BRANCH IF NO
2451  016232  000000                HALT                    ;;YES
2452  016234  032777  001000  162674 3$:   BIT   #BIT09,@SWR      ;;LOOP ON ERROR SWITCH SET?
2453  016242  001402                BEQ   4$                ;;BR IF NO
2454  016244  013716  001110                MOV   $LPERR,(SP)       ;;FUDGE RETURN FOR LOOPING
2455  016250  005737  001212          4$:   TST   $ESCAPE           ;;CHECK FOR AN ESCAPE ADDRESS
2456  016254  001402                BEQ   5$                ;;BR IF NONE
2457  016256  013716  001212                MOV   $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
2458  016262                        5$:
2459  016262  000002                RTI                     ;;RETURN
2460                                ;*****************************************************************
2461
2462                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
2463
2464                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
2465                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
2466                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
2467
2468  016264                        $ERRTYP:
2469  016264  104400  001221                TYPE  ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
2470  016270  010046                MOV   R0,-(SP)          ;;SAVE R0
2471  016272  005000                CLR   R0                ;;PICKUP THE ITEM INDEX
2472  016274  153700  001114                BISB  @#$ITEMB,R0
2473  016300  001004                BNE   1$                ;;IF ITEM NUMBER IS ZERO, JUST
2474                                                        ;;TYPE THE PC OF THE ERROR
2475  016302  013746  001116                MOV   $ERRPC,-(SP)      ;;SAVE $ERRPC FOR TYPEOUT
2476                                                        ;;ERROR ADDRESS
2477  016306  104401                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2478  016310  000426                BR    6$                ;;GET OUT
2479  016312  005300          1$:   DEC   R0                ;;ADJUST THE INDEX SO THAT IT WILL
2480  016314  006300                ASL   R0                ;;      WORK FOR THE ERROR TABLE
2481  016316  006300                ASL   R0
2482  016320  006300                ASL   R0
2483  016322  062700  001224                ADD   #$ERRTB,R0        ;;FORM TABLE POINTER
2484  016326  012037  016336                MOV   (R0)+,2$          ;;PICKUP "ERROR MESSAGE" POINTER
2485  016332  001404                BEQ   3$                ;;SKIP TYPEOUT IF NO POINTER
2486  016334  104400                TYPE                    ;;TYPE THE "ERROR MESSAGE"
2487  016336  000000          2$:   .WORD 0                 ;;"ERROR MESSAGE" POINTER GOES HERE
2488  016340  104400  001221                TYPE  ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
2489  016344  012037  016354          3$:   MOV   (R0)+,4$          ;;PICKUP "DATA HEADER" POINTER
2490  016350  001404                BEQ   5$                ;;SKIP TYPEOUT IF 0
2491  016352  104400                TYPE                    ;;TYPE THE "DATA HEADER"
2492  016354  000000          4$:   .WORD 0                 ;;"DATA HEADER" POINTER GOES HERE
2493  016356  104400  001221                TYPE  ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
2494  016362  011000          5$:   MOV   (R0),R0           ;;PICKUP "DATA TABLE" POINTER
2495  016364  001004                BNE   7$                ;;GO TYPE THE DATA
2496  016366  012600          6$:   MOV   (SP)+,R0          ;;RESTORE R0
2497  016370  104400  001221                TYPE  ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
2498  016374  000207                RTS   PC                ;;RETURN
```

```
                                7$:
016376  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
016400  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
016402  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
016404  001770                          BEQ     6$              ;;BR IF NO
016406  104400  016414                  TYPE    ,9$             ;;TYPE TWO(2) SPACES
016412  000771                          BR      7$              ;;LOOP
016414  020040    000           8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
        016420                          .EVEN
                                ;***************************************************
                                .SBTTL  TTY INPUT ROUTINE
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;
016420  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
016422  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
016430  105777  162506          1$:     TSTB    @$TKS           ;;WAIT FOR
016434  100375                          BPL     1$              ;;A CHARACTER
016436  117766  162502  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
016444  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
016452  000002                          RTI                     ;;GO BACK TO USER
                                ;***************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
016454  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
016456  012703  016562          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
016462  022703  016600          2$:     CMP     #$TTYIN+16,R3   ;;BUFFER FULL?
016466  101405                          BLOS    4$              ;;BR IF YES
016470  104405                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
016472  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
016474  122713  000177                  CMPB    #177,(R3)       ;;IS IT A RUBOUT
016500  001003                          BNE     3$              ;;SKIP IF NOT
016502  104400  001220          4$:     TYPE    ,$QUES          ;;TYPE A '?'
016506  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
016510  111337  016560          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
016514  104400  016560                  TYPE    ,9$
016520  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
016524  001356                          BNE     2$              ;;LOOP IF NOT RETURN
016526  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
016532  104400  001222                  TYPE    ,$LF            ;;TYPE A LINE FEED
016536  012603                          MOV     (SP)+,R3        ;;RESTORE R3
016540  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
016542  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
016550  012766  016562  000004          MOV     #$TTYIN,4(SP)
016556  000002                          RTI                     ;;RETURN
016560    000           9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
016561    000                   .BYTE   0               ;;TERMINATOR
```

```
2555  016562  000016                  $TTYIN: .BLKB   16              ;;RESERVE 16 BYTES FOR TTY INPUT
2556                                  ;*******************************************************************
2557
2558                                  .SBTTL  ROUTINE TO SIZE MEMORY
2559
2560                                  ;*CALL:
2561                                  ;*      JSR     PC,$SIZE
2562                                  ;*      RETURN
2563                                  ;*$LSTAD WILL CONTAIN:
2564                                  ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
2565                                  ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
2566                                  ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
2567                                  ;*$KT11 IS THE MEMORY MANAGEMENT KEY
2568                                  ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
2569                                  ;*      MUST BE SETUP BEFORE THE CALL
2570                                  ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
2571                                  ;*      DETERMINED BY ROUTINE
2572
2573  016600  010046                  $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
2574  016602  010146                          MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
2575  016604  010246                          MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
2576  016606  010346                          MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
2577  016610  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
2578  016614  013746  000006                  MOV     @#ERRVEC+2,-(SP)
2579  016620  010600                          MOV     SP,R0           ;;SAVE THE STACK POINTER
2580  016622  013746  000034                  MOV     @#34,-(SP)      ;;SETUP THE TRAP VECTOR
2581  016626  012737  016636  000034          MOV     #1$,@#34        ;; TO GET THE PS
2582  016634  104400                          TRAP
2583  016636  016637  000002  000006  1$:     MOV     2(SP),@#ERRVEC+2 ;;SET ERRVEC PS
2584  016644  012716  016652                  MOV     #$SIZE1,(SP)    ;; TO PRESENT PS
2585  016650  000002                          RTI
2586  016652  012637  000034          $SIZE1: MOV     (SP)+,@#34      ;;RESTORE TRAP VECTOR
2587  016656  012701  003776                  MOV     #3776,R1        ;;SETUP ADDRESS
2588  016662  105727                          TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
2589  016664  000200                  $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
2590  016666  100055                          BPL     $CORE           ;;BR IF NO
2591  016670  012737  017014  000004          MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
2592  016676  005737  177572                  TST     @#SR0           ;;KT11 ARE YOU THERE?
2593  016702  052737  100000  016664          BIS     #100000,$KT11   ;;YES--SET KT11 KEY
2594  016710  005046                          CLR     -(SP)           ;;INITIALIZE FOR "PAR" LOADING
2595  016712  012702  172340                  MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST "PAR"
2596  016716  012703  000010                  MOV     #^D8,R3         ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
2597  016722  012762  077406  177740  1$:     MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
2598  016730  011622                          MOV     (SP),(R2)+      ;;LOAD "PAR"
2599  016732  062716  000200                  ADD     #200,(SP)       ;;UPDATE FOR NEXT "PAR"
2600  016736  077307                          SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
2601  016740  012742  177600                  MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
2602  016744  005042                          CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
2603  016746  012737  000020  172516          MOV     #20,@#SR3       ;;ENABLE 22 BIT MODE
2604  016754  005237  177572                  INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
2605  016760  012737  017004  000004          MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
2606  016766  005737  143776          2$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
2607  016772  062712  000040                  ADD     #40,(R2)        ;;MAKE A 1K STEP
2608  016776  023712  172356                  CMP     @#KIPAR7,(R2)   ;;LAST ONE?
2609  017002  101371                          BHI     2$              ;;NO--TRY IT
2610  017004  011202                  $KTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
```

```
 2611  017006  005037  177572                CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
 2612  017012  000421                       BR      $SIZEX
 2613  017014  042737  100000  016664  $KTNEX: BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
 2614  017022  012737  017052  000004  $CORE:  MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
 2615  017030  005002                       CLR     R2              ;;SET UP BANK
 2616  017032  062701  004000          1$:  ADD     #4000,R1        ;;INCREMENT BY 1K
 2617  017036  062702  000040               ADD     #40,R2          ;;1K STEP
 2618  017042  005711                       TST     (R1)            ;;TRAP ON TIME OUT
 2619  017044  022701  177776               CMP     #177776,R1      ;;LAST ONE
 2620  017050  001370                       BNE     1$              ;;NO--TRY AGAIN
 2621  017052  162701  004000  $CROUT: SUB     #4000,R1
 2622  017056  162702  000040  $SIZEX: SUB     #40,R2          ;;DROP BACK
 2623  017062  010006                       MOV     R0,SP           ;;RESTORE THE STACK
 2624  017064  012637  000006               MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
 2625  017070  012637  000004               MOV     (SP)+,@#ERRVEC
 2626  017074  010137  017116               MOV     R1,$LSTAD       ;;LAST ADDRESS
 2627  017100  010237  017120               MOV     R2,$LSTBK       ;;LAST BANK
 2628  017104  012603                       MOV     (SP)+,R3        ;;RESTORE R3
 2629  017106  012602                       MOV     (SP)+,R2        ;;RESTORE R2
 2630  017110  012601                       MOV     (SP)+,R1        ;;RESTORE R1
 2631  017112  012600                       MOV     (SP)+,R0        ;;RESTORE R0
 2632  017114  000207                       RTS     PC
 2633  017116  000000  $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
 2634  017120  000000  $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
 2635                  ;*******************************************************************
 2636
 2637                  .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
 2638
 2639                  ;*SAVE R0-R5
 2640                  ;*CALL:
 2641                  ;*      SAVREG
 2642                  ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
 2643                  ;*
 2644                  ;*TOP---(+16)
 2645                  ;* +2---(+18)
 2646                  ;* +4---R5
 2647                  ;* +6---R4
 2648                  ;* +8---R3
 2649                  ;*+10---R2
 2650                  ;*+12---R1
 2651                  ;*+14---R0
 2652
 2653  017122          $SAVREG:
 2654  017122  010046          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 2655  017124  010146          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 2656  017126  010246          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 2657  017130  010346          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 2658  017132  010446          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 2659  017134  010546          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 2660  017136  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
 2661  017142  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
 2662  017146  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
 2663  017152  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
 2664  017156  000002          RTI
 2665
 2666                  ;*RESTORE R0-R5
```

```
                                        ;*CALL:
                                        ;*      RESREG
                                        $RESREG:
017160                                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
017162  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
017164  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
017170  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
017174  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
017200  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
017202  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
017204  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
017206  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
017210  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
017212  000002                          RTI
                                        ;*****************************************************************

                                        .SBTTL  TYPE ROUTINE

                                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                        ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                        ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                        ;*
                                        ;*CALL:
                                        ;*1) USING A TRAP INSTRUCTION
                                        ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                        ;*OR
                                        ;*      TYPE
                                        ;*      MESADR
                                        ;*

017216  105737  001155          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
017222  100002                          BPL     1$              ;;BR IF YES
017224  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
017226  000407                          BR      3$              ;;LEAVE
017230  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
017232  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
017236  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
017240  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
017242  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
017244  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
017246  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
017252  000002                          RTI                     ;;RETURN
017254  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
017260  001426                          BEQ     8$
017262  122716  000200                  CMPB    #TCRLF,(SP)     ;;BRANCH IF NOT <CRLF>
017266  001004                          BNE     5$
017270  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
017272  104400                          TYPE                    ;;TYPE A CR AND LF
017274  001221                          $CRLF
017276  000757                          BR      2$              ;;GET NEXT CHARACTER
017300  004737  017362          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
017304  123726  001154          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
017310  001352                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
017312  013746  001152                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
```

```
2723                                                    ;;AND THE NULL CHAR.
2724  017316  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
2725  017322  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2726  017324  004737  017362                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
2727  017330  105337  017426                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2728  017334  000770                          BR      7$              ;;LOOP
2729
2730                                          ;HORIZONTAL TAB PROCESSOR
2731
2732  017336  112716  000040          8$:     MOVB    #40,(SP)        ;;REPLACE TAB WITH SPACE
2733  017342  004737  017362          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
2734  017346  132737  000007  017426          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2735  017354  001372                          BNE     9$              ;;TAB STOP
2736  017356  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
2737  017360  000726                          BR      2$              ;;GET NEXT CHARACTER
2738  017362  105777  161560          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2739  017366  100375                          BPL     $TYPEC
2740  017370  116677  000002  161552          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2741  017376  122766  000015  000002          CMPB    #15,2(SP)       ;;BRANCH IF
2742  017404  001003                          BNE     1$              ;;NOT <CR>
2743  017406  105037  017426                  CLRB    $CHARCNT        ;;
2744  017412  000406                          BR      $TYPEX          ;;EXIT
2745  017414  122766  000012  000002  1$:     CMPB    #12,2(SP)       ;;BRANCH IF
2746  017422  002002                          BGE     $TYPEX          ;;<LF>
2747  017424  105227                          INCB    (PC)+           ;;INC SPACE
2748  017426  000000          $CHARCNT:.WORD  0                       ;;COUNT
2749  017430  000207          $TYPEX: RTS     PC
2750                                  ;       EQUATES
2751          000011                  TAT=11
2752          000200                  TCRLF=200
2753
2754                                  ;*********************************************************************
2755
2756                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
2757
2758                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
2759                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
2760                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
2761                                  ;*CALL:
2762                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2763                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
2764                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
2765                                  ;*      .BYTE   M               ;;M=1 OR 0
2766                                  ;*                                      ;;1=TYPE LEADING ZEROS
2767                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
2768                                  ;*
2769                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
2770                                  ;*$TYPOS OR $TYPOC
2771                                  ;*CALL:
2772                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2773                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
2774                                  ;*
2775                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
2776                                  ;*CALL:
2777                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2778                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
```

```
2779
2780  017432  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
2781  017436  116637  000001  017655          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
2782  017444  112637  017657                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
2783  017450  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
2784  017454  000406                          BR      $TYPON
2785  017456  112737  000001  017655  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
2786  017464  112737  000006  017657          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
2787  017472  112737  000005  017654  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
2788  017500  010346                          MOV     R3,-(SP)        ;;SAVE R3
2789  017502  010446                          MOV     R4,-(SP)        ;;SAVE R4
2790  017504  010546                          MOV     R5,-(SP)        ;;SAVE R5
2791  017506  113704  017657                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
2792  017512  005404                          NEG     R4
2793  017514  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
2794  017520  110437  017656                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
2795  017524  113704  017655                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
2796  017530  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
2797  017534  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
2798  017536  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
2799  017540  000404                          BR      3$              ;;GO DO MSB
2800  017542  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
2801  017544  006105                          ROL     R5
2802  017546  006105                          ROL     R5
2803  017550  010503                          MOV     R5,R3
2804  017552  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
2805  017554  105337  017656                  DECB    $OMODE          ;;TYPE THIS DIGIT?
2806  017560  100016                          BPL     7$              ;;BR IF NO
2807  017562  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
2808  017566  001002                          BNE     4$              ;;TEST FOR 0
2809  017570  005704                          TST     R4              ;;SUPPRESS THIS 0?
2810  017572  001403                          BEQ     5$              ;;BR IF YES
2811  017574  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2812  017576  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2813  017602  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2814  017606  110337  017652                  MOVB    R3,8$           ;;SAVE FOR TYPING
2815  017612  104400  017652                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
2816  017616  105337  017654          7$:     DECB    $OCNT           ;;COUNT BY 1
2817  017622  003347                          BGT     2$              ;;BR IF MORE TO DO
2818  017624  002402                          BLT     6$              ;;BR IF DONE
2819  017626  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2820  017630  000744                          BR      2$              ;;GO DO THE LAST DIGIT
2821  017632  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
2822  017634  012604                          MOV     (SP)+,R4        ;;RESTORE R4
2823  017636  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2824  017640  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2825  017646  012616                          MOV     (SP)+,(SP)
2826  017650  000002                          RTI                     ;;RETURN
2827  017652     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2828  017653     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2829  017654     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2830  017655     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2831  017656  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2832                                  ;******************************************************************
2833
2834                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
```

```
2835
2836                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
2837                                    ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
2838                                    ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
2839                                    ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
2840                                    ;*REPLACED WITH SPACES.
2841                                    ;*CALL:
2842                                    ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
2843                                    ;*      TYPDS                   ;;GO TO THE ROUTINE
2844
2845  017660                            $TYPDS:
2846  017660  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2847  017662  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2848  017664  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2849  017666  010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2850  017670  010546                           MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2851  017672  012746  020200                   MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
2852  017676  016605  000020                   MOV     20(SP),R5       ;;GET THE INPUT NUMBER
2853  017702  100004                           BPL     1$              ;;BR IF INPUT IS POS.
2854  017704  005405                           NEG     R5              ;;MAKE THE BINARY NUMBER POS.
2855  017706  112766  000055  000001           MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
2856  017714  005000                    1$:    CLR     R0              ;;ZERO THE CONSTANTS INDEX
2857  017716  012703  020074                   MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
2858  017722  112723  000040                   MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
2859  017726  005002                    2$:    CLR     R2              ;;CLEAR THE BCD NUMBER
2860  017730  016001  020064                   MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
2861  017734  160105                    3$:    SUB     R1,R5           ;;FORM THIS BCD DIGIT
2862  017736  002402                           BLT     4$              ;;BR IF DONE
2863  017740  005202                           INC     R2              ;;INCREASE THE BCD DIGIT BY 1
2864  017742  000774                           BR      3$
2865  017744  060105                    4$:    ADD     R1,R5           ;;ADD BACK THE CONSTANT
2866  017746  005702                           TST     R2              ;;CHECK IF BCD DIGIT=0
2867  017750  001002                           BNE     5$              ;;FALL THROUGH IF 0
2868  017752  105716                           TSTB    (SP)            ;;STILL DOING LEADING 0'S?
2869  017754  100407                           BMI     7$              ;;BR IF YES
2870  017756  106316                    5$:    ASLB    (SP)            ;;MSD?
2871  017760  103003                           BCC     6$              ;;BR IF NO
2872  017762  116663  000001  177777           MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
2873  017770  052702  000060            6$:    BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
2874  017774  052702  000040            7$:    BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
2875  020000  110223                           MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
2876  020002  005720                           TST     (R0)+           ;;JUST INCREMENTING
2877  020004  020027  000010                   CMP     R0,#10          ;;CHECK THE TABLE INDEX
2878  020010  002746                           BLT     2$              ;;GO DO THE NEXT DIGIT
2879  020012  003002                           BGT     8$              ;;GO TO EXIT
2880  020014  010502                           MOV     R5,R2           ;;GET THE LSD
2881  020016  000764                           BR      6$              ;;GO CHANGE TO ASCII
2882  020020  105726                    8$:    TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
2883  020022  100003                           BPL     9$              ;;BR IF NO
2884  020024  116663  177777  177776           MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
2885  020032  105013                    9$:    CLRB    (R3)            ;;SET THE TERMINATOR
2886  020034  012605                           MOV     (SP)+,R5        ;;POP STACK INTO R5
2887  020036  012603                           MOV     (SP)+,R3        ;;POP STACK INTO R3
2888  020040  012602                           MOV     (SP)+,R2        ;;POP STACK INTO R2
2889  020042  012601                           MOV     (SP)+,R1        ;;POP STACK INTO R1
2890  020044  012600                           MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
2891  020046  104400  020074                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
2892  020052  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
2893  020060  012616                          MOV     (SP)+,(SP)
2894  020062  000002                          RTI                     ;;RETURN TO USER
2895  020064  023420                  $DTBL:  10000.
2896  020066  001750                          1000.
2897  020070  000144                          100.
2898  020072  000012                          10.
2899  020074  000004                  $DBLK:  .BLKW   4
2900                                  ;*****************************************************************
2901
2902                                  .SBTTL  TRAP DECODER
2903
2904                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2905                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2906                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2907                                  ;*GO TO THAT ROUTINE.
2908
2909  020104  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2910  020106  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
2911  020112  005740                          TST     -(R0)           ;;BACKUP BY 2
2912  020114  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2913  020116  006300                          ASL     R0              ;;POSITION FOR INDEXING
2914  020120  016000  020126                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2915  020124  000200                          RTS     R0              ;;GO TO ROUTINE
2916
2917
2918                                  .SBTTL  TRAP TABLE
2919
2920                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2921                                  ;*BY THE "TRAP" INSTRUCTION.
2922
2923                                  ;       ROUTINE
2924                                  ;       -------
2925  020126                          $TRPAD:
2926  020126  017216                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
2927  020130  017456                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2928  020132  017432                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2929  020134  017472                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2930  020136  017660                          $TYPDS  ;;CALL=TYPDS    TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)
2931  020140  016420                          $RDCHR  ;;CALL=RDCHR    TRAP+5(104405)  TTY TYPEIN CHARACTER ROUTINE
2932  020142  016454                          $RDLIN  ;;CALL=RDLIN    TRAP+6(104406)  TTY TYPEIN STRING ROUTINE
2933  020144  017122                          $SAVREG ;;CALL=SAVREG   TRAP+7(104407)  SAVE R0-R5 ROUTINE
2934  020146  017160                          $RESREG ;;CALL=RESREG   TRAP+10(104410) RESTORE R0-R5 ROUTINE
2935                                  ;*****************************************************************
2936
2937                                  .SBTTL  POWER DOWN AND UP ROUTINES
2938
2939                                  ;POWER DOWN ROUTINE
2940  020150  012737  020272  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
2941  020156  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2942  020164  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2943  020166  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2944  020170  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2945  020172  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2946  020174  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
```

```
2947  020176  010546                          MOV     R5,-(SP)          ;;PUSH R5 ON STACK
2948  020200  010637  020276                  MOV     SP,$SAVR6         ;;SAVE SP
2949  020204  012737  020216  000024          MOV     #$PWRUP,@#PWRVEC  ;;SET UP VECTOR
2950  020212  000000                          HALT
2951  020214  000776                          BR      .-2               ;;HANG UP
2952
2953                                  ;POWER UP ROUTINE
2954  020216  013706  020276          $PWRUP: MOV     $SAVR6,SP         ;;GET SP
2955  020222  005037  020276                  CLR     $SAVR6            ;;WAIT LOOP FOR THE TTY
2956  020226  005237  020276          1$:     INC     $SAVR6            ;;WAIT FOR THE INC
2957  020232  001375                          BNE     1$                ;;OF  WORD
2958  020234  012605                          MOV     (SP)+,R5          ;;POP STACK INTO R5
2959  020236  012604                          MOV     (SP)+,R4          ;;POP STACK INTO R4
2960  020240  012603                          MOV     (SP)+,R3          ;;POP STACK INTO R3
2961  020242  012602                          MOV     (SP)+,R2          ;;POP STACK INTO R2
2962  020244  012601                          MOV     (SP)+,R1          ;;POP STACK INTO R1
2963  020246  012600                          MOV     (SP)+,R0          ;;POP STACK INTO R0
2964  020250  012737  020150  000024          MOV     #$PWRDN,@#PWRVEC  ;;SET UP THE POWER DOWN VECTOR
2965  020256  012737  000340  000026          MOV     #340,@#PWRVEC+2   ;;PRIO:7
2966  020264  104400                          TYPE                      ;REPORT THE POWER FAILURE
2967  020266  020300                  $PWRMG: .WORD   $POWER            ;;POWER FAIL MESSAGE POINTER
2968  020270  000002                          RTI
2969  020272  000000                  $ILLUP: HALT                      ;;THE POWER UP SEQUENCE WAS STARTED
2970  020274  000776                          BR      .-2               ;; BEFORE THE POWER DOWN WAS COMPLETE
2971  020276  000000                  $SAVR6: 0                         ;;PUT THE SP HERE
2972  020300  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
2973  020306  000122
2974                                          .EVEN
2975                                  ;*****************************************************************
2976                                  ;*****************************************************************
2977                                  ;*****************************************************************
2978  020310                          ATEND:
2979          000001                          .END
```

```
ADD20   002644          1127    1136#
ADLI    011064          1604    1606    1609    1612    2272#
ALLERR  001574           981#   1470
ATEND   020310          1287    1320    1354    1388    1422    1451    1491    1498    1692    1788    1797    1806    1814
                        1835    1858    1953    1957    1968    1977    1987    2013    2047    2066    2209    2290    2292
                        2978#
BE1BA   001604           987#   1287*   1320*   1354*   1388*   1422*   1451*   1498*   1626    1632*   1756    1797*   1922
                        1957*   2209*   2226    2276#
BE1CC   001602           986#   1288*   1292    1295    1321*   1325    1328    1355*   1359    1362    1389*   1393    1396
                        1423*   1427    1430    1450*   1497*   1693*   1798*   1958*   2210*   2213    2225    2275*
BE1CLR  001610           989#   1621*
BE1CR1  001606           988#   1263*   1264*   1289*   1294    1322*   1327    1356*   1361    1390*   1395    1424*   1429
                        1457*   1463    1500*   1503    1533*   1624    1696*   1799*   1826    1925    1962*   2211*   2227
                        2277*
BE1CR2  001612           990#   1456*   1460*   1461    1464    1470    1499*   1505*   1512    1605*   1607*   1608*   1610*
                        1619    1625    1653*   1656*   1669*   1694*   1695    1713*   1961*   2012*   2119    2134
BE1DB   001600           985#   1028    1096    1155    1208    1506    1508    1567    2224
BE1PSW  001662          1010#   1249*   1286*   1319*   1353*   1387*   1421*   1455*   1496*   1532*   1602*   1650*   1796*
                        1960*
BE1VEC  001660          1009#   1097    1248*   1285*   1318*   1352*   1386*   1420*   1454*   1495*   1531*   1601*   1649*
                        1689*   1795*   1959*
BE2BA   001620           993#   1761    1806*   1929    1968*
BE2CC   001616           992#   1805*   1969*
BE2CLR  001624           995#
BE2CR1  001622           994#   1538*   1807*   1932    1970*
BE2CR2  001626           996#
BE2DB   001614           991#   1572
BE2PSW  001666          1012#   1537*   1804*   1967*
BE2VEC  001664          1011#   1536*   1803*   1966*
BE3BA   001634           999#   1766    1814*   1977*
BE3CC   001632           998#   1813*   1978*
BE3CLR  001640          1001#
BE3CR1  001636          1000#   1543*   1815*   1877    1937    1980*
BE3CR2  001642          1002#   1979*   2046*
BE3DB   001630           997#   1577    1881    1883    1975*
BE3PSW  001672          1014#   1542*   1812*   1974*
BE3VEC  001670          1013#   1541*   1811*   1976*
BE4BA   001650          1005#   1771    1987*
BE4CC   001646          1004#   1822*   1988*
BE4CLR  001654          1007#
BE4CR1  001652          1006#   1549*   1824*   1897    1943    1989*
BE4CR2  001656          1008#   1823*   1905*
BE4DB   001644          1003#   1582    1984*
BE4PSW  001676          1016#   1548*   1821*   1985*
BE4VEC  001674          1015#   1547*   1820*   1986*
BGIN    002736          1101    1128    1168#
BIT0  = 000001           634#
BIT00 = 000001           624#    634     2187    2195
BIT01 = 000002           623#    633
BIT02 = 000004           622#    632
BIT03 = 000010           621#    631     1456    1460
BIT04 = 000020           620#    630     1653    1656    1669    2122
BIT05 = 000040           619#    629     1264    2152
BIT06 = 000100           618#    628     2156
BIT07 = 000200           617#    627     1069    1080    1175
BIT08 = 000400           616#    626     2160    2382
```

```
BIT09 = 001000           615*   625    2164   2390   2452
BIT1  = 000002           633*
BIT10 = 002000           614*  2168    2434
BIT11 = 004000           613*  1263    1512   2148   2397
BIT12 = 010000           612*  1694
BIT13 = 020000           611*  2441
BIT14 = 040000           610*  1499    1505   1823   1905   1961   1979   2012   2046   2368
BIT15 = 100000           609*
BIT2  = 000004           632*
BIT3  = 000010           631*
BIT4  = 000020           630*
BIT5  = 000040           629*
BIT6  = 000100           628*
BIT7  = 000200           627*
BIT8  = 000400           626*
BIT9  = 001000           625*
BKEX    011016          2247   2250*
BKGO    010744          1827   2233*
BPTVEC= 000014           641*
BRK     005604          1601   1616*
BR4TST  004436          1414*
BR5TST  004276          1380*
BR6TST  004136          1346*
BR7TST  003776          1312*
CHEX    007456          1927   1934    1939   1945*
CHKERR  010422          2141*
CLRREG  002700          1152*  1171    1245   1449   1490   1528   1599   1727   1792
CLRRTN  001724          1027*  1091
CMPARE  005340          1555*
CNTR    011036          1459   1554    1654   2264*  2278
DATA1   001712          1020*  1756*   1757*  1837   1922*  1923*  2017   2021
DATA2   001714          1021*  1761*   1762*  1859   1929*  1930*
DATA3   001716          1022*  1766*   1767*  2048
DATA4   001720          1023*  1771*   1772*  2067
DDISP = 177570           556*   776
DEVCNT  001700          1017*  1106*   1129   1162   1195   1229   1232   1534   1539   1544   1555   1731   1800
                        1808   1816    1849   1868   1887   1963   1971   1981   2001   2035   2037   2057   2089
                        2125
DEVS    001702          1018*  1530    1557   1558   1559   1560
DH1     011442           826    975    2298*
DH10    012247           862   2298*
DH11    012473           868   2298*
DH12    012632           873   2298*
DH13    012756           878    883    2298*
DH15    013225           889    895     900    905    910   2298*
DH2     011606           832   2298*
DH22    013775           915    920     925    930    935    940    945   2298*
DH3     011730           837    842     847    852    857   2298*
DH31    014550           951    958     963    969   2298*
DISPLA  001140           776*  1065*   2411*  2433*
DISPRE  000174           715*  1065
DOUP    011020          1493   1790    1955   2253*
DC14    005532          1603*
DSWR  = 177570           555*   775
DT1     015252           827    976    2298*
DT10    015320           864   2298*
```

```
DT11    015336             869    2298#
DT12    015352             874    2298#
DT13    015364             879     884    2298#
DT15    015372             891     896     901     906     911    2298#
DT2     015262             833    2298#
DT22    015410             916     921     926     931     936     941     946    2298#
DT3     015302             838     843     848     853     858    2298#
DT31    015422             953     959     964     970    2298#
EMTVEC= 000030             644#   1047*   1048*
EM1     011374             825    2298#
EM10    012201             861    2298#
EM11    012405             867    2298#
EM12    012544             872    2298#
EM13    012673             877    2298#
EM14    012777             882    2298#
EM15    013062             887    2298#
EM16    013337             894    2298#
EM17    013425             899    2298#
EM2     011473             830    2298#
EM20    013513             904    2298#
EM21    013607             909    2298#
EM22    013703             914    2298#
EM23    014037             919    2298#
EM24    014110             924    2298#
EM25    014156             929    2298#
EM26    014210             934    2298#
EM27    014252             939    2298#
EM3     011676             836    2298#
EM30    014321             944    2298#
EM31    014410             949    2298#
EM32    014643             956    2298#
EM33    014727             962    2298#
EM34    015001             967    2298#
EM35    015155             973    2298#
EM4     012011             841    2298#
EM5     012047             846    2298#
EM6     012105             851    2298#
EM7     012143             856    2298#
ENDMEM  001722            1024#   1030
ERRCHK  004730            1248    1285    1318    1352    1386    1420    1454    1469#   1495    1649
ERRS    010704            2220    2223#
ERRTN   010376            1474    1586    1628    1708    1777    1947    2112    2132#
ERRVEC= 000004             637#   1093*   1103*   1105*   1250*   1453*   2373    2374*   2376*   2379*   2577    2578    2583*
                          2591*   2605*   2614*   2624*   2625*
EXAD    011130            2280    2283#
EXBRK   005664            1622    1629#
EXNO    002660            1130    1139#
EXNOG   010702            2218    2221#
EXS     007466            1926    1933    1938    1944    1948#
FAILCK  006004            1657#
FOR4    011267            1234    2298#
GNS   = ****** U           714    1084    1088    1133    1189    1193    1201    1205    1214    1222    2325    2332    2926
                          2927    2928    2929    2930    2931    2932    2933    2934
GO      005320            1535    1540    1545    1550#
INK     006530            1759    1764    1769    1774#
INTRTN  005470            1569    1574    1579    1584#
```

```
INTR1   005402          1531    1565*
INTR2   005420          1536    1570*
INTR3   005436          1541    1575*
INTR4   005454          1547    1580*
IOTVEC= 000020           642*   1045*   1046*
KIPAR0= 172340           698*   2563    2595
KIPAR1= 172342           699*
KIPAR2= 172344           700*
KIPAR3= 172346           701*
KIPAR4= 172350           702*
KIPAR5= 172352           703*
KIPAR6= 172354           704*
KIPAR7= 172356           705*   2608
KIPDR0= 172300           687*
KIPDR1= 172302           688*
KIPDR2= 172304           689*
KIPDR3= 172306           690*
KIPDR4= 172310           691*
KIPDR5= 172312           692*
KIPDR6= 172314           693*
KIPDR7= 172316           694*
LEEV    010536          2147    2169    2171*
LOAD1   005154          1529*
LOAD2   005212          1536*
LOAD3   005244          1541*
LOAD4   005276          1546*
LODDEV  002426          1098*   1122    1138
MMVEC = 000250           654*
MOVREG  002462          1107*   1112
MOVVEC  002526          1117*
MULT1   007470          1730    1739    1748    1952*
NG      003514          1246*
NODEV   002544          1103    1125*
NODO    006260          1681    1716*
NOG     010626          1253    1256    1259    1262    2206*
NPRTST  003636          1279*
NXT     006172          1687    1698*   1715
PBERR   006224          1689    1704*
PBPSW = 000116           984*   1691*   1699    1700*
PBRTN   006246          1690    1712*
PBVEC = 000114           983*   1690*   1699*
PC    =%000007           568*   1036*   1091*   1092*   1164*   1170*   1171*   1245*   1253*   1256*   1259*   1262*   1449*
                        1459*   1474*   1490*   1493*   1528*   1554*   1586*   1599*   1604*   1606*   1609*   1612*   1629*
                        1654*   1708*   1727*   1730*   1738*   1739*   1747*   1748*   1777*   1790*   1791*   1792*   1827*
                        1947*   1955*   1956*   2076*   2091*   2112*   2175*   2203*   2220*   2221*   2230*   2250*   2260*
                        2269*   2278*   2284*   2294*   2317*   2320*   2344*   2443*   2498*   2588    2632*   2719*   2726*
                        2733*   2747*   2749*
PIRQ  = 177772           554*
PIRQVE= 000240           648*
PR0   = 000000           571*   1301    1334    1368    1402    1436    1501    1553    1794
PR1   = 000040           572*
PR2   = 000100           573*
PR3   = 000140           574*   1415    1458    1600    1688    2288
PR4   = 000200           575*   1260    1381    1419    1431
PR5   = 000240           576*   1257    1347    1385    1397
PR6   = 000300           577*   1254    1313    1351    1363
```

```
PR7     = 000340          578* 1079 1093 1249 1251 1280 1284 1286 1296 1317 1319 1329 1353
                         1387 1421 1452 1455 1489 1496 1527 1532 1537 1542 1548 1602 1648
                         1650 1682 1796 1804 1912 1821 1960 1967 1974 1985 2086
PS      = 177776          551#  552
PSW     = 177776          552# 1284* 1290* 1317* 1323* 1351* 1357* 1385* 1391* 1419* 1425* 1452* 1458*
                         1489* 1501* 1527* 1553* 1600* 1648* 1688* 1794* 2208* 2298*
PWRFAL  010332           1169 1655 2116#
PWRVEC= 000024            643# 1051* 1052* 1169* 1652* 1655* 2940* 2941* 2949* 2964* 2965*
QNO     011164           1071 2298#
Q1      010266           2082 2100#
Q2      010276           2103#
Q3      010306           2106#
Q4      010316           2109#
RDCHR = 104405           2536 2931#
RDLIN = 104406           2932#
RESREG= 104410           2174 2934#
RESVEC= 000010            638#
ROTATE  010570           1738 1747 2186#
R0      =%000000          559# 1094* 1099 1104 1107 1108 1109* 1114* 1115 1116* 1126 1136* 1153*
                         1156* 1158 1160* 1208* 1211 1228* 1251* 1254* 1257* 1260* 1280* 1290 1297
                         1300* 1301 1313* 1323 1330 1333* 1334 1347* 1357 1364 1367* 1368 1381*
                         1391 1398 1401* 1402 1415* 1425 1432 1435* 1436 1826* 1912* 1913 2013*
                         2015 2017 2019 2021 2025 2028 2030 2031 2047* 2048 2050 2053 2054
                         2066* 2067 2069 2072 2073 2081* 2088* 2089 2118 2119* 2122* 2123* 2127*
                         2134* 2138* 2142 2146 2148 2152 2156 2160 2164 2168 2172* 2173* 2208
                         2228 2239 2273* 2276 2279 2281* 2290* 2291* 2292 2341* 2344 2470 2471*
                         2472* 2479* 2480* 2481* 2482* 2483* 2484 2489 2494* 2496* 2500 2502 2573
                         2579* 2623 2631* 2654 2679* 2703 2704* 2705 2708* 2846 2856* 2860 2876
                         2877 2890* 2909 2910* 2911 2912* 2913* 2914* 2915* 2942 2963*
R1      =%000001          560# 1096* 1107* 1115* 1155* 1157* 1252* 1255* 1258* 1261* 1551* 1555 1566*
                         1571* 1576* 1581* 1858* 1859 1861 1863 1864 1998* 1999* 2000* 2082* 2085
                         2087* 2117 2120* 2124* 2125 2128* 2211 2215* 2216 2235* 2236* 2237* 2238*
                         2241* 2245 2246 2254* 2256 2257* 2258 2574 2587* 2616* 2618 2619 2621*
                         2626 2630* 2655 2678* 2847 2860* 2861 2865 2889* 2943 2962*
R2      =%000002          561# 1095* 1118 1119* 1120 1121* 1137* 1154* 1161* 1162 1491* 1788* 1835*
                         1837 1839 1844 1845 1953* 2083* 2085* 2086* 2256* 2575 2595* 2597* 2598*
                         2601* 2602* 2607* 2608 2610* 2615* 2617* 2622* 2627 2629* 2656 2677* 2848
                         2859* 2863* 2866 2873* 2874* 2875 2880* 2888* 2944 2961*
R3      =%000003          562# 1028* 1029* 1030 1032* 1033* 1034 1097* 1118* 1120* 1728* 1733* 1740
                         1742* 1749 1975 2050 2052 2187 2194* 2532 2533* 2534 2537* 2538 2542
                         2544 2546* 2548* 2576 2596* 2600* 2628* 2657 2676* 2788 2797* 2803* 2804*
                         2807* 2812* 2813* 2814 2823* 2849 2857* 2858* 2972* 2875* 2884* 2885* 2887*
                         2945 2960*
R4      =%000004          563# 1530* 1567* 1572* 1577* 1582* 1729* 1734* 1743* 1984 2069 2071 2195
                         2202* 2658 2675* 2789 2791* 2792* 2793* 2794 2795* 2809 2811* 2819* 2822*
                         2946 2959*
R5      =%000005          564# 1492* 1643* 1645* 1789* 1954* 2135* 2137* 2139 2258 2659 2674* 2790
                         2796* 2798* 2800* 2801* 2802* 2803 2821* 2850 2852* 2854* 2861* 2865* 2880
                         2886* 2947 2958*
R6      =%000006          565#  567 1040* 1041* 1042
R7      =%000007          566#  568
SAVREG= 104407           2133 2933#
SERV1   006412           1755# 1959
SERV2   006436           1760# 1966
SERV3   006462           1765# 1976
SERV4   006506           1770# 1986
```

```
SIMLGO  001576          982#   1552*  1991*
SP    =%000006          567#   1044*  1059*  1060*  1066   1067   1068   1128*  1138*  1145   1184*  1195*  1216*
                       1224*   1472   1585   1617   1705   1715*  1776   1946   2117*  2118*  2127   2128   2180
                       2327*  2334*  2373*  2376   2378   2379   2407   2408   2412*  2438   2454*  2457*  2470*
                       2475*  2496   2500*  2518*  2519*  2522*  2523*  2532*  2537   2548   2549*  2550*  2551*
                       2573*  2574*  2575*  2576*  2577*  2578*  2579   2580*  2583   2594*  2586   2594*  2598
                       2599*  2623*  2624   2625   2628   2629   2630   2631   2654*  2655*  2656*  2657*  2658*
                       2659*  2660*  2661*  2662*  2663*  2670*  2671*  2672*  2673*  2674   2675   2676   2677
                       2678   2679   2703*  2704   2705*  2707   2708   2709*  2711   2713   2715   2720   2722*
                       2724*  2732*  2736   2740   2741   2745   2780*  2781   2782   2783*  2788*  2789*  2790*
                       2796   2821   2822   2823   2824*  2825*  2846*  2847*  2848*  2849*  2850*  2851*  2952
                       2855*  2868   2870*  2872   2882   2884   2886   2887   2888   2889   2890   2892*  2893*
                       2909*  2910   2942*  2943*  2944*  2945*  2946*  2947*  2948   2954*  2958   2959   2960
                       2961   2962   2963
SR0   = 177572          658#   2592   2604*  2611*
SR1   = 177574          659#
SR2   = 177576          660#
SR3   = 172516          661#   2603*
STACK = 001100          548#   1044
START   001756          719     736   1039#
STKLMT= 177774          553#
STVEC   010230         1170   2080#
SWR     001136          775#   1062   1064*  1069   1080   1175   2368   2382   2384   2390   2397   2434   2441
                       2446   2452
SWREG   000176          716#   1064
SW0   = 000001          606#
SW00  = 000001          596#    606
SW01  = 000002          595#    605
SW02  = 000004          594#    604
SW03  = 000010          593#    603
SW04  = 000020          592#    602
SW05  = 000040          591#    601
SW06  = 000100          590#    600
SW07  = 000200          589#    599
SW08  = 000400          588#    598
SW09  = 001000          587#    597
SW1   = 000002          605#
SW10  = 002000          586#
SW11  = 004000          585#
SW12  = 010000          584#
SW13  = 020000          583#
SW14  = 040000          582#
SW15  = 100000          581#
SW2   = 000004          604#
SW3   = 000010          603#
SW4   = 000020          602#
SW5   = 000040          601#
SW6   = 000100          600#
SW7   = 000200          599#
SW8   = 000400          598#
SW9   = 001000          597#
S1      007334         1795   1920#
S2      007366         1803   1928#
S3      007420         1811   1935#
S4      007436         1820   1940#
TBITVE= 000014          639#
```

```
TCRLF = 000200          2713    2752*
THRUO   010552          1078    2179*
THT   = 000011          2711    2751*
TKVEC = 000060           646*
TMPPWR  006226          1652    1667*
TPVEC = 000064           647*
TRAPVE= 000034           645*   1049*   1050*
TRTVEC= 000014           640*
TSTOVR  011132          1791    1956    2287*
TST1    003506          1075    1244*   2349
TST10   004760          1462    1466    1487*
TST11   005140          1513    1515    1526*
TST12   005502          1556    1562    1597*
TST13   005666          1613    1639*
TST14   006046          1642    1664    1680*
TST15   006262          1703    1726*
TST16   006546          1732    1737    1746    1751    1787*
TST17   007304          1850    1869    1888    1909*
TST2    003634          1277*
TST3    003774          1283    1311*
TST4    004134          1316    1345*
TST5    004274          1350    1379*
TST6    004434          1384    1413*
TST7    004574          1418    1448*
TYMOUT  002662          1105    1143*   1250    1453
TYPDS = 104404          1186    2329    2336    2930*
TYPE  = 104400          1071    1082    1086    1131    1183    1187    1191    1199    1203    1212    1220    1231    1234
                        2323    2330    2337    2436    2444    2469    2486    2488    2491    2493    2497    2504    2540
                        2543    2547    2716    2815    2891    2926*   2966
TYPOC = 104401          2477    2501    2927*
TYPON = 104403          2929*
TYPOS = 104402          1196    1217    1225    2928*
UIPAR0= 177640           676*
UIPAR1= 177642           677*
UIPAR2= 177644           678*
UIPAR3= 177646           679*
UIPAR4= 177650           680*
UIPAR5= 177652           681*
UIPAR6= 177654           682*
UIPAR7= 177656           683*
UIPDR0= 177600           665*
UIPDR1= 177602           666*
UIPDR2= 177604           667*
UIPDR3= 177606           668*
UIPDR4= 177610           669*
UIPDR5= 177612           670*
UIPDR6= 177614           671*
UIPDR7= 177616           672*
XQ      010324          2102    2105    2108    2111*
XTST    006016          1660    1662*
$BDADR  001122           769*
$BDDAT  001126           771*   1042
$BELL   001214           801*   2436    2460
$CHARC  017426          2727*   2734    2743*   2748*
$CMTAG  001100           757*   1040    1047    1053    1054    1055
$CM1  = 000006           787*    788*    789*    790*    791*    792*    793*
```

```
$CM2  = 000014            787*   788*   789*   790*   791*   792*   793*
$CM3  = 000006            795*   787
$CM4  = 000006            793*   794*   795*   796*   797*   798*   799*
$CORE   017022           2590   2614*
$CRLF   001221            803*  1183    1231   1658   1668   2337   2444   2460   2469   2488   2493   2497   2556
                         2717   2754
$CROUT  017052           2614   2621*
$DBLK   020074           2857   2891   2899*
$DORGN  015636           2319   2342   2348*
$DTBL   020064           2860   2895*
$ENDAD  015626            744   2344*  2449
$ENDCT  015472           1053   2321*
$ENULL  015642           2350*
$EOP    015436           1917   2311*
$EOPCT  015464           1053*  2318*  2322
$ERFLG  001103            760*  1282   1315   1349   1383   1417   1736   1745   2358   2386   2388   2394*  2415
                         2431*  2460
$ERMAX  001115            766*  1056*  1282   1315   1349   1383   1417   1736   1745   2388   2410*  2415
$ERROR  016116           1047   2430*
$ERRPC  001116            767*  2298   2438*  2439*  2440   2460   2475
$ERRTB  001224            821*  2483
$ERRTY  016264           2443   2468*
$ERTTL  001112            764*  2334   2338*  2437*  2460
$ESCAP  001212            800*  1055*  2409*  2455   2457   2460
$FILLC  001154            783*  2720   2754
$FILLS  001153            782*  2754
$GDADR  001120            768*
$GDDAT  001124            770*
$GET42  015616           2339*
$HD   = 000001             531    532
$ICNT   001104            761*  2298   2401*  2402   2404*  2414
$ILLUP  020272           2940   2969*
$ITEMB  001114            765*  2440*  2460   2472
$KTNEX  017014           2591   2613*
$KTOUT  017004           2605   2610*
$KTII   016664           2589*  2593*  2613*
$LF     001222            804*  2460   2547   2556   2754
$LPADR  001106            762*  1057*  2392*  2407*  2412   2414
$LPERR  001110            763*  1058*  2392   2408*  2414   2454
$LSTAD  017116           2626*  2633*
$LSTBK  017120           1174   2627*  2634*
$MAIL = ****** U         1069   1245   1278   1312   1346   1380   1414   1449   1488   1527   1598   1641   1681
                         1727   1788   1911   2407   2446   2705
$MXCNT  016114           2405   2414*
$NULL   001152            781*  2722   2754
$NWTST= 000001           1238*  1240   1268*  1270   1304*  1306   1338*  1340   1372*  1374   1406*  1408   1440*
                         1442   1479*  1481   1518*  1520   1590*  1592   1632*  1634   1673*  1675   1719*  1721
                         1781*  1783   1906*
$OCNT   017654           2787*  2816*  2829*
$OMODE  017656           2782*  2786*  2791   2794*  2805*  2831*
$OVER   016100           2369   2385   2393   2403   2411*
$PASS   001100            758*  2315*  2316*  2327   2350   2399   2415
$POWER  020300           1663   2967   2972*
$PWRDN  020150           1051   1670   2129   2940*  2964
$PWRMG  020266           1658   1663*  1668*  2967*
$PWRUP  020216           2949   2954*
```

```
$QUES   001220            802*  2460  2540  2555  2754
$RDCHR  016420           2518*  2931
$RDDEC= ****** U         2933
$RDLIN  016454           2532*  2932
$RDOCT= ****** U         2933
$RDSZ = 000016           2525*
$REGAD  001156            795*
$REGO   001160            787*  1032  1173*  1178*  1180  1211*  1224  1294*  1327*  1361*  1395*  1429*  1463*
                         1845*  1864*  2031*  2054*  2073*  2224*  2298
$REG1   001162            788*  1295*  1328*  1362*  1396*  1430*  1464*  1508*  1557*  1624*  1844*  1863*  1883*
                         1901*  1997*  1998  2030*  2053*  2072*  2225*  2234*  2235  2298
$REG2   001164            789*  1145*  1146*  1296*  1329*  1363*  1397*  1431*  1509*  1558*  1625*  1775*  1836*
                         1839  1841*  1846  1884*  1902*  2001  2180*  2181*  2226*  2248*  2289*  2298
$REG3   001166            790*  1297*  1330*  1364*  1398*  1432*  1559*  1626*  1846*  1865*  2032*  2052*  2071*
                         2227*  2298
$REG4   001170            791*  1560*  1819*  1899  1901  1941*  2228*  2265*  2266*  2267  2298
$REG5   001172            792*  1108*  1472*  1585*  1617*  1705*  1776*  1946*  2298
$RESRE  017160           2669*  2934
$R2A  = ****** U         2935
$SAVRE  017122           2653*  2933
$SAVR6  020276           2948*  2954  2955*  2956*  2971*
$SCOPE  015646           1045  2367*
$SETUP= 000037            707*  1045  1047  1049  1051  1053  1054  1055  1057  2313  2449
$SIZE   016600           1092  2573*
$SIZEX  017056           2612  2622*
$SIZE1  016652           2584  2586*
$STUP = 177777            707*
$SVLAD  016052           2377  2406*
$SVPC = 000232            742*  747
$SWR  = 167400            520*  531  537  538  539  540  541  542  543  799  800  801  1054
                         1055  1057  1058  1245  1278  1312  1346  1380  1414  1449  1488  1527  1599
                         1640  1681  1727  1788  1910  2308  2314  2340  2349  2350  2359  2360  2361
                         2362  2363  2368  2380  2382  2383  2386  2387  2388  2395  2396  2397  2408
                         2411  2414  2422  2423  2424  2425  2426  2434  2441  2446  2452  2460  2968
$SWRMK= 000000            543  544  2363  2364  2384
$TIMES  001210            799*  1054*  1640*  1910*  2314*  2395*  2402  2405*  2414
$TKB    001144           1778*  1912  2512  2522
$TKS    001142           1777*  2512  2520
$TMP0   001174            718*  735*  793*  1073  1110*  1111  1113*  1172*  1174*  1179*  1182*  1184  1207*
                         1210*  1216  1229
$TMP1   001176            794*  1651*
$TMP2   001200            795*
$TMP3   001202            796*  1683*  1685*  1686
$TMP4   001204            797*  1473*  1568*  1573*  1578*  1583*  1618*  1647*  1707*  1758*  1763*  1768*  1773*
                         1924*  1931*  1936*  1942*  2101*  2104*  2107*  2110*  2139  2298
$TMP5   001206            798*  1034
$TN   = 000020            531*  1238  1245*  1268  1278*  1283  1304  1312*  1316  1338  1346*  1350  1372
                         1380*  1384  1406  1414*  1418  1440  1449*  1462  1466  1479  1488*  1513  1515
                         1518  1527*  1556  1562  1590  1598*  1613  1632  1640*  1642  1664  1673  1681*
                         1703  1719  1727*  1732  1737  1746  1751  1781  1788*  1850  1869  1888  1906
                         1910*
$TPB    001150            780*  2740*  2754
$TPFLG  001155            784*  2699  2754
$TPS    001146            779*  2738  2754
$TRAP   020104           1049  2909*
$TRP  = 000011           2917*  2927*  2928*  2929*  2930*  2931*  2932*  2933*  2934*  2935*
```

```
$TRPAD  020126            2914    2925*
$TSTNM  001102             759*   2298    2313*   2358    2384    2406*   2411    2415    2433    2460
$TTYIN  016562            2533    2534    2551    2555*
$TYPBN= ****** U          2931
$TYPDS  017660            2845*   2930
$TYPE   017216            2699*   2917    2926
$TYPEC  017362            2719    2726    2733    2738*   2739
$TYPEX  017430            2744    2746    2749*
$TYPOC  017456            2785*   2927
$TYPON  017472            2784    2787*   2929
$TYPOS  017432            2780*   2928
$XTSTR  015656            2371*
$ZFILL  017555            2781*   2785*   2795    2830*
$40CAT= ****** U          2368    2443
.       = 020310           710*    714*    717*    734*    742     743*    745*    747*    756*    805     1043    1057    1058
                          1082    1085*   1086    1131    1134*   1187    1191    1199    1202*   1203    1206*   1212    1215*
                          1220    2298*   2323    2326*   2330    2350    2351*   2414    2415    2460    2507*   2512    2555*
                          2556    2754    2899*   2951    2970
```

```
BE1          1278#  1281   1314   1348   1382   1416
COMMEN          1#   649#    721   1828   1851   1870   1889   1992   2004   2039   2053
ENDCOM          1#   649#    730   1832   1855   1874   1893   1996   2009   2043   2063
ERROR         549#  1147   1298   1331   1365   1399   1433   1465   1510   1514   1561   1627   1661   1706   1947
              1866  1885   1903   2033   2055   2074   2144   2150   2154   2158   2162   2166   2170   2182   2229
              2249
ESCAPE          1#   649#
GETPRI          1#
MSSG         1237#  1240   1268#  1270   1303#  1306   1337#  1340   1371#  1374   1405#  1408   1439#  1442   1478#
              1481  1517#  1520   1589#  1592   1631#  1634   1672#  1675   1718#  1721   1781#  1783
MULT            1#   649#
NEWTST          1#   649#   1238   1268   1304   1338   1372   1406   1440   1479   1518   1590   1632   1673   1719
              1791  1906
NOGRNT       1267#  2207
POP             1#   649#   2674   2886   2958
PUSH            1#   649#   2654   2845   2942
REPORT          1#
SCOPE         550#  1244   1277   1311   1345   1379   1413   1448   1487   1526   1597   1639   1680   1726   1787
              1909  2312
SETPRI          1#
SETTRA       2917#  2927   2928   2929   2930   2931   2932   2933   2934
SETUP           1#   649#  1040
SKIP            1#   649#  1283   1316   1350   1384   1418   1462   1466   1513   1515   1556   1562   1613   1642
              1664  1703   1732   1737   1746   1751   1850   1869   1888
SLASH           1#   649#  1165   2077
SPACE         649#
STARS           1#   649#   738    749    805    822    978   1025   1037   1123   1141   1149   1238   1243   1268
              1276  1304   1310   1338   1344   1372   1378   1406   1412   1438   1440   1447   1467   1479   1486
              1518  1525   1563   1590   1596   1614   1632   1638   1665   1673   1679   1710   1719   1725   1752
              1779  1781   1786   1906   1908   1918   1950   2003   2092   2114   2130   2176   2184   2204   2222
              2231  2232   2251   2261   2270   2285   2295   2298   2301   2352   2415   2460   2508   2525   2556
              2635  2681   2754   2832   2900   2935   2975
SWRSU           1#  1059#
TRMTRP       2917#
TYPBIN          1#   649#
TYPDEC          1#   649#  1184   2327   2334
TYPNHM          1#
TYPNUM          1#   649#
TYPOCS          1#   649#  1195   1216   1224
TYPOCT          1#   649#  2475   2499
TYPTXT          1#   649#  1082   1086   1131   1187   1191   1199   1203   1212   1220   2323   2330
$$CMRE       749#   787    788    789    790    791    792
$$CMTM       749#   793    794    795    796    797    798
$$ESCA          1#   649#
$$NEWT          1#   649#  1238   1268   1304   1338   1372   1406   1440   1479   1518   1590   1632   1673   1719
              1781  1906
$$SET        2917#  2927   2928   2929   2930   2931   2932   2933   2934
$$SKIP          1#   649#  1283   1316   1350   1384   1418   1462   1466   1513   1515   1556   1562   1613   1642
              1664  1703   1732   1737   1746   1751   1850   1869   1888
.EQUAT          1#   520#   544
.HEADE          1#   520#   521
.KT11           1#   520#   649
.SETUP          1#   520#   707
.SWRHI          1#   520#   532
.SWRLO       520#   544#
.$ACTI          1#   520#   738
```

```
.$APTB      1#
.$APTH      1#
.$APTY      1#
.$ASTA      1#
.$CATC      1#    520#    707
.$CMTA      1#    520#    749
.$DB2D      1#
.$DB2O      1#
.$DIV       1#
.$EOP       1#    520#   2301
.$ERRO      1#    520#   2415
.$ERRT      1#    520#   2460
.$MULT      1#
.$POWE      1#    520#   2935
.$RAND      1#
.$RDDE      1#
.$RDOC      1#
.$READ      1#    520#   2508
.$R2AZ      1#
.$SAVE      1#    520#   2635
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#    520#   2352
.$SIZE      1#    520#   2556
.$SUPR      1#
.$TRAP      1#    520#   2900
.$TYPB      1#
.$TYPD      1#    520#   2832
.$TYPE      1#    520#   2681
.$TYPO      1#    520#   2754
.$4CCA      1#
```

```
JSR     1091  1092  1170  1171  1245  1253  1256  1259  1262  1449  1459  1474  1490  1493  1528
        1554  1586  1599  1604  1606  1609  1612  1628  1654  1708  1727  1730  1738  1739  1747
        1749  1777  1790  1791  1792  1827  1947  1955  1956  2112  2220  2278  2344  2443  2719
        2726  2733
MOV      725  1029  1032  1040  1044  1045  1046  1047  1048  1049  1050  1051  1052  1053  1057
        1058  1059  1060  1061  1064  1065  1067  1068  1078  1079  1093  1094  1095  1096  1097
        1103  1105  1107  1108  1115  1118  1120  1128  1138  1145  1155  1169  1174  1184  1195
        1208  1211  1216  1224  1248  1249  1250  1251  1252  1254  1255  1257  1258  1260  1261
        1280  1284  1285  1286  1287  1288  1289  1290  1294  1295  1296  1297  1313  1317  1318
        1319  1320  1321  1322  1323  1327  1328  1329  1330  1347  1351  1352  1353  1354  1355
        1356  1357  1361  1362  1363  1364  1381  1385  1386  1387  1388  1389  1390  1391  1395
        1396  1397  1398  1415  1419  1420  1421  1422  1423  1424  1425  1429  1430  1431  1432
        1450  1451  1452  1453  1454  1455  1457  1458  1463  1464  1472  1473  1489  1491  1492
        1495  1496  1497  1498  1500  1501  1508  1509  1527  1530  1531  1532  1533  1536  1537
        1538  1541  1542  1543  1547  1548  1549  1553  1557  1558  1559  1560  1567  1568  1572
        1573  1577  1578  1582  1583  1585  1600  1601  1602  1617  1618  1624  1625  1626  1640
        1643  1647  1648  1649  1650  1652  1655  1663  1668  1681  1682  1688  1689  1690  1691
        1692  1693  1696  1699  1700  1701  1702  1705  1707  1713  1715  1728  1729  1733  1734
        1742  1743  1756  1758  1761  1763  1766  1768  1771  1773  1776  1788  1789  1794  1795
        1796  1797  1798  1799  1803  1804  1805  1806  1807  1811  1812  1813  1814  1815  1820
        1821  1822  1824  1826  1835  1836  1844  1845  1846  1858  1863  1864  1865  1883  1884
        1901  1902  1910  1912  1922  1924  1929  1931  1936  1942  1946  1953  1954  1957  1958
        1959  1960  1962  1966  1967  1968  1969  1970  1974  1975  1976  1977  1978  1980  1984
        1985  1986  1987  1988  1989  1997  1998  2013  2030  2031  2032  2047  2052  2053  2054
        2066  2071  2072  2073  2082  2083  2085  2086  2101  2104  2107  2110  2117  2118  2119
        2127  2128  2134  2180  2208  2209  2210  2211  2224  2225  2226  2227  2228  2234  2235
        2248  2254  2256  2265  2273  2275  2276  2277  2281  2288  2290  2291  2320  2327  2334
        2341  2373  2374  2376  2379  2392  2404  2405  2407  2408  2411  2412  2433  2438  2454
        2457  2470  2475  2484  2489  2494  2496  2500  2518  2519  2532  2533  2548  2549  2550
        2551  2573  2574  2575  2576  2577  2578  2579  2580  2581  2583  2584  2586  2587  2591
        2595  2596  2597  2598  2601  2603  2605  2610  2614  2623  2624  2625  2626  2627  2628
        2629  2630  2631  2654  2655  2656  2657  2658  2659  2660  2661  2662  2663  2670  2671
        2672  2673  2674  2675  2676  2677  2678  2679  2703  2704  2708  2722  2780  2788  2789
        2790  2796  2803  2821  2822  2823  2824  2825  2846  2847  2848  2849  2850  2851  2852
        2857  2860  2880  2886  2887  2888  2889  2890  2892  2893  2909  2910  2914  2940  2941
        2942  2943  2944  2945  2946  2947  2948  2949  2954  2958  2959  2960  2961  2962  2963
        2964  2965
MOVB    1056  2410  2440  2522  2537  2542  2705  2732  2740  2781  2782  2785  2786  2787  2791
        2794  2795  2814  2855  2858  2872  2875  2884  2912
NEG     2792  2854
NEGB    2236  2237  2238  2241
NOP     1265  1697  2212  2345  2346  2347
RESET   2343
ROL     2798  2800  2801  2802  2804
ROR     2000  2194  2202
RTI     1140  1148  1476  1587  1629  1709  1716  1778  1949  2113  2183  2413  2459  2524  2552
        2585  2664  2680  2710  2826  2894  2968
RTS     1036  1164  2076  2091  2175  2203  2221  2230  2250  2260  2269  2284  2294  2498  2632
        2749  2915
SEC     2189  2197
SEN     1684
SOB     2600
SUB     1146  1300  1333  1367  1401  1435  1757  1762  1767  1772  1923  1930  2172  2181  2439
        2621  2622  2861
SXT     1685
TRAP    2582  2917  2927  2928  2929  2930  2931  2932  2933  2934
```

```
TST      1062  1104  1641  1686  1695  1925  1932  1937  1943  2028  2245  2375  2399  2446  2455
         2502  2592  2606  2618  2707  2715  2736  2809  2866  2876  2911
TSTB     1461  1503  1877  1897  2025  2142  2239  2386  2520  2588  2699  2738  2868  2882
.ASCII    802   803  2298
.ASCIZ    801   804  1085  1089  1134  1190  1194  1202  1206  1215  1223  2298  2326  2333  2506
         2972
.BLKB    2555
.BLKW    2899
.BYTE     759   760   765   766   781   782   783   784  1197  1198  1218  1219  1226  1227  2350
         2553  2554  2827  2828  2829  2830
.ENABL      1   520
.END     2979
.ENDC     526   540   542   543   544   549   635   649   673   684   695   706   707   717   725
          733   739   745   747   750   757   759   785   793   799   800   801   802   806   824
          981  1027  1039  1044  1045  1047  1049  1051  1053  1054  1055  1057  1059  1069  1085
         1089  1125  1134  1143  1152  1168  1190  1194  1198  1199  1202  1206  1215  1219  1220
         1223  1227  1228  1239  1240  1243  1244  1245  1269  1270  1276  1277  1278  1284  1305
         1306  1310  1311  1312  1317  1339  1340  1344  1345  1346  1351  1373  1374  1378  1379
         1380  1385  1407  1408  1412  1413  1414  1419  1439  1441  1442  1447  1448  1449  1463
         1467  1469  1480  1481  1486  1487  1488  1514  1516  1519  1520  1525  1526  1527  1557
         1563  1565  1591  1592  1596  1597  1598  1614  1616  1633  1634  1638  1639  1640  1641
         1643  1665  1667  1674  1675  1679  1680  1681  1704  1712  1720  1721  1725  1726  1727
         1733  1738  1747  1752  1755  1780  1782  1783  1786  1787  1788  1831  1834  1851  1854
         1857  1870  1873  1876  1889  1892  1895  1907  1908  1909  1910  1911  1920  1952  1995
         1996  2004  2007  2011  2042  2045  2062  2065  2080  2094  2116  2132  2179  2186  2206
         2223  2232  2233  2253  2263  2272  2287  2297  2301  2302  2305  2306  2308  2310  2313
         2319  2322  2323  2326  2333  2339  2340  2349  2350  2351  2353  2359  2364  2368  2370
         2381  2384  2385  2386  2388  2390  2397  2401  2406  2407  2411  2414  2415  2416  2422
         2431  2438  2443  2444  2445  2446  2452  2459  2460  2461  2479  2508  2509  2512  2525
         2526  2533  2535  2540  2555  2556  2557  2594  2635  2636  2682  2705  2755  2833  2901
         2910  2913  2926  2927  2928  2929  2930  2931  2932  2933  2934  2935  2936  2948  2958
         2968  2975  2978
.EQUIV    549   550   552   567   568   597   598   599   600   601   602   603   604   605   606
          625   626   627   628   629   630   631   632   633   634
.EVEN    1085  1089  1134  1190  1194  1202  1206  1215  1223  2298  2326  2333  2351  2507  2974
.IF       522   540   541   542   543   544   547   607   635   662   673   684   695   706   707
          717   722   730   738   743   745   749   756   758   785   793   799   800   801   805
          806   822   978  1025  1037  1040  1044  1045  1047  1049  1051  1053  1054  1055  1057
         1069  1084  1088  1123  1133  1141  1149  1165  1189  1193  1197  1198  1201  1205  1214
         1218  1219  1222  1226  1227  1238  1240  1243  1245  1268  1270  1276  1278  1283  1304
         1306  1310  1312  1316  1338  1340  1344  1346  1350  1372  1374  1378  1380  1384  1406
         1408  1412  1414  1418  1438  1440  1442  1447  1449  1462  1466  1467  1479  1481  1486
         1488  1513  1515  1518  1520  1525  1527  1556  1562  1563  1590  1592  1596  1598  1613
         1614  1632  1634  1638  1640  1641  1642  1664  1665  1673  1675  1679  1681  1703  1710
         1719  1721  1725  1727  1732  1737  1746  1751  1752  1779  1781  1783  1786  1788  1829
         1832  1850  1852  1855  1869  1871  1874  1888  1890  1893  1906  1908  1910  1911  1918
         1950  1993  1996  2003  2005  2009  2040  2043  2060  2063  2077  2092  2114  2130  2176
         2184  2204  2222  2231  2232  2251  2261  2270  2285  2295  2298  2301  2305  2306  2308
         2309  2310  2312  2318  2321  2323  2325  2332  2340  2349  2350  2352  2358  2363  2368
         2380  2382  2383  2384  2386  2387  2388  2397  2399  2407  2408  2413  2414  2415  2421
         2431  2434  2441  2443  2444  2446  2449  2452  2459  2460  2479  2494  2508  2512  2525
         2533  2534  2539  2555  2556  2563  2594  2635  2681  2705  2754  2832  2900  2909  2913
         2917  2927  2928  2929  2930  2931  2932  2933  2934  2935  2948  2958  2966  2968  2972
         2975
.IFF      540   542   543   544   547   739   745   750   756   758   785   806   822   978  1025
         1037  1044  1123  1141  1149  1165  1198  1219  1227  1239  1240  1244  1245  1269  1270
```

```
                1277  1278  1284  1305  1306  1311  1312  1317  1339  1340  1345  1346  1351  1373  1374
                1379  1380  1385  1407  1408  1413  1414  1419  1439  1441  1442  1448  1449  1463  1467
                1480  1481  1487  1488  1514  1516  1519  1520  1526  1527  1557  1563  1591  1592  1597
                1598  1614  1633  1634  1639  1640  1641  1643  1665  1674  1675  1680  1681  1704  1710
                1720  1721  1726  1727  1733  1738  1747  1752  1780  1782  1783  1787  1788  1851  1870
                1889  1907  1908  1909  1910  1911  1918  1950  2004  2077  2092  2114  2130  2176  2184
                2204  2223  2232  2233  2251  2261  2270  2285  2295  2298  2302  2309  2313  2319  2322
                2350  2353  2381  2384  2385  2388  2414  2416  2421  2434  2459  2460  2461  2479  2508
                2509  2518  2525  2526  2534  2539  2555  2557  2572  2623  2636  2682  2755  2833  2901
                2910  2936  2968  2975
.IFT            1085  1089  1134  1190  1194  1202  1206  1215  1223  2326  2333  2396  2444  2518  2575
                2587  2627  2634
.IFTF           1085  1089  1134  1190  1194  1202  1206  1215  1223  2326  2333  2394  2443  2512  2572
                2577  2623  2630
.IIF             521   526   531   532   537   538   539   540   543   544   714   805  1045  1047  1053
                1054  1055  1057  1058  1185  1196  1217  1225  2306  2313  2314  2328  2335  2350  2351
                2359  2360  2361  2362  2363  2364  2395  2396  2411  2414  2415  2422  2423  2424  2425
                2426  2449  2460  2476  2501  2512  2548  2556  2754  2926  2927  2928  2929  2930  2931
                2932  2933  2934
.IRP             707  1238  1268  1304  1338  1372  1406  1440  1479  1518  1590  1632  1673  1719  1781
                1906  2654  2674  2846  2886  2942  2958
.LIST               1   520   543   649   707   714   721   734   785   787   788   789   790   791   792
                 793   794   795   796   797   798   799   822   823   824   978   979   980   981  1025
                1026  1027  1037  1038  1039  1059  1085  1089  1123  1124  1125  1134  1141  1142  1143
                1149  1150  1151  1152  1165  1166  1167  1168  1190  1194  1202  1206  1215  1223  1238
                1245  1268  1278  1304  1312  1338  1346  1372  1380  1406  1414  1440  1449  1467  1468
                1469  1479  1488  1518  1527  1563  1564  1565  1590  1598  1614  1615  1616  1632  1640
                1665  1666  1667  1673  1681  1710  1711  1712  1719  1727  1752  1753  1754  1755  1781
                1788  1828  1835  1851  1858  1870  1877  1889  1896  1906  1910  1918  1919  1920  1950
                1951  1952  1992  1997  2004  2012  2039  2046  2059  2066  2077  2078  2079  2080  2092
                2093  2094  2114  2115  2116  2130  2131  2132  2176  2177  2178  2179  2184  2185  2186
                2204  2205  2206  2251  2252  2253  2261  2262  2263  2270  2271  2272  2285  2286  2287
                2295  2296  2297  2298  2299  2300  2301  2313  2326  2333  2363  2449  2525  2917  2926
                2927  2928  2929  2930  2931  2932  2933  2934  2935  2975  2976  2977  2978
.MACRO             1   544   749  1237  1267  1268  1278  1303  1337  1371  1405  1439  1478  1517  1589
                1631  1672  1718  1781  2917
.MCALL           520   649  1059
.NLIST             1   520   543   649   707   714   721   734   785   787   788   789   790   791   792
                 793   794   795   796   797   798   799   822   823   824   978   979   980   981  1025
                1026  1027  1037  1038  1039  1059  1085  1089  1123  1124  1125  1134  1141  1142  1143
                1149  1150  1151  1152  1165  1166  1167  1168  1190  1194  1202  1206  1215  1223  1238
                1245  1268  1278  1304  1312  1338  1346  1372  1380  1406  1414  1440  1449  1467  1468
                1469  1479  1488  1518  1527  1563  1564  1565  1590  1598  1614  1615  1616  1632  1640
                1665  1666  1667  1673  1681  1710  1711  1712  1719  1727  1752  1753  1754  1755  1781
                1788  1828  1835  1851  1858  1870  1877  1889  1896  1906  1910  1918  1919  1920  1950
                1951  1952  1992  1997  2004  2012  2039  2046  2059  2066  2077  2078  2079  2080  2092
                2093  2094  2114  2115  2116  2130  2131  2132  2176  2177  2178  2179  2184  2185  2186
                2204  2205  2206  2251  2252  2253  2261  2262  2263  2270  2271  2272  2285  2286  2287
                2295  2296  2297  2298  2299  2300  2301  2313  2326  2333  2363  2449  2525  2917  2926
                2927  2928  2929  2930  2931  2932  2933  2934  2935  2975  2976  2977  2978
.PAGE            520   749   805
.REM               1   262
.REPT            714   722   730   787   793   822   978  1025  1037  1123  1141  1149  1165  1467  1563
                1614  1665  1710  1752  1829  1832  1852  1955  1871  1874  1890  1893  1918  1950  1993
                2005  2009  2040  2043  2060  2063  2077  2092  2114  2130  2176  2184  2204  2251  2261
                2270  2285  2295  2298  2975
```

DZKUAC.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

*DZKUAC,DZKUAC.SEQ/SOL/CRF/DS:ERFZ/EN:ABS=DSKM:SYSMAC.SML,DSKM:DZKUAC.P11
RUN-TIME: 52 47 6 SECONDS
RUN-TIME RATIO: 150/107=1.3
CORE USED: 30K (59 PAGES)

Spooler runtime 12 Seconds, 57 KCS, 347 disk reads, 3 disk writes, 76 pages