

IDENTIFICATION

B 1

SEQ 0001

PRODUCT CODE: AC-8859C-MC
PRODUCT NAME: CZKUBCO UNIBUS EXER MOD
DATE CREATED: 8-JUNE-79
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: WARREN SALTZ
MODIFIED BY: CHUCK ROBINSON

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this manual.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by Digital.

Copyright (C) 1975,1978,1979 by Digital Equipment Corporation

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION
DIGITAL PDP UNIBUS MASSBUS
DEC DECUS DECTAPE

HISTORY SECTION

1. CZKUBCO WAS RELEASED OCT 1979.

- A. CZKUBCO WAS REVISED TO ACCOMODATE THE ASYNCHRONOUS ISSUINGG OF BUS GRANTS OF THE 11/44 CPU. IN ALL TESTS WHICH ALLOW FOR INTERRUPT ADD A 'NOP' INSTRUCTION DIRECTLY FOLLOWING THE INSTRUCTION WHICH LOWERS THE PRIORITY LEVEL, THUS ALLOWING ONE INSTRUCTION TIME FOR INTERRUPT TO OCCUR. THIS CHANGE AFFECTS THE 'TIME DELAY AND BUSS LATENCY ERROR BITS TEST'.
- B. CZKUBCO WAS REVISED TO ACCOMODATE THE XON/XOFF FEATURE OF CERTAIN TERMINALS. THE \$TYPE SYSMAC MACRO WAS AFFECTED.

Table of Contents

- 1.0 Abstract
- 2.0 Requirements
 - 2.1 Equipment
 - 2.2 Preliminary Requirements
 - 2.3 Execution time
- 3.0 Starting Address
- 4.0 Program Control and Operator Action
- 5.0 Switch Options
- 6.0 Program Description
- 7.0 Error Reporting
- 8.0 Handlers and Common Routines
 - 8.1 Trap Handler
 - 8.2 Scope Handler
 - 8.3 Error Handler
 - 8.4 Trap Catcher
 - 8.5 Power Down and Up Routines
 - 8.6 CLRREG Routine
 - 8.7 RCATCH Routine
 - 8.8 CRDY Routine
 - 8.9 DINT Routine
 - 8.10 RVEC Routine
 - 8.11 TERRPC ROUTINE

1.0 Abstract

The Unibus Exercisor (UBE) module diagnostic is comprised of a series of tests that check all programmatically accessible areas of the exercisors (95%). The tests are arranged in a logical order such that simpler functions are examined first followed by the more complex ones. The tests build on one another such that the present test will use hardware previously tested. This should provide a very effective degree of fault isolation.

The program is written to test a maximum of four UBE's at one time and is intended to run in a stand-alone environment.

2.0 Requirements

2.1 Equipment

1. A working PDP-11 and Unibus
2. A working Teletype
3. A good 6K of Memory
4. A minimum of 1 to a maximum of 4 UBE on the system

2.2 Preliminary Requirements

It is expected that the module will have been tested on a GR or similar tester. This is to ensure that those areas that can not be thoroughly exercised by this program are working. These areas are:

1. Wrong Grant Error bit
2. No, No SACK time out Error bit
3. Wrong A lines Error bit
4. No Grant or not one Grant Error bit
5. No Interrupt SSYN Error bit
6. Inhibit Sack Logic.

In addition the passing of grants can not be tested if only one exercisor is present (see section 6.0). On those machines that don't have a parity trap (11/05, 11/20), the parity hardware is not checked. THE PARITY OPTION TEST (TEST 6) SHOULD BE DESELECTED BY SETTING SWITCH 5 FOR OTHER MACHINES WITHOUT PARITY MEMORY. ALSO, THE POWER DOWN TEST SHOULD NOT BE RUN ON THE 11/05.

2.3 Execution Time

For an error free, first pass run on an 11/45 with core memory, it takes approximately 15 seconds per UBE tested.

3.0 Starting Address

200 - for normal startup and restart
1100 - if halted in Interrupt test and wish to restart

4.0 Program Control and Operator Action

4.1

The paper tape is loaded using the standard procedure for ABS. tapes.

4.2

Load address 200

4.3

If the power down sequence is to be tested set SW4=1.

4.4

If more than one exercisor is present and it is desired to inhibit testing one or more of them, set the corresponding SW0,1,2,3=1. Switch 0 corresponds to the UBE which has the lowest address on the bus. Switch 1 to the next highest etc.. All UBE should not be inhibited. If this is done the program will trap to 4 after several end of passes. If all exercisors are to be tested SW0,1,2,3=0.

4.5

Start Test

5.0 Switch Options

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

SW<15>=1 Halt on Error
SW<14>=1 Loop on Test

SW<13>=1 Inhibit Error Typouts
SW<12>=1 Inhibit Most Typeouts Except Error

G 1

SEQ 0006

SW<11>=1	Inhibit Test Iterations
SW<10>=1	Bell on Error
SW<09>=1	Loop on Error
SW<05>=1	INHIBIT TEST 6
SW<04>=1	Test Power Down
SW<03>=1	inhibit Test of UBE4
SW<02>=1	Inhibit Test of UBE3
SW<01>=1	Inhibit Test of UBE2
SW<00>=1	Inhibit Test of UBE1

5.1 SW<15>

The program halts on encountering an error after printing out the error message. Pressing 'continue' restores normal program operation.

5.2 SW<14>

The program loops on the subtest that is being executed when the switch is put on.

5.3 SW<13>

This switch inhibits all error typeouts

5.4 SW<12>

This switch inhibits most typeouts except error typeouts.

5.5 SW<11>

When one iterations of each test is inhibited.

5.6 SW<10>

The bell is rung upon encountering an error.

5.7 SW<09>

Upon finding an error, the program will cycle from the point of error to the previous scope statement (see sec. 8.2).

5.8 SW<05>

THE PARITY OPTION TEST (TEST 6) SHOULD BE DESELECTED BY SETTING SWITCH 5^{I 1}
FOR MACHINES WITHOUT PARITY MEMORY.

SEQ 0008

5.9 SW<04>

When set this switch enables the test of the power down sequence and the test that DCLO clears BECC, BEBA, BECR2 and BECR1 registers. This switch should not be set when running under ACT11 since a power down will cause an error statement from ACT.

5.10 SW<03>

When set this switch inhibits testing of the fourth UBE on the bus. The fourth exercisor is defined as the exercisor that responds to the fourth lowest address of the four exercisors. If there are less than four this switch has no effect on the program.

5.11 SW<02>

When set this switch inhibits test of that UBE with the third lowest address. If there are less than three, this switch has no effect on the program.

5.12 SW<01>

When set this switch inhibits test of that UBE with the second lowest address. If there are less than two, this switch has no effect on the program.

5.13 SW<00>

When set this switch inhibits testing the lowest address exercisor on the buss. If there is one exercisor, this switch should not be set.

6.0 Program Description

Upon start of the program, a map, called EMAP, of all the exercisors present is typed out in octal. Each bit set in the map corresponds to a UBE present. The least significant bit represents the UBE whose BEBD address is 770000. The second bit represents the UBE whose BEBD address is 770020 and so on. A maximum of 4 consecutive UBEs are allowed up to the maximum address of 770076. The addresses of the first UBE to be examined are then calculated and tests 1-37 are run.

The program then checks if more exercisors are to be tested up to a maximum of four. When these are done and if there were more than one UBE, the last test is executed. This tests the passing of grants

between the exercisers.

7.0 Error Reporting

Error calls are made via the EMT instruction. The lower byte of the instruction is encoded to indicate the error number. For example ERROR 1 would be (EMT+1) or 104001. Once an error instruction is executed, an error handler routine will then process the error call. The error message to be typed is determined from the item table at the beginning of the program. Item 1 corresponds to error 1 and so on. The item table contains a series of pointers to the message to be typed.

Every time an error occurs, the PC of the error call is typed out. This will tell the user the exact test where the error occurred. Many times other pertinent information is typed out as the contents of registers and bad addresses.

All messages refer to the UBE. For example, the message 'DATI failed to set ready' means that the UBE when it did a DATI failed to set its ready.

It should be pointed out when trouble shooting a failing board, that the first error reported should be the first one fixed. This is because the nature of the hardware and software can cause additional, false or misleading error messages to appear after the first one. Since the tests build on one another and involve previously tested hardware, it will aid in the fault isolation to look up the tests previously run to know which hardware has been tested. Also, when multiple UBEs are being tested, a UBE can fail in such a way as to cause false error reports on a good board. This is especially true when the first failing UBE reports a "fatal error". Due to this, it is suggested that the first failing board reported should be repaired before proceeding to test the others.

8.0 Handlers and Common Routines

8.1 Trap Handler

This handler uses the trap instruction. The lower byte of the instruction is encoded differently for each of the different routines that use it. When a call for a routine is executed a trap occurs to the handler located at \$TRAP. The handler then determines by looking at the lower byte which address to go to for servicing the call. The following routines use this handler:

1. TYPE - this routine is used to type ASCII messages.

2. TYPDC, TYPOS, TYPON - these routines are used to change a binary number to a 6 digit octal number and type it.
3. TYPDS - this routine converts a binary number to decimal number and types it.

8.2 Scope Handler

This handler is called via the 'IOT' trap. When 'scope' is executed an 'IOT' trap occurs to the memory location '\$SCOPE'. Depending on the switch settings, the handler then decides to loop on test, loop on error etc. The scope statement that is located at the first instruction of the following test is the one that enables the desired action (looping etc.) for the present test.

8.3 Error Handler

This handler uses the 'EMT' trap. The lower byte of the instruction is encoded to indicate the error number. For example ERROR 1 would be (EMT+1) or 104001. Once an error instruction is executed the error handler determines the message to be typed. An item table at the beginning of the program contains pointers for each message to be typed. Each item corresponds to each error (Item 1 corresponds to error 1). The 'ERRTYP' routine then processes the table for the final error type out.

8.4 Trap Catcher

This is a series of instructions starting in location 0 to detect unexpected traps and interrupts to the trap and interrupt vector area of memory.

Each vector PC address is loaded with the address of the next location. The next location is loaded with a halt. Thus an illegal trap or interrupt will cause a halt at the trap PSW location plus 2.

Once a halt occurs, by examining the contents of the address pointed to by the stack, the value of the PC when the trap or interrupt occurred can be determined.

8.5 Power Down and Up Routines

When a power fail condition occurs, the contents of registers R0-R7 are saved on the stack. When the power returns, the same registers are restored.

8.6 CLRREG Routine

This subroutine will clear all the registers and error conditions of the UBE presently being tested.

8.7 RCATCH Routine

This routine restores the trap catcher to the vector area of the UBE presently being tested.

8.8 CRDY Routine

This routine checks for the ready bit to set from the UBE presently being tested. If ready fails to set in a time > 100 microseconds, the LSB of register R4 is set to a one.

8.9 DINT Routine

This routine is used to disregard interrupts from the UBE under test. It places the address of the next location in the UBE's vector area. The next location then contains an 'RTI' instruction.

8.10 RVEC Routine

This subroutine restores the vector area 0-56 from the stack and puts the trap catcher in the remaining locations.

8.11 TERRPC Routine

This routine is used any time an error occurs. It types out the PC of the error message, AND THE TEST NUMBER.

```

13      .TITLE CZKUBCO UNIBUS EXER MOD
        : *COPYRIGHT (C) 1979
        : *DIGITAL EQUIPMENT CORP.
        : *MAYNARD, MASS. 01754
        : *
        : *PROGRAM BY CHUCK ROBINSON
        : *
        : *THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
        : *PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
        : *
14      000001 $TN=1
        .SBTTL OPERATIONAL SWITCH SETTINGS
        : *
        : *      SWITCH      USE
        : *      -----
        : *      15      HALT ON ERROR
        : *      14      LOOP ON TEST
        : *      13      INHIBIT ERROR TYPEOUTS
        : *      12      INHIBIT MOST TYPEOUTS EXCEPT ERROR
        : *      11      INHIBIT ITERATIONS
        : *      10      BELL ON ERROR
        : *      9       LOOP ON ERROR
15      : *      5       WHEN SET, INHIBIT TEST 6
        : *      4       TEST POWER DOWN
        : *      3       INHIBIT TEST OF UBE 4
16      : *      2       INHIBIT TEST OF UBE 3
        : *      1       INHIBIT TEST OF UBE 2
17      : *      0       INHIBIT TETS OF UBE 1
        .SBTTL BASIC DEFINITIONS

        : *INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
        STACK= 1100
        ERROR=EMT
        SCOPE=IOT

        : *MISCELLANEOUS DEFINITIONS
000011 HT= 11      ::CODE FOR HORIZONTAL TAB
000012 LF= 12      ::CODE FOR LINE FEED
000015 CR= 15      ::CODE FOR CARRIAGE RETURN
000200 CRLF= 200     ::CODE FOR CARRIAGE RETURN-LINE FEED
177776 PS= 177776 ::PROCESSOR STATUS WORD
177776 PSW=PS
177774 STKLMT= 177774 ::STACK LIMIT REGISTER
177772 PIRQ= 177772 ::PROGRAM INTERRUPT REQUEST REGISTER
177570 DSWR= 177570 ::HARDWARE SWITCH REGISTER
177570 DDISP= 177570 ::HARDWARE DISPLAY REGISTER

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

```

```
000007      PC=      %7          ;;PROGRAM COUNTER

;*PRIORITY LEVEL DEFINITIONS
000000      PR0=      0           ;;PRIORITY LEVEL 0
000040      PR1=     40          ;;PRIORITY LEVEL 1
000100      PR2=    100         ;;PRIORITY LEVEL 2
000140      PR3=    140         ;;PRIORITY LEVEL 3
000200      PR4=    200         ;;PRIORITY LEVEL 4
000240      PR5=    240         ;;PRIORITY LEVEL 5
000300      PR6=    300         ;;PRIORITY LEVEL 6
000340      PR7=    340         ;;PRIORITY LEVEL 7

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

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

```

001000 BIT9=BIT09
000400 BIT8=BIT08
000200 BIT7=BIT07
000100 BIT6=BIT06
000040 BIT5=BIT05
000020 BIT4=BIT04
000010 BIT3=BIT03
000004 BIT2=BIT02
000002 BIT1=BIT01
000001 BIT0=BIT00

;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004 ERRVEC= 4 ;;TIME OUT AND OTHER ERRORS
000010 RESVEC= 10 ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014 TBITVEC=14 ;;'T' BIT
000014 TRTVEC= 14 ;;TRACE TRAP
000014 BPTVEC= 14 ;;BREAKPOINT TRAP (BPT)
000020 IOTVEC= 20 ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024 PWRVEC= 24 ;;POWER FAIL
000030 EMTVEC= 30 ;;EMULATOR TRAP (EMT) **ERROR**
000034 TRAPVEC=34 ;;'TRAP' TRAP
000060 TKVEC= 60 ;;TTY KEYBOARD VECTOR
000064 TPVEC= 64 ;;TTY PRINTER VECTOR
000240 PIRQVEC=240 ;;PROGRAM INTERRUPT REQUEST VECTOR
.MCALL TYPTXT,POP,PUSH,NEWTST,$$NEWTEST,SWRSU,SETUP,SPACE,STARS
18 DB=170000 ;DATA BUFFER OF LOWEST ADDRESS USE
20 .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
000174 000000 DISPREG: .WORD 0 ;;SOFTWARE DISPLAY REGISTER
000176 000000 SWREG: .WORD 0 ;;SOFTWARE SWITCH REGISTER
.SBTTL STARTING ADDRESS(ES)
000200 000137 002632 JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
21 001100
22 001100 012737 000137 000200 RSTART: MOV #000137,@#200 ;RESTART HERE IF HALTED IN INTERRUPT TEST
23 001106 012737 002632 000202 MOV #START,@#202
24 001114 020627 001014 CMP R6,#1014 ;WAS VECTOR AREA DESTROYED IN INT. TEST?
25 001120 101002 BHI B ;BRANCH IF NO
26 001122 004767 015170 JSR PC,RVEC ;RESTORE VECTOR AREA
27 001126 000137 002632 B: JMP @#START ;GO TO BEGINNING OF PROGRAM
28 .SBTTL ACT11 HOOKS

;*****
;HOOKS REQUIRED BY ACT11
001132 $SVPC=. ;SAVE PC
000046 .=46
000046 016136 $ENDAD ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052 .=52
000052 000000 .WORD 0 ;;2)SET LOC.52 TO ZERO
001132 .=$SVPC ;RESTORE PC

```


29

.SBTTL COMMON TAGS

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

001132	001132		.=1132	\$CMTAG:		:: START OF COMMON TAGS
001132	000000			\$PASS:	.WORD 0	:: CONTAINS PASS COUNT
001134	000			\$TSTNM:	.BYTE 0	:: CONTAINS THE TEST NUMBER
001135	000			\$ERFLG:	.BYTE 0	:: CONTAINS ERROR FLAG
001136	000000			\$ICNT:	.WORD 0	:: CONTAINS SUBTEST ITERATION COUNT
001140	000000			\$LPADR:	.WORD 0	:: CONTAINS SCOPE LOOP ADDRESS
001142	000000			\$LPERR:	.WORD 0	:: CONTAINS SCOPE RETURN FOR ERRORS
001144	000000			\$ERTTL:	.WORD 0	:: CONTAINS TOTAL ERRORS DETECTED
001146	000			\$ITEMB:	.BYTE 0	:: CONTAINS ITEM CONTROL BYTE
001147	001			\$ERMAX:	.BYTE 1	:: CONTAINS MAX. ERRORS PER TEST
001150	000000			\$ERRPC:	.WORD 0	:: CONTAINS PC OF LAST ERROR INSTRUCTION
001152	000000			\$GDADR:	.WORD 0	:: CONTAINS ADDRESS OF 'GOOD' DATA
001154	000000			\$BDADR:	.WORD 0	:: CONTAINS ADDRESS OF 'BAD' DATA
001156	000000			\$GDDAT:	.WORD 0	:: CONTAINS 'GOOD' DATA
001160	000000			\$BDDAT:	.WORD 0	:: CONTAINS 'BAD' DATA
001162	000000				.WORD 0	:: RESERVED--NOT TO BE USED
001164	000000				.WORD 0	
001166	000			\$AUTOB:	.BYTE 0	:: AUTOMATIC MODE INDICATOR
001167	000			\$INTAG:	.BYTE 0	:: INTERRUPT MODE INDICATOR
001170	000000				.WORD 0	
001172	177570			\$SWR:	.WORD DSWR	:: ADDRESS OF SWITCH REGISTER
001174	177570			\$DISPLAY:	.WORD DDISP	:: ADDRESS OF DISPLAY REGISTER
001176	177560			\$TKS:	177560	:: TTY KBD STATUS
001200	177562			\$TKB:	177562	:: TTY KBD BUFFER
001202	177564			\$TPS:	177564	:: TTY PRINTER STATUS REG. ADDRESS
001204	177566			\$TPB:	177566	:: TTY PRINTER BUFFER REG. ADDRESS
001206	000			\$NULL:	.BYTE 0	:: CONTAINS NULL CHARACTER FOR FILLS
001207	002			\$FILLS:	.BYTE 2	:: CONTAINS # OF FILLER CHARACTERS REQUIRED
001210	012			\$FILLC:	.BYTE 12	:: INSERT FILL CHARS. AFTER A 'LINE FEED'
001211	000			\$TPFLG:	.BYTE 0	:: 'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
001212	000000			\$REGAD:	.WORD 0	:: CONTAINS THE ADDRESS FROM WHICH (\$REGO) WAS OBTAINED
001214	000004				.REPT \$CM3	
001214	000000			\$REG0:	.WORD 0	:: CONTAINS ((\$REGAD)+0)
001216	000000			\$REG1:	.WORD 0	:: CONTAINS ((\$REGAD)+2)
001220	000000			\$REG2:	.WORD 0	:: CONTAINS ((\$REGAD)+4)
001222	000000			\$REG3:	.WORD 0	:: CONTAINS ((\$REGAD)+6)
001224	000004				.REPT 4	
001224	000000			\$TMP0:	.WORD 0	:: USER DEFINED
001226	000000			\$TMP1:	.WORD 0	:: USER DEFINED
001230	000000			\$TMP2:	.WORD 0	:: USER DEFINED
001232	000000			\$TMP3:	.WORD 0	:: USER DEFINED
001234	000000			\$TIMES:	0	:: MAX. NUMBER OF ITERATIONS
001236	000000			\$ESCAPE:	0	:: ESCAPE ON ERROR ADDRESS
001240	207	377	377	\$BELL:	.ASCIZ <207><377><377>	:: CODE FOR BELL
001243	000					
001244	077			\$QUES:	.ASCII /?/	:: QUESTION MARK
001245	015			\$CRLF:	.ASCII <15>	:: CARRIAGE RETURN
001246	012	000		\$LF:	.ASCIZ <12>	:: LINE FEED
						::*****

.SBTTL ERROR POINTER TABLE

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

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

			\$ERRTB:
30	001250		:ITEM1
31	001250	020750	EM1
32	001252	000000	0
33	001254	000000	0
34	001256	000000	0
35			:ITEM2
36	001260	021032	EM2
37	001262	021072	DH2
38	001264	021120	DT2
39	001266	000000	0
40			:ITEM3
41	001270	021126	EM3
42	001272	021173	DH3
43	001274	021202	DT3
44	001276	000000	0
45			:ITEM 4
46	001300	021206	EM4
47	001302	021254	DH4
48	001304	021314	DT4
49	001306	000000	0
50			:ITEM 5
51	001310	021324	EM5
52	001312	021254	DH4
53	001314	021314	DT4
54	001316	000000	0
55			:ITEM 6
56	001320	021372	EM6
57	001322	021254	DH4
58	001324	021314	DT4
59	001326	000000	0
60			:ITEM 7
61	001330	021442	EM7
62	001332	021504	DH7
63	001334	021562	DT7
64	001336	000000	0
65			:ITEM 8
66	001340	021570	EM8
67	001342	000000	0
68	001344	000000	0
69	001346	000000	0
70			:ITEM 9
71	001350	021655	EM9

72	001352	000000	0
73	001354	000000	0
74	001356	000000	0
75			:ITEM 10
76	001360	021716	EM10
77	001362	000000	0
78	001364	000000	0
79	001366	000000	0
80			:ITEM 11
81	001370	021757	EM11
82	001372	000000	0
83	001374	000000	0
84	001376	000000	0
85			:ITEM 12
86	001400	022023	EM12
87	001402	000000	0
88	001404	000000	0
89	001406	000000	0
90			:ITEM 13
91	001410	000000	0
92	001412	000000	0
93	001414	000000	0
94	001416	000000	0
95			:ITEM 14
96	001420	022070	EM14
97	001422	000000	0
98	001424	000000	0
99	001426	000000	0
100			:ITEM 15
101	001430	022121	EM15
102	001432	022205	DH15
103	001434	021202	DT3
104	001436	000000	0
105			:ITEM 16
106	001440	022227	EM16
107	001442	000000	0
108	001444	000000	0
109	001446	000000	0
110			:ITEM 17
111	001450	022322	EM17
112	001452	022361	DH17
113	001454	021120	DT2
114	001456	000000	0
115			:ITEM 18
116	001460	022406	EM18
117	001462	022475	DH18
118	001464	021202	DT3
119	001466	000000	0
120			:ITEM 19
121	001470	022510	EM19
122	001472	022571	DH19
123	001474	021202	DT3
124	001476	000000	0
125			:ITEM 20
126	001500	022612	EM20
127	001502	000000	0
128	001504	000000	0

Line	Code	Value	Label
129	001506	000000	
130			;ITEM 21
131	001510	022661	EM21
132	001512	000000	0
133	001514	000000	0
134	001516	000000	0
135			;ITEM 22
136	001520	022717	EM22
137	001522	000000	0
138	001524	000000	0
139	001526	000000	0
140			;ITEM 23
141	001530	022762	EM23
142	001532	000000	0
143	001534	000000	0
144	001536	000000	0
145			;ITEM 24
146	001540	023026	EM24
147	001542	023075	DH24
148	001544	023152	DT24
149	001546	000000	0
150			;ITEM 25
151	001550	023164	EM25
152	001552	023075	DH24
153	001554	023152	DT24
154	001556	000000	0
155			;ITEM 26
156	001560	023224	EM26
157	001562	023075	DH24
158	001564	023152	DT24
159	001566	000000	0
160			;ITEM 27
161	001570	023265	EM27
162	001572	023075	DH24
163	001574	023152	DT24
164	001576	000000	0
165			;ITEM 28
166	001600	023325	EM28
167	001602	000000	0
168	001604	000000	0
169	001606	000000	0
170			;ITEM 29
171	001610	023354	EM29
172	001612	000000	0
173	001614	000000	0
174	001616	000000	0
175			;ITEM 30
176	001620	023403	EM30
177	001622	000000	0
178	001624	000000	0
179	001626	000000	0
180			;ITEM 31
181	001630	023433	EM31
182	001632	000000	0
183	001634	000000	0
184	001636	000000	0
185			;ITEM 32

186	001640	023463	EM32
187	001642	023075	DH24
188	001644	023152	DT24
189	001646	000000	0
190			;ITEM 33
191	001650	023532	EM33
192	001652	023075	DH24
193	001654	023152	DT24
194	001656	000000	0
195			;ITEM 34
196	001660	023567	EM34
197	001662	023637	DH34
198	001664	021202	DT3
199	001666	000000	0
200			;ITEM 35
201	001670	023656	EM35
202	001672	023733	DH35
203	001674	021120	DT2
204	001676	000000	0
205			;ITEM 36
206	001700	023761	EM36
207	001702	024030	DH36
208	001704	021202	DT3
209	001706	000000	0
210			;ITEM 37
211	001710	024040	EM37
212	001712	023733	DH35
213	001714	021120	DT2
214	001716	000000	0
215			;ITEM 38
216	001720	024077	EM38
217	001722	023733	DH35
218	001724	021120	DT2
219	001726	000000	0
220			;ITEM 39
221	001730	024167	EM39
222	001732	000000	0
223	001734	000000	0
224	001736	000000	0
225			;ITEM 40
226	001740	024245	EM40
227	001742	000000	0
228	001744	000000	0
229	001746	000000	0
230			;ITEM 41
231	001750	024323	EM41
232	001752	000000	0
233	001754	000000	0
234	001756	000000	0
235			;ITEM 42
236	001760	024365	EM42
237	001762	000000	0
238	001764	000000	0
239	001766	000000	0
240			;ITEM 43
241	001770	024424	EM43
242	001772	024510	DH43

243	001774	021120	DT2
244	001776	000000	0
245			:ITEM 44
246	002000	024541	EM44
247	002002	000000	0
248	002004	000000	0
249	002006	000000	0
250			:ITEM 45
251	002010	024605	EM45
252	002012	000000	0
253	002014	000000	0
254	002016	000000	0
255			:ITEM 46
256	002020	024632	EM46
257	002022	024702	DH46
258	002024	021314	DT4
259	002026	000000	0
260			:ITEM 47
261	002030	024740	EM47
262	002032	024510	DH43
263	002034	021120	DT2
264	002036	000000	0
265			:ITEM 48
266	002040	024740	EM47
267	002042	000000	0
268	002044	000000	0
269	002046	000000	0
270			:ITEM 49
271	002050	025016	EM49
272	002052	000000	0
273	002054	000000	0
274	002056	000000	0
275			:ITEM 50
276	002060	025016	EM49
277	002062	024510	DH43
278	002064	021120	DT2
279	002066	000000	0
280			:ITEM 51
281	002070	025075	EM51
282	002072	000000	0
283	002074	000000	0
284	002076	000000	0
285			:ITEM 52
286	002100	025126	EM52
287	002102	000000	0
288	002104	000000	0
289	002106	000000	0
290			:ITEM 53
291	002110	025161	EM53
292	002112	000000	0
293	002114	000000	0
294	002116	000000	0
295			:ITEM 54
296	002120	025233	EM54
297	002122	000000	0
298	002124	000000	0
299	002126	000000	0

300			:ITEM 55	
301	002130	024424		EM43
302	002132	000000		0
303	002134	000000		0
304	002136	000000		0
305			:ITEM 56	
306	002140	025476		EM56
307	002142	000000		0
308	002144	000000		0
309	002146	000000		0
310			:ITEM 57	
311	002150	025555		EM57
312	002152	000000		0
313	002154	000000		0
314	002156	000000		0
315			:ITEM 58	
316	002160	025605		EM58
317	002162	022205		DH15
318	002164	021202		DT3
319	002166	000000		0
320			:ITEM 59	
321	002170	025626		EM59
322	002172	000000		0
323	002174	000000		0
324	002176	000000		0
325			:ITEM 60	
326	002200	025674		EM60
327	002202	000000		0
328	002204	000000		0
329	002206	000000		0
330			:ITEM 61	
331	002210	025722		EM61
332	002212	000000		0
333	002214	000000		0
334	002216	000000		0
335			:ITEM 62	
336	002220	025751		EM62
337	002222	000000		0
338	002224	000000		0
339	002226	000000		0
340			:ITEM 63	
341	002230	026001		EM63
342	002232	022205		DH15
343	002234	021202		DT3
344	002236	000000		0
345			:ITEM 64	
346	002240	026031		EM64
347	002242	000000		0
348	002244	000000		0
349	002246	000000		0
350			:ITEM 65	
351	002250	026127		EM65
352	002252	026207		DH65
353	002254	021202		DT3
354	002256	000000		0
355			:ITEM 66	
356	002260	026226		EM66

357	002262	000000	0
358	002264	000000	0
359	002266	000000	0
360			:ITEM 67
361	002270	026254	EM67
362	002272	026207	DH65
363	002274	021202	DT3
364	002276	000000	0
365			:ITEM 68
366	002300	000000	0
367	002302	026207	DH65
368	002304	021202	DT3
369	002306	000000	0
370			:ITEM 69
371	002310	026316	EM69
372	002312	000000	0
373	002314	000000	0
374	002316	000000	0
375			:ITEM 70
376	002320	026347	EM70
377	002322	000000	0
378	002324	000000	0
379	002326	000000	0
380			:ITEM 71
381	002330	026430	EM71
382	002332	000000	0
383	002334	000000	0
384	002336	000000	0
385			:ITEM 72
386	002340	026456	EM72
387	002342	000000	0
388	002344	000000	0
389	002346	000000	0
390			:ITEM 73
391	002350	026506	EM73
392	002352	000000	0
393	002354	000000	0
394	002356	000000	0
395			:ITEM 74
396	002360	026553	EM74
397	002362	000000	0
398	002364	000000	0
399	002366	000000	0
400			:ITEM 75
401	002370	026575	EM75
402	002372	000000	0
403	002374	000000	0
404	002376	000000	0
405			:ITEM 76
406	002400	026615	EM76
407	002402	000000	0
408	002404	000000	0
409	002406	000000	0
410			:ITEM 77
411	002410	026642	EM77
412	002412	000000	0
413	002414	000000	0

414 002416 000000
 415
 416 002420 026670
 417 002422 021254
 418 002424 021314
 419 002426 000000
 420
 421 002430 000000
 422 002432 000000
 423 002434 000000
 424 002436 000000
 425
 426 002440 026730
 427 002442 000000
 428 002444 000000
 429 002446 000000
 430
 431 002450 026772
 432 002452 024702
 433 002454 021314
 434 002456 000000
 435
 436 002460 027016
 437 002462 000000
 438 002464 000000
 439 002466 000000
 440
 441 002470 027063
 442 002472 000000
 443 002474 000000
 444 002476 000000
 445
 446 002500 027144
 447 002502 000000
 448 002504 000000
 449 002506 000000
 450 002510 000000
 451 002512 000000
 452 002514 000000
 453 002516 000000
 454 002520 000000
 455 002522 000000
 456 002524 000000
 457 002526 000000
 458 002530 000000
 459 002532 000000
 460 002534 170014
 461 002536 000000
 462 002540 000000
 463 002542 000000
 464 002544 000000
 465 002546 000000
 466 002550 000000
 467 002552 000000
 468 002554 000000
 469 002556 000000
 470 002560 000000

0
 :ITEM 78
 EM78
 DH4
 DT4
 0
 :ITEM 79
 0
 0
 0
 0
 :ITEM 80
 EM80
 0
 0
 0
 :ITEM 81
 EM81
 DH46
 DT4
 0
 :ITEM 82
 EM82
 0
 0
 0
 :ITEM 83
 EM83
 0
 0
 0
 :ITEM 84
 EM84
 0
 0
 0
 EMAP: .WORD 0
 TMAP: .WORD 0
 SPTR: .WORD 0
 BEBD: .WORD 0
 BECC: .WORD 0
 BEBA: .WORD 0
 BECR1: .WORD 0
 BECR2: .WORD 0
 BERE: .WORD 0
 INTVEC: .WORD 0
 BEGO: .WORD 170014
 BE1BD: .WORD 0
 BE1CC: .WORD 0
 BE1BA: .WORD 0
 BE1CR1: .WORD 0
 BE1CR2: .WORD 0
 BE1RE: .WORD 0
 BE1VEC: .WORD 0
 BE2BD: .WORD 0
 BE2CC: .WORD 0
 BE2BA: .WORD 0

:MAP OF UBE PRESENT
 :TEMPORARY MAP
 :SWITCH POINTER
 :BEBD ADDRESS OF UBE UNDER TEST
 :BECC ADDRESS OF UBE UNDER TEST
 :BEBA ADDRESS OF UBE UNDER TEST
 :BECR1 ADDRESS OF UBE UNDER TEST
 :BECR2 ADDRESS OF UBE UNDER TEST
 :CLEAR ERROR ADDRESS OF UBE UNDER TEST
 :INTERRUPT VECTOR ADDRESS OF UBE UNDER TEST
 :GO ADDRESS
 :BEBD ADDRESS OF FIRST UBE TESTED
 :BECC ADDRESS OF FIRST UBE TESTED
 :BEBA ADDRESS OF FIRST UBE TESTED
 :BECR1 ADDRESS OF FIRST UBE TESTED
 :BECR2 ADDRESS OF FIRST UBE TESTED
 :CLEAR ERROR ADDRESS OF FIRST UBE TESTED
 :INTERRUPT VECTOR ADDRESS OF FIRST UBE TESTED
 :BEBD ADDRESS OF SECOND UBE TESTED
 :BECC ADDRESS OF SECOND UBE TESTED
 :BEBA ADDRESS OF SECOND UBE TESTED

471	002562	000000	BE2CR1: .WORD 0	;BECR1 ADDRESS OF SECOND UBE TESTED
472	002564	000000	BE2CR2: .WORD 0	;BECR2 ADDRESS OF SECOND UBE TESTED
473	002566	000000	BE2RE: .WORD 0	;CLEAR ERROR ADDRESS OF SECOND UBE TESTED
474	002570	000000	BE2VEC: .WORD 0	;INTERRUPT VECTOR ADDRESS OF SECOND UBE TESTED
475	002572	000000	BE3BD: .WORD 0	;BEBD ADDRESS OF THIRD UBE TESTED
476	002574	000000	BE3CC: .WORD 0	;BECC ADDRESS OF THIRD UBE TESTED
477	002576	000000	BE3BA: .WORD 0	;BEBA ADDRESS OF THIRD UBE TESTED
478	002600	000000	BE3CR1: .WORD 0	;BECR1 ADDRESS OF THIRD UBE TESTED
479	002602	000000	BE3CR2: .WORD 0	;BECR2 ADDRESS OF THIRD UBE TESTED
480	002604	000000	BE3RE: .WORD 0	;CLEAR ERROR ADDRESS OF THIRD UBE TESTED
481	002606	000000	BE3VEC: .WORD 0	;INTERRUPT VECTOR ADDRESS OF THIRD UBE TESTED
482	002610	000000	BE4BD: .WORD 0	;BEBD ADDRESS OF FOURTH UBE TESTED
483	002612	000000	BE4CC: .WORD 0	;BECC ADDRESS OF FOURTH UBE TESTED
484	002614	000000	BE4BA: .WORD 0	;BEBA ADDRESS OF FOURTH UBE TESTED
485	002616	000000	BE4CR1: .WORD 0	;BECR1 ADDRESS OF FOURTH UBE TESTED
486	002620	000000	BE4CR2: .WORD 0	;BECR2 ADDRESS OF FOURTH UBE TESTED
487	002622	000000	BE4RE: .WORD 0	;CLEAR ERROR ADDRESS OF FOURTH UBE TESTED
488	002624	000000	BE4VEC: .WORD 0	;INTERRUPT VECTOR ADDRESS OF FOURTH UBE TESTED
489	002626	000000	UCNT: .WORD 0	;COUNT OF UBE TESTED
490	002630	000000	NO: .WORD 0	;INDEX NUMBER FOR ADDRESS OF 1,2,3,4 UBE
491				

492 002632

START:

			.SBTTL	INITIALIZE THE COMMON TAGS
			::CLEAR	THE COMMON TAGS (\$CMTAG) AREA
002632	012706	001132	MOV	#\$CMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
002636	005026		CLR	(R6)+ ;:CLEAR MEMORY LOCATION
002640	022706	001172	CMP	#\$SWR,R6 ;:DONE?
002644	001374		BNE	.-6 ;:LOOP BACK IF NO
002646	012706	001100	MOV	#\$STACK,SP ;:SETUP THE STACK POINTER
			::INITIALIZE	A FEW VECTORS
002652	012737	016472	MOV	#\$SCOPE,@#IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
002660	012737	000340	MOV	#340,@#IOTVEC+2 ;:LEVEL 7
002666	012737	016722	MOV	#\$ERROF,@#EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
002674	012737	000340	MOV	#340,@#EMTVEC+2 ;:LEVEL 7
002702	012737	020166	MOV	#\$TRAP,@#TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
002710	012737	000340	MOV	#340,@#TRAPVEC+2 ;:LEVEL 7
002716	012737	020236	MOV	#\$PWRDN,@#PWRVEC ;:POWER FAILURE VECTOR
002724	012737	000340	MOV	#340,@#PWRVEC+2 ;:LEVEL 7
002732	016767	013146	MOV	\$ENDCT,\$EOPCT ;:SETUP END-OF-PROGRAM COUNTER
002740	005067	176270	CLR	\$TIMES ;:INITIALIZE NUMBER OF ITERATIONS
002744	005067	176266	CLR	\$ESCAPE ;:CLEAR THE ESCAPE ON ERROR ADDRESS
002750	112767	000001	MOVB	#1,\$ERMAX ;:ALLOW ONE ERROR PER TEST
002756	012767	002756	MOV	#,\$LPADR ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
002764	012767	002764	MOV	#,\$LPERR ;:SETUP THE ERROR LOOP ADDRESS
			::SIZE	FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
			::EQUAL	TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
002772	013746	000004	MOV	@#ERRVEC,-(SP) ;:SAVE ERROR VECTOR
002776	012737	003032	MOV	#64\$,@#ERRVEC ;:SET UP ERROR VECTOR
003004	012767	177570	MOV	#\$DSWR,\$SWR ;:SETUP FOR A HARDWARE SWITCH REGISTER
003012	012767	177570	MOV	#\$DDISP,\$DISPLAY ;:AND A HARDWARE DISPLAY REGISTER
003020	022777	177777	CMP	#-1,@\$SWR ;:TRY TO REFERENCE HARDWARE SWR
003026	001012		BNE	66\$;:BRANCH IF NO TIMEOUT TRAP OCCURRED
				;:AND THE HARDWARE SWR IS NOT = -1
003030	000403		BR	65\$;:BRANCH IF NO TIMEOUT
003032	012716	003040	64\$: MOV	#65\$, (SP) ;:SET UP FOR TRAP RETURN
003036	000002		RTI	

```

003040 012767 000176 176124 65$: MOV #SWREG,SWR ;:POINT TO SOFTWARE SWR
003046 012767 000174 176120 MOV #DISPREG,DISPLAY
003054 012637 000004 66$: MOV (SP)+,@#ERRVEC ;:RESTORE ERROR VECTOR

493 003060 032777 010000 176104 BIT #SW12,@SWR ;:INHIBIT TYPEOUTS?
494 003066 001004 BNE START1 ;:BRANCH IF YES
495 003070 104401 027737 TYPE ,MSG16 ;:UBE MODULE TEST
496 003074 104401 027442 TYPE ,MSG12 ;:JUMPER W1 SHOULD BE IN TO PREVENT MULTIPLE SSYNS
497 003100 005067 177404 START1: CLR EMAP ;:INIT. EMAP
498 003104 012706 001100 MOV #STACK, SP ;:SETUP THE STACK POINTER
499 003110 012767 000001 177376 MOV #1,SPTR ;:INITIALIZE SWITCH POINTER TO LOOK AT FIRST SWITCH
500 003116 012767 002632 176016 MOV #START,$LPERR ;:SET UP RETURN FOR ERROR1
501 003124 012737 003234 000004 MOV #MTRAP,@#4 ;:SET UP MAP TRAP
502 003132 012737 000340 000006 MOV #340,@#6 ;:SET PSW PRIORITY=7
503 003140 012701 170000 MOV #DB,R1 ;:DATA REG ADDR. OF FIRST REG
504 003144 012700 000001 MOV #1,R0 ;:LD PTER
505 003150 005711 LOOP1: TST (R1) ;:LOOK IF EXER. PRESENT,NO TRAPS
506 003152 050067 177332 BIS R0,EMAP ;:YES,INDIC. EXER. PRESENT
507 003156 062701 000020 LOOP2: ADD #20,R1 ;:LOOK AT NEXT EXER. ADDR.
508 003162 006100 ROL R0 ;:UPDATE PTER
509 003164 020027 000020 CMP R0,#20 ;:AT LAST UBE?
510 003170 001367 BNE LOOP1 ;:BRANCH IF NOT AT LAST POSSIBLE EXER.
511 003172 012737 000006 000004 A: MOV #6,@#4 ;:RESTORE TRAP CATCHER
512 003200 005037 000006 CLR @#6 ;:
513 003204 032777 010000 175760 BIT #SW12,@SWR ;:INHIBIT TYPEOUTS?
514 003212 001007 BNE 1$ ;:BRANCH IF YES
515 003214 104401 020414 TYPE ,MSG1 ;:TYPE MAP
516 003220 016746 177264 MOV EMAP,-(SP) ;:SAVE EMAP FOR TYPEOUT
003224 104402 TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
517 003226 104401 001245 TYPE ,$CRLF ;:
518 003232 000415 1$: BR IADD ;:GO CALC. ADDRESSES OF UBE
519 ;:
520 003234 022626 MTRAP: CMP (SP)+,(SP)+ ;:RESTORE THE STACK
521 003236 020027 000010 CMP R0,#10 ;:AT END OF UBE ADDRESS SPACE?
522 003242 001345 BNE LOOP2 ;:NO LOOK AT NEXT EXER.
523 003244 026727 177240 000000 CMP EMAP,#0 ;:YES,IS MAP = 0?
524 003252 001347 BNE A ;:NO,BRANCH TO A
525 003254 104001 ERROR +^D1 ;:NO RESPONSE TO REG ADDRESSES OR NO DEVICE PRESENT
526 003256 004767 013150 JSR PC,TERRPC ;:TYPE PC OF ERROR MSG
527 003262 000167 012562 JMP $EOP ;:GO TO END OF TEST
528 ;:
529 ;:
530 ;:
531 ;:
532 ;:
533 ;:
534 ;:
535 ;:
536 ;:
537 ;:
538 ;:
539 ;:
540 ;:
541 ;:
542 ;:
543 ;:
544 ;:
545 ;:
546 ;:
547 ;:
548 ;:
549 ;:
550 ;:
551 ;:
552 ;:
553 ;:
554 ;:
555 ;:
556 ;:
557 ;:
558 ;:
559 ;:
560 ;:
561 ;:
562 ;:
563 ;:
564 ;:
565 ;:
566 ;:
567 ;:
568 ;:
569 ;:
570 ;:
571 ;:
572 ;:
573 ;:
574 ;:
575 ;:
576 ;:
577 ;:
578 ;:
579 ;:
580 ;:
581 ;:
582 ;:
583 ;:
584 ;:
585 ;:
586 ;:
587 ;:
588 ;:
589 ;:
590 ;:
591 ;:
592 ;:
593 ;:
594 ;:
595 ;:
596 ;:
597 ;:
598 ;:
599 ;:
600 ;:
601 ;:
602 ;:
603 ;:
604 ;:
605 ;:
606 ;:
607 ;:
608 ;:
609 ;:
610 ;:
611 ;:
612 ;:
613 ;:
614 ;:
615 ;:
616 ;:
617 ;:
618 ;:
619 ;:
620 ;:
621 ;:
622 ;:
623 ;:
624 ;:
625 ;:
626 ;:
627 ;:
628 ;:
629 ;:
630 ;:
631 ;:
632 ;:
633 ;:
634 ;:
635 ;:
636 ;:
637 ;:
638 ;:
639 ;:
640 ;:
641 ;:
642 ;:
643 ;:
644 ;:
645 ;:
646 ;:
647 ;:
648 ;:
649 ;:
650 ;:
651 ;:
652 ;:
653 ;:
654 ;:
655 ;:
656 ;:
657 ;:
658 ;:
659 ;:
660 ;:
661 ;:
662 ;:
663 ;:
664 ;:
665 ;:
666 ;:
667 ;:
668 ;:
669 ;:
670 ;:
671 ;:
672 ;:
673 ;:
674 ;:
675 ;:
676 ;:
677 ;:
678 ;:
679 ;:
680 ;:
681 ;:
682 ;:
683 ;:
684 ;:
685 ;:
686 ;:
687 ;:
688 ;:
689 ;:
690 ;:
691 ;:
692 ;:
693 ;:
694 ;:
695 ;:
696 ;:
697 ;:
698 ;:
699 ;:
700 ;:
701 ;:
702 ;:
703 ;:
704 ;:
705 ;:
706 ;:
707 ;:
708 ;:
709 ;:
710 ;:
711 ;:
712 ;:
713 ;:
714 ;:
715 ;:
716 ;:
717 ;:
718 ;:
719 ;:
720 ;:
721 ;:
722 ;:
723 ;:
724 ;:
725 ;:
726 ;:
727 ;:
728 ;:
729 ;:
730 ;:
731 ;:
732 ;:
733 ;:
734 ;:
735 ;:
736 ;:
737 ;:
738 ;:
739 ;:
740 ;:
741 ;:
742 ;:
743 ;:
744 ;:
745 ;:
746 ;:
747 ;:
748 ;:
749 ;:
750 ;:
751 ;:
752 ;:
753 ;:
754 ;:
755 ;:
756 ;:
757 ;:
758 ;:
759 ;:
760 ;:
761 ;:
762 ;:
763 ;:
764 ;:
765 ;:
766 ;:
767 ;:
768 ;:
769 ;:
770 ;:
771 ;:
772 ;:
773 ;:
774 ;:
775 ;:
776 ;:
777 ;:
778 ;:
779 ;:
780 ;:
781 ;:
782 ;:
783 ;:
784 ;:
785 ;:
786 ;:
787 ;:
788 ;:
789 ;:
790 ;:
791 ;:
792 ;:
793 ;:
794 ;:
795 ;:
796 ;:
797 ;:
798 ;:
799 ;:
800 ;:
801 ;:
802 ;:
803 ;:
804 ;:
805 ;:
806 ;:
807 ;:
808 ;:
809 ;:
810 ;:
811 ;:
812 ;:
813 ;:
814 ;:
815 ;:
816 ;:
817 ;:
818 ;:
819 ;:
820 ;:
821 ;:
822 ;:
823 ;:
824 ;:
825 ;:
826 ;:
827 ;:
828 ;:
829 ;:
830 ;:
831 ;:
832 ;:
833 ;:
834 ;:
835 ;:
836 ;:
837 ;:
838 ;:
839 ;:
840 ;:
841 ;:
842 ;:
843 ;:
844 ;:
845 ;:
846 ;:
847 ;:
848 ;:
849 ;:
850 ;:
851 ;:
852 ;:
853 ;:
854 ;:
855 ;:
856 ;:
857 ;:
858 ;:
859 ;:
860 ;:
861 ;:
862 ;:
863 ;:
864 ;:
865 ;:
866 ;:
867 ;:
868 ;:
869 ;:
870 ;:
871 ;:
872 ;:
873 ;:
874 ;:
875 ;:
876 ;:
877 ;:
878 ;:
879 ;:
880 ;:
881 ;:
882 ;:
883 ;:
884 ;:
885 ;:
886 ;:
887 ;:
888 ;:
889 ;:
890 ;:
891 ;:
892 ;:
893 ;:
894 ;:
895 ;:
896 ;:
897 ;:
898 ;:
899 ;:
900 ;:
901 ;:
902 ;:
903 ;:
904 ;:
905 ;:
906 ;:
907 ;:
908 ;:
909 ;:
910 ;:
911 ;:
912 ;:
913 ;:
914 ;:
915 ;:
916 ;:
917 ;:
918 ;:
919 ;:
920 ;:
921 ;:
922 ;:
923 ;:
924 ;:
925 ;:
926 ;:
927 ;:
928 ;:
929 ;:
930 ;:
931 ;:
932 ;:
933 ;:
934 ;:
935 ;:
936 ;:
937 ;:
938 ;:
939 ;:
940 ;:
941 ;:
942 ;:
943 ;:
944 ;:
945 ;:
946 ;:
947 ;:
948 ;:
949 ;:
950 ;:
951 ;:
952 ;:
953 ;:
954 ;:
955 ;:
956 ;:
957 ;:
958 ;:
959 ;:
960 ;:
961 ;:
962 ;:
963 ;:
964 ;:
965 ;:
966 ;:
967 ;:
968 ;:
969 ;:
970 ;:
971 ;:
972 ;:
973 ;:
974 ;:
975 ;:
976 ;:
977 ;:
978 ;:
979 ;:
980 ;:
981 ;:
982 ;:
983 ;:
984 ;:
985 ;:
986 ;:
987 ;:
988 ;:
989 ;:
990 ;:
991 ;:
992 ;:
993 ;:
994 ;:
995 ;:
996 ;:
997 ;:
998 ;:
999 ;:
1000 ;:

```

```

545 003376 062767 000020 177112 ACALC: ADD #20, BEBD ;CALC. ADDR. OF BEBD TESTING
546 003404 062767 000020 177106 ADD #20, BECC ;CALC. ADDR. OF BECC TESTING
547 003412 062767 000020 177102 ADD #20, BEBA ;CALC. ADDR. OF BEBA TESTING
548 003420 062767 000020 177076 ADD #20, BECR1 ;CALC. ADDR. OF BECR1 TESTING
549 003426 062767 000020 177072 ADD #20, BECR2 ;CALC. ADDR. OF BECR2 TESTING
550 003434 062767 000020 177066 ADD #20, BERE ;CALC. ADDR. OF BERE TESTING
551 003442 062767 000004 177062 ADD #4, INTVEC ;CALC. ADDR. OF INTERRUPT VECTOR
552 003450 000241 CLC ;INIT. CARRY
553 003452 006267 177034 ASR TMAP ;LOOK FOR BIT INDICATING EXERCISOR
554 003456 042767 100000 177026 BIC #100000, TMAP ;CLEAR MSB IF SET
555 003464 103405 BCS C ;IF EXERCISOR PRESENT GO SEE IF TO BE TESTED
556 003466 005767 177020 TST TMAP ;ANY EXERCISORS LEFT?
557 003472 001341 BNE ACALC ;BRANCH IF MORE
558 003474 000167 010740 JMP LAST ;GO TO LAST TEST
559 003500 032767 000020 177006 C: BIT #20, SPTR ;TESTED 4 UBE?
560 003506 001402 BEQ D ;BRANCH IF NO
561 003510 000167 010724 JMP LAST ;GO TO LAST TEST
562 003514 036777 176774 175450 D: BIT SPTR, @SWR ;SHOULD THIS UBE BE TESTED?
563 003522 001403 BEQ E ;BRANCH IF YES
564 003524 006367 176764 ASL SPTR ;ROTATE POINTER TO NEXT SWITCH
565 003530 000722 BR ACALC ;LOOK FOR NEXT UBE
566 003532 006367 176756 E: ASL SPTR ;ROTATE POINTER TO NEXT SWITCH
567 003536 005267 177064 INC UCNT ;UPDATE COUNT OF UBE TESTED
568 003542 104401 027570 TYPE ,MSG13 ;TESTING UBE WITH BEBD ADDRESS:
569 003546 016746 176744 MOV BEBD, -(SP) ;SAVE BEBD FOR TYPEOUT
003552 104402 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
570 003554 104401 001245 TYPE ,$CRLF
571 ;////////////////////////////////////
572 ;ROUTINE TO STORE TEMPORARY ADDRESS OF UBE TESTING IN PERMANENT LOC
573 ;////////////////////////////////////
574 003560 016701 177044 MOV NO, R1 ;GET POINTER TO BE1BD
575 003564 012700 002516 MOV #BEBD, R0 ;GET POINTER FOR BEBD
576 003570 012021 F: MOV (R0)+, (R1)+ ;SAVE ADDRESSES
577 003572 020027 002534 CMP R0, #BEGU ;ALL SAVED?
578 003576 001374 BNE F ;BRANCH IF NO
579 003600 062767 000016 177022 ADD #16, NO ;UPDATE PTER TO NEXT UBE
580
581 003606 012767 003632 175324 MOV #FIRST, $LPADR ;INIT. SCOPE WHEN MORE THAN 1 UBE
582 003614 012767 003632 175320 MOV #FIRST, $LPERR ;INIT. SCOPE WHEN MORE THAN 1 UBE
583 003622 105067 175306 CLRB $TSTNM ;INIT. TEST NUMBER
584 003626 000005 RESET ;INIT. ALL UBE FOR LOOPS
585
586
587
588
589
590
591
592
593
;*****
;*TEST 1 TEST ALL UBE REG CAN BE CLEARED
;*
;*RO CONTAINS ADDRESS OF REG UNDER TEST
;*
;*IF THIS TEST FAILS, ALL FOLLOWING TESTS FOR THIS MODULE ARE ABORTED.
;*****
TST1: SCOPE
FIRST: MOV #STACK, SP ;RESTORE STACK
MOV #340, @#PSW ;LOCK OUT INTERRUPTS
MOV #STRAP, @#4 ;SET UP NSSYN TRAP
MOV #340, @#6 ;SET PSW PRIORITY =7
MOV #0, @BECR2 ;DO DATO TO CLEAR PB BIT IF SET
CLR @BERE ;CLEAR ERROR CONDITIONS
003630 000004
594 003632 012706 001100
595 003636 012737 000340 177776
596 003644 012737 004030 000004
597 003652 012737 000340 000003
598 003660 012777 000000 176640
599 003666 005077 176636
    
```

```

600 003672 016700 176620          MOV BEBD,R0          ;SETUP TO LOOK AT FIRST REG.
601 003676 005010          T01L01: CLR (R0)      ;CLR UBE REG
602 003700 020067 176620          CMP R0,BECR1        ;TESTING BECR1?
603 003704 001425          BEQ T01L04          ;BRANCH IF YES
604 003706 005710          TST (R0)            ;IS REG CLEARED?
605 003710 001421          BEQ T01L02          ;BRANCH IF YES
606 003712 010067 175276          T01L03: MOV R0,$REGO   ;SAVE FAILING ADDRESS
607 003716 011067 175274          MOV (R0),$REG1      ;SAVE BAD DATA
608 003722 104002          ERROR +^D2          ;FATAL ERROR:REG FAILED TO CLEAR
609 003724 020067 176576          CMP R0,BECR2        ;DID BECR2 FAIL?
610 003730 001006          BNE T01L06          ;BRANCH IF NO
611 003732 032777 020000 176566          BIT #20000,@BECR2  ;WAS CCOVF =1?
612 003740 001402          BEQ T01L06          ;BRANCH IF NO
613 003742 104401 027632          TYPE ,MSG14         ;DISREGARD BIT 13=1 OF BECR2
614 003746 004767 012460          T01L06: JSR PC,TERRPC ;TYPE PC OF ERROR MSG
615 003752 000433          BR T01L05           ;RESTORE TRAP
616 003754 005720          T01L02: TST (R0)+    ;INC ADDRESS
617 003756 000747          BR T01L01           ;CONTINUE LOOP
618 003760 022777 000200 176536          T01L04: CMP #200,@BECR1 ;ALL BITS IN BECR1 0 EXCEPT RDY?
619 003766 001351          BNE T01L03          ;BRANCH TO ERROR IF NO
620 003770 016700 176532          MOV BECR2,R0        ;INDICATE LOOKING AT BECR2
621 003774 005077 176530          CLR @BERE           ;RESET ERROR CONDITIONS
622 004000 005077 176522          CLR @BECR2          ;CLEAR BECR2
623 004004 032777 157777 176514          BIT #157777,@BECR2 ;IS BECR2 =0 EXCEPT CCOVF?
624 004012 001337          BNE T01L03          ;NO, TYPE ADDRESS AND DATA ERROR
625 004014 012737 000006 000004          MOV #6,@#4          ;RESTORE TRAP CATCHER
626 004022 005037 000006          CLR @#6
627 004026 000414          BR TST2             ;GO TO NEXT TEST
628
629 004030 011667 175160          STRAP: MOV (SP),$REGO ;SAVE PC FROM STACK
630 004034 104003          ERROR +^D3          ;FATAL ERROR:CPU DID NOT RECEIVE SSYN
631 004036 004767 012370          JSR PC,TERRPC       ;TYPE PC OF ERROR MSG
632
633 004042 012737 000006 000004          T01L05: MOV #6,@#4    ;RESTORE TRAP CATCHER
634 004050 005037 000006          CLR @#6
635 004054 000167 010354          JMP NUBE1           ;TEST NEXT UBE
636
646

```

```

:*****
:*TEST 2          TST 1,6,8,14 BECR1 & BITS 0-3,14 OF BECR2 CHANGE
:*
:*R2, R3 CONTAIN THE TRUE AND COMPLEMENT TEST DATA
:*R4 CONTAINS A POINTER TO THE REG ADDRESS BEING TESTED
:*R5 CONTAINS THE MASKED CONTENTS OF THE REG BEING TESTED
:*$TMP1 CONTAINS THE MASK FOR THE REG
:*
:*IF THIS TEST FAILS, ALL FOLLOWING TESTS FOR THIS MODULE ARE ABORTED
:*****

```

```

004060 000004
647 004062 012706 001100          TST2: SCOPE
648 004066 012737 000340 177776          MOV #STACK,SP      ;RESTORE STACK
649 004074 012702 052652          MOV #340,@#PSW     ;LOCK OUT INTERRUPTS
650 004100 012703 025324          MOV #52652,R2      ;SETUP TEST DATA BECR1
651 004104 012704 002524          MOV #25324,R3      ;SETUP COMP. TEST DATA BECR1
652 004110 005077 176414          MOV #BECR1,R4      ;LOAD ADDRESS PTER. FOR BECR1
653 004114 012767 177777 175104          CLR @BERE          ;CLEAR ERROR CONDITCNS
654 004122 016705 175100          T02L03: MOV #177777,$TMP1 ;LOAD MASK TO LOOK AT ALL BECR1
655 004126 011400          MOV $TMP1,R5       ;LOAD R5 WITH MASK
          MOV (R4),R0   ;GET ADDRESS OF BECR TESTING

```

```

657 004130 010210          MOV R2,(R0)          ;LOAD BECR WITH DATA
658 004132 011001          MOV (R0),R1         ;GET CONTENTS OF BECR
659 004134 005101          COM R1              ;ONLY LOOK AT BITS
660 004136 040105          BIC R1,R5           ;SET IN MASK =R5
661 004140 020502          CMP R5,R2           ;DATA OK?
662 004142 001424          BEQ T02L01          ;BRANCH IF YES
663 004144 011467 175044    T02L02: MOV (R4),$REG0    ;SAVE BECR ADDRESS
664 004150 011067 175042    MOV (R0),$REG1      ;SAVE BECR BAD DATA
665 004154 010267 175040    MOV R2,$REG2        ;SAVE GOOD DATA
666 004160 104006          ERROR +^D6         ;FATAL ERROR: CONTROL REG HELD WRONG DATA
667 004162 021467 176340    CMP (R4),BECR2      ;DID BECR2 FAIL?
668 004166 001006          BNE T02L04          ;BRANCH IF NO
669 004170 032777 020000 176330 BIT #20000,@BECR2    ;WAS CCOVF=1?
670 004176 001402          BEQ T02L04          ;BRANCH IF NO
671 004200 104401 027632    TYPE ,MSG14         ;DISREGARD BIT 13=1 OF BECR2
672 004204 004767 012222    T02L04: JSR PC,TERRPC ;TYPE PC OF ERROR MSG
673 004210 000167 010214    JMP NUBE            ;TEST NEXT UBE
674 004214 010302          T02L01: MOV R3,R2      ;XFER NEW TEST DATA
675 004216 010210          MOV R2,(R0)         ;LOAD BECR WITH COMP.DATA
676 004220 011001          MOV (R0),R1         ;GET CONTENTS OF BECR
677 004222 016705 175000    MOV $TMP1,R5        ;LOAD R5 WITH MASK
678 004226 005101          COM R1              ;ONLY LOOK AT BITS
679 004230 040105          BIC R1,R5           ;SET IN MASK =R5
680 004232 020502          CMP R5,R2           ;DATA OK?
681 004234 001343          BNE T02L02          ;BRANCH IF NO
682 004236 012702 040012    MOV #40012,R2       ;SETUP TEST DATA BECR2
683 004242 012703 000005    MOV #5,R3           ;SETUP COMP. TEST DATA BECR2
684 004246 012704 002526    MOV #BECR2,R4       ;LOAD ADDRESS PTER. FOR BECR2
685 004252 012767 157777 174746 MOV #157777,$TMP1   ;HAVE MASK LOOK AT ALL BECR2 EXECPT CCOVF
686 004260 020067 176242    CMP R0,BECR2        ;TESTED BECR2?
687 004264 001316          BNE T02L03          ;NO, BRANCH TO START TEST OF BECR2
688
696

```

```

:*****
:*TEST 3          FLOAT A '1' THROUGH BEBD, BECC, BEBA
:*
:*R0 CONTAINS A POINTER TO THE REG ADDRESS BEING TESTED
:*R1 CONTAINS TEST DATA
:*
:*IF THIS TEST FAILS, ALL FOLLOWING TESTS FOR THIS MODULE ARE ABORTED
:*****

```

```

004266 000004
697 004270 012706 001100          TST3:  SCOPE
698 004274 012737 000340 177776    MOV #STACK,SP       ;RESTORE STACK
699 004302 012700 002516          MOV #340,@#PSW      ;LOCK OUT INTERRUPTS
700 004306 012701 000001          MOV #BEBD,R0        ;GET BEBD ADDRESS PTER.
701 004312 010130          T03L04: MOV #1,R1    ;SETUP TEST DATA REG
702 004314 025001          T03L03: MOV R1,@(R0)+ ;PUT TEST DATA IN REG
703 004316 001413          CMP @-(R0),R1       ;TEST REG
704 004320 011067 174670          BEQ T03L01          ;BRANCH IF OK
705 004324 010167 174670          MOV (R0),$REG0      ;SAVE FAILING REG ADDRESS
706 004330 013067 174662          MOV R1,$REG2        ;SAVE GOOD DATA
707 004334 104004          MOV @(R0)+,$REG1    ;SAVE BAD DATA
708 004336 004767 012070          ERROR +^D4         ;FATAL ERROR:REG FAILED TO FLOAT A '1'
709 004342 000167 010062          JSR PC,TERRPC       ;TYPE PC OF ERROR MSG
710 004346 005701          JMP NUBE            ;TEST NEXT UBE
711 004350 100402          T03L01: TST R1      ;TESTED ALL 16 BITS?
712 004352 006301          BMI T03L02          ;BRANCH IF YES
                          ASL R1      ;TEST NEXT BIT

```

713 004354 000756
714 004356 022067 176140
715 004362 001351
716
724

T03L02: BR T03L03 ;CONTINUE LOOP
CMP (R0)+,BEBA ;TESTED LAST REG? ALSO UPDATE ADDR. PTER.
BNE T03L04 ;BRANCH IF REGS NOT TESTED

*TEST 4 FLOAT A '0' THROUGH BEBD, BECC, BEBA
*
*R0 CONTAINS A POINTER TO THE REG ADDRESS BEING TESTED
*R1 CONTAINS TEST DATA
*
*IF THIS TEST FAILS, ALL FOLLOWING TESTS FOR THIS MODULE ARE ABORTED

004364 000004
725 004366 012706 001100
726 004372 012737 000340 177776
727 004400 012700 002516
728 004404 012701 177776
729 004410 010130
730 004412 025001
731 004414 001413
732 004416 011067 174572
733 004422 010167 174572
734 004426 013067 174564
735 004432 104005
736 004434 004767 011772
737 004440 000167 007764
738 004444 005701
739 004446 100002
740 004450 006001
741 004452 000756
742 004454 022067 176042
743 004460 001351
744
756

TST4: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,@#PSW ;LOCK OUT INTERRUPTS
MOV #BEBD,R0 ;GET BEBD ADDRESS PTER.
T04L04: MOV #177776,R1 ;SETUP TEST DATA REG
T04L03: MOV R1,@(R0)+ ;PUT TEST DATA IN REG
CMP@-(R0),R1 ;TEST REG
BEQ T04L01 ;BRANCH IF OK
MOV (R0),\$REG0 ;SAVE FAILING REG ADDRESS
MOV R1,\$REG2 ;SAVE GOOD DATA
MOV @(R0)+,\$REG1 ;SAVE BAD DATA
ERROR +^D5 ;FATAL ERROR: REG FAILED TO FLOAT A '0'
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
JMP NUBE ;TEST NEXT UBE
T04L01: TST R1 ;TESTED ALL 16 BITS?
BPL T04L02 ;BRANCH IF YES
ROR R1 ;TEST NEXT BIT
BR T04L03 ;CONTINUE LOOP
T04L02: CMP(R0)+,BEBA ;TESTED LAST REG? ALSO UPDATE ADDR. PTER.
BNE T04L04 ;BRANCH IF REG NOT TESTED

*TEST 5 TEST FOR DUAL ADDRESSING IN REGS
*
*THIS TEST CLEARS ALL REGS AND THEN WRITES INTO THE
*REG BEING TESTED. ALL OTHER REGS ARE THEN CHECKED IF THEY WERE
*SIMULTANEOUSLY WRITTEN. THIS IS THEN REPEATED FOR ALL REGS.
*R0 CONTAINS ADDRESS OF REG BEING WRITTEN
*R1 CONTAINS ADDRESS OF REG BEING EXAMINED
*R2 CONTAINS MASK OF BITS TO BE LOOKED AT
*
*IF THIS TEST FAILS, ALL FOLLOWING TESTS FOR THIS MODULE ARE ABORTED

004462 000004
757 004464 012706 001100
758 004470 012737 000340 177776
759 004476 004767 011470
760 004502 016700 176010
761 004506 016701 176004
762 004512 012710 000002
763 004516 012702 177777
764 004522 030211
765 004524 001422
766 004526 020100
767 004530 001420

TST5: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,@#PSW ;LOCK OUT INTERRUPTS
JSR PC,CLRREG ;CLEAR ALL REG
MOV BEBD,R0 ;INITIALIZE TEST ADDRESS
T05L04: MOV BEBD,R1 ;INITIALIZE PTER.
MOV #2,(R0) ;LOAD TEST REG
MOV #177777,R2 ;INITIALIZE MASK TO LOOK AT ALL BITS
T05L03: BIT R2,(R1) ;IS DATA IN REG =0?
BEQ T05L01 ;BRANCH IF DATA OK(=0)
CMP R1,R0 ;LOOKING AT REG LOADED?
BEQ T05L01 ;BRANCH IF YES (DATA OK)

```
768 004532 020167 175766      CMP R1,BECR1      ;LOOKING AT BECR1?
769 004536 001411              BEQ T05L07        ;BRANCH IF YES
770 004540 010067 174450      T05L08: MOV R0,$REG0  ;ERROR: SAVE REG ADDRESS LOADED
771 004544 010167 174446      MOV R1,$REG1     ;SAVE REG ADDRESS EXAMINED
772 004550 104007              ERROR +^D7       ;FATAL ERROR: DUAL ADDRESSING ERROR
773 004552 004767 011654      JSR PC,TERRPC    ;TYPE PC OF ERROR MSG
774 004556 000167 007646      JMP NUBE         ;TEST NEXT UBE
775 004562 022777 000200 175734 T05L07: CMP #200,@BECR1 ;ALL BITS IN BECR1 0 EXCEPT RDY?
776 004570 001363              BNE T05L08        ;BRANCH IF NO
777 004572 020167 175730      T05L01: CMP R1,BECR2  ;LOOKED AT BECR2?
778 004576 001412              BEQ T05L02        ;BRANCH IF YES
779 004600 020167 175720      CMP R1,BECR1     ;PTER UP TO BECR1?
780 004604 001005              BNE T05L06        ;NO, LOOK AT NEXT REG
781 004606 016701 175714      MOV BECR2,R1     ;NOW LOOK AT BECR2
782 004612 012702 157777      MOV #157777,R2   ;LOOK AT ALL BECR2 EXCEPT CCOVF
783 004616 000741              BR T05L03         ;CONTINUE LOOKING
784 004620 005721              T05L06: TST (R1)+  ;UPDATE PTER.
785 004622 000737              BR T05L03         ;LOOK AT NEXT REG.
786 004624 004767 011342      T05L02: JSR PC,CLRREG ;CLEAR ALL REG
787 004630 020067 175672      CMP R0,BECR2     ;LOADED AND TESTED BECR2?
788 004634 001410              BEQ TST6         ;BRANCH IF YES TO NEXT TEST
789 004636 020067 175662      CMP R0,BECR1     ;LOADED BECR1 WITH DATA YET?
790 004642 001003              BNE T05L05        ;BRANCH IF NO
791 004644 016700 175656      MOV BECR2,R0     ;YES, NOW LOAD BECR2
792 004650 000716              BR T05L04         ;CONTINUE LOOKING
793 004652 005720              T05L05: TST(R0)+  ;UPDATE ADDRESS OF REG LOADED
794 004654 000714              BR T05L04         ;TEST THIS REG
795
807
```

```
:::*****
:*TEST 6          TEST BUS PARITY BIT PB
:*
:*THIS TEST IS NOT RUN ON THOSE MACHINE
:*WITH NO PARITY TRAP (11/05, 11/20)
:*
:*FOR OTHER MACHINES, THIS TEST SHOULD BE DESELECTED IF THE
:*MEMORY PARITY OPTION IS NOT PRESENT OR NOT ENABLED, ELSE
:*AN ERROR WILL BE REPORTED ALTHOUGH HARDWARE IS FUNCTIONING
:*PROPERLY.
:*SW05=1          INHIBIT TEST 6 AND GO TO NEXT TEST
:::*****
```

```
808 004656 000004              TST6:  SCOPE
809 004660 012706 001100      MOV #STACK,SP    ;RESTORE STACK
810
811 004664 032777 000040 174300 ;////////////////////
812 004672 001057              BIT #SW05, @SWR  ;INHIBIT TEST 6?
813                      BNE TST7          ;GO TO NEXT TEST
814                      ;ROUTINE TO DETERMINE IF RUNNING UNDER 11/05 OR 11/20
815                      ;IF 11/05 OR 11/20 BUSS PARITY TEST IS SKIPPED
816 004674 012737 004770 000010 ;////////////////////
817 004702 012737 000340 000012      MOV #ITRAP,@#10  ;SET UP TO GO TO NEXT TEST IF ILLEGAL INST TRAP
818 004710 006700              MOV #340,@#12
819                      SXT R0          ;IF INST TRAPS HAVE 11/05 OR 11/20
820 004712 012737 000340 177776      MOV #340,@#PSW   ;SET PSW PRIORITY=7
821 004720 012737 004754 000114      MOV #PTRAP,@#114 ;SET UP PARITY TRAP
822 004726 012737 000340 000116      MOV #340,@#116
823 004734 012777 010000 175564      MOV #10000,@BECR2 ;ENABLE PB PARITY
```



```

824 004742 005777 175560          TST @BECR2          ;START PARITY TRAP
825 004746 104010          ERROR +^D8         ;SETTING PB PARITY FAILED TO CAUSE CPU TO TRAP
826 004750 004767 011456          JSR PC,TERRPC     ;TYPE PC OF ERROR MSG
827 004754 012737 000116 000114 PTRAP: MOV #116,@#114   ;RESTORE TRAP CATCHER
828 004762 005037 000116          CLR @#116         ;RESTORE TRAP CATCHER
829 004766 000411          BR T06L01        ;SKIP MSG
830 004770 032777 010000 174174 ITRAP: BIT #SW12,@SWR  ;INHIBIT TYPEOUTS?
831 004776 001005          BNE T06L01       ;BRANCH IF YES
832 005000 012767 000001 174226          MOV #1,$TIMES     ;DO 1 ITERATION WHEN TEST NOT NOT RUN
833 005006 104401 020663          TYPE ,MSG5       ;BUS PARITY NOT TESTED ON 11/05 OR 11/20 MACHINES
834 005012 012737 000012 000010 T06L01: MOV #12,@#10 ;RESTORE TRAP CATCHER
835 005020 005037 000012          CLR @#12         ;RESTORE TRAP CATCHER
836 005024 012777 000000 175474          MOV #0,@BECR2    ;DO DATO TO CLEAR PB BIT
837
845

```

```

:*****
:*TEST 7          TST GO ,RDY SETS & CLRS,RELEASE BUS IMMED
:*
:*THE READY AND GO BIT ARE CHECKED USING A RELEASE
:*BUSS IMMEDIATE FUNCTION. FALSE INTERRUPT ARE CHECKED FOR
:*
:*IF THE GO OR READY BITS FAIL, ALL FOLLOWING TESTS FOR THIS MODULE ARE ABORTED.
:*****

```

```

005032 000004
846 005034 012706 001100          TST7:  SCOPE
847 005040 012737 000340 177776          MOV #STACK,SP    ;RESTORE STACK
848 005046 004767 011120          MOV #340,@#PSW  ;LOCK OUT INTERRUPTS
849 005052 012777 005172 175452          JSR PC,CLRREG   ;CLR ALL REG
850 005060 016700 175446          MOV #FINT1,@INTVEC ;SET UP FOR FALSE INTERRUPT
851 005064 012760 000340 000002          MOV INTVEC,R0   ;GET INTERRUPT VECTOR
852 005072 012777 006003 175424          MOV #340,2(R0)  ;SET PSW PRIORITY=7
853 005100 032777 000200 175416          MOV #6003,@BECR1 ;SET GO BIT AND DO RELEASE BUSS IMMEDIATE WITH BR4=1
854 005106 001035          BIT #200,@BECR1 ;LOOK AT RDY BIT
855 005110 005037 177776          BNE T07L08      ;BRANCH IF NOT CLEARED
856 005114 005000          CLR @#PSW       ;ALLOW INTERRUPTS
857 005116 005200          T07L07: CLR R0   ;INITIALIZE A COUNT TO WAIT FOR RDY=1
858 005120 022700 000011          T07L03: INC R0   ;UPDATE COUNT AND LOOP
859 005124 001416          CMP #11,R0      ;TILL COUNT=10 OR RDY=1
860 005126 105777 175372          BEQ T07L04     ;BRANCH IF RDY WAS NOT SET
861 005132 100371          TSTB @BECR1    ;READY SET?
862 005134 032777 000001 175362          BPL T07L03     ;CONTINUE TO LOOK FOR RDY
863 005142 001426          BIT #1,@BECR1  ;SEE IF GO BIT CLEARED
864 005144 004767 011054          BEQ T07L05     ;PROCEED TO NEXT TEST IF YES
865 005150 104013          JSR PC,RCATCH  ;RESTORE TRAP CATCHER
866 005152 004767 011254          ERROR +^D11    ;FATAL ERROR: GO BIT FAILED TO CLEAR
867 005156 000167 007246          JSR PC,TERRPC  ;TYPE PC OF ERROR MSG
868 005162 104014          JMP NUBE       ;TEST NEXT UBE
869 005164 004767 011242          T07L04: ERROR +^D12 ;FATAL ERROR: RDY BIT FAILED TO SET
870 005170 000407          JSR PC,TERRPC  ;TYPE PC OF ERROR MSG
871          BR T07L06    ;ABORT UBE TEST
872 005172 104123          FINT1: ERROR +^D83 ;ERROR: FALSE INTERRUPT WHEN DO RELEASE BUSS IMMED.
873 005174 004767 011232          JSR PC,TERRPC  ;TYPE PC OF ERROR MSG
874 005200 000745          BR T07L07     ;NOW CHECK IF RDY=1 AND GO BIT=0
875
876 005202 104020          T07L08: ERROR +^D16 ;FATAL ERROR: RDY BIT FAILED TO CLEAR OR GO DID NOT SET
877 005204 004767 011222          JSR PC,TERRPC  ;TYPE PC OF ERROR MSG
878 005210 004767 011010          T07L06: JSR PC,RCATCH ;RESTORE TRAP CATCHER
879 005214 000167 007210          JMP NUBE       ;TEST NEXT UBE

```

880 005220 004767 011000
881
902

T07L05: JSR PC,RCATCH ;RESTORE TRAP CATCHER

```

:*****
:*TEST 10 TEST UBE CAN INTERRUPT 4,7 NO SSYN BIT SET
:*
:*THE PSW PRIORITY IS FIRST SET EQUAL TO THE BR
:*LEVEL OF THE UBE. ALL LEVELS ARE FIRST CHECKED
:*THIS WAY. IF THE UBE FALSELY INTERRUPTS, A
:*SUBROUTINE, FINT3, WILL DETERMINE THE LEVEL IT
:*INTERRUPTED.
:*AFTER THIS, THE UBE IS ALLOWED TO INTERRUPT BY
:*SETTING THE PSW PRIORITY ONE LEVEL BELOW THE BR.
:*ALL LEVELS ARE THEN CHECKED THIS WAY. THE
:*PROPER INTERRUPT VECTOR IS TESTED FOR BY SETTING UP
:*THE ENTIRE VECTOR AREA 0-776 TO DETECT FOR WRONG
:*INTERRUPTS.
:*
:*NOTE: IF THIS TEST IS HALTED IN THE MIDDLE
:* AND IT IS DESIRED TO RESTART THE PROGRAM,
:* THE PROGRAM SHOULD BE RESTARTED AT 1100 AND
:* NOT AT 200.
:* TEST UBE CAN INTERRUPT 4,7,& NO INTERRUPT SSYN BIT DOESN T SET
:*****

```

005224 000004
903 005226 012706 001100
904 005232 012737 000340 177776
905 005240 004767 010726
906 005244 010667 173754
907 005250 005000
908 005252 012046
909 005254 022700 000060
910 005260 001374
911 005262 013746 000174
912 005266 013746 000176
913 005272 012737 000341 000002
914 005300 012700 000004
915 005304 012720 005716
916 005310 012720 000341
917 005314 022700 001000
918 005320 001371
919 005322 012777 005600 175202
920 005330 012767 000004 173656
921 005336 012767 000200 173652
922 005344 012777 000003 175152
923 005352 012737 000200 177776
924 005360 000240
925 005362 012767 000005 173624
926 005370 012767 000240 173620
927 005376 012737 000240 177776
928 005404 012777 000005 175112
929 005412 000240
930 005414 012767 000006 173572
931 005422 012767 000300 173566
932 005430 012737 000300 177776
933 005436 012777 000011 175060
934 005444 000240
935

```

TST10: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,@#PSW ;LOCK OUT INTERRUPTS
JSR PC,CLRREG ;CLEAR ALL UBE REG
MOV SP,$TMP0 ;SAVE STACK ADDRESS
CLR R0 ;INIT. R0
T08L08: MOV (R0)+,-(SP) ;SAVE VECTOR AREA 0-56
CMP #60,R0 ;ALL SAVED?
BNE T08L08 ;BRANCH IF NO
MOV @#174,-(SP) ;SAVE SOFTWARE SWR
MOV @#176,-(SP)
MOV #341,@#2 ;SET UP VECTOR AREA TO DETECT WRONG INT. VECTORS
MOV #4,R0 ;INITIALIZE ADDRESS REG
T08L01: MOV #WINT,(R0)+ ;PUT WRONG INTERRUPT PTER IN ALL VECTOR LOCATIONS
MOV #341,(R0)+ ;PUT AN ODD PSW IN ALL PSW LOCATIONS
CMP #1000,R0 ;AT END OF VECTOR AREA?
BNE T08L01 ;BRANCH IF NO
MOV #FINT3,@INTVEC ;SET UP UBE VECTOR AREA FOR FALSE INT.
MOV #4,$REG0 ;INDICATE DOING BR=4
MOV #200,$REG1 ;INDICATE PSW PRIORITY=4
MOV #3,@#BECCR1 ;HAVE UBE DO BR=4
MOV #200,@#PSW ;SET PRIORITY=4
NOP ;UBE SHOULD NOT INTERRUPT HERE
MOV #5,$REG0 ;INDICATE DOING BR=5
MOV #240,$REG1 ;INDICATE PSW PRIORITY=5
MOV #240,@#PSW ;SET PRIORITY=5
MOV #5,@#BECCR1 ;HAVE UBE DO BR=5
NOP ;UBE SHOULD NOT INTERRUPT HERE
MOV #6,$REG0 ;INDICATE DOING BR=6
MOV #300,$REG1 ;INDICATE PRIORITY=6
MOV #300,@#PSW ;SET PRIORITY=6
MOV #11,@#BECCR1 ;HAVE UBE DO BR=6
NOP ;UBE SHOULD NOT INTERRUPT HERE

```

```

936                                     ;NOW TEST UBE WILL INTERRUPT WITH PRIORITY ONE LEVEL BELOW BR
937
938 005446 012777 000002 175050          MOV #2,@BECR1          ;INITIALIZE UBE TO DO BR=4
939 005454 012767 000004 173532          MOV #4,$REG0          ;INITIALIZE INDICATOR FOR BR=4
940 005462 012767 000003 173526          MOV #3,$REG1          ;INITIALIZE INDICATOR FOR PRIORITY=3
941 005470 012777 005552 175034          MOV #T08L02,@INTVEC  ;SET RETURN ADDRESS WHEN GET PROPER INTERRUPT
942 005476 012737 000140 177776          MOV #140,@#PSW        ;INITIALIZE PSW PRIORITY=3
943 005504 000240                          NOP                    ;UBE SHOULD INTERRUPT HERE
944 005506 000413                          BR T08L09              ;BRANCH TO ERROR IF NO INT.
945 005510 005267 173500          T08L03: INC $REG0        ;INDICATE BR LEVEL DOING
946 005514 005267 173476          INC $REG1              ;INDICATE PSW PRIORITY LEVEL DOING
947 005520 000257                          CCC                    ;CLEAR N,Z,V,C
948 005522 062737 000040 177776          ADD #40,@#PSW         ;SET PRIORITY LEVEL BELOW BR LEVEL
949 005530 005277 174770          INC @BECR1            ;HAVE UBE DO BR 1 LEVEL ABOVE PRIORITY
950 005534 000240                          NOP                    ;UBE SHOULD INTERRUPT HERE
951 005536 004767 010554          T08L09: JSR PC,RVEC     ;RESTORE TRAP CATCHER AND HANDLER
952 005542 104021                          ERROR +^D17           ;ERROR: UBE FAILED TO INTERRUPT
953 005544 004767 010662          JSR PC,TERRPC         ;TYPE PC OF ERROR MSG
954 005550 000472                          BR T08L06             ;BRANCH TO TEST NO INT. SSYN ERROR BIT
955 005552 022626          T08L02: CMP (SP)+,(SP)+ ;RESTORE STACK AFTER INTERRUPT
956 005554 032777 000020 174742          BIT #20,@BECR1        ;TESTED LAST BR?
957 005562 001063                          BNE T08L07           ;BRANCH IF YES TO TEST NO INT. SSYN ERROR BIT
958 005564 006377 174734          ASL @BECR1            ;SHIFT BECR1 FOR NEXT BR LEVEL
959 005570 042777 000400 174726          BIC #400,@BECR1      ;CLEAR SHIFTED RDY BIT
960 005576 000744                          BR T08L03            ;GO TEST NEXT BR
961
962 005600 022626          FINT3:  CMP (SP)+,(SP)+ ;RESTORE STACK AFTER INTERRUPT
963 005602 004767 010510          JSR PC,RVEC           ;RESTORE VECTOR AREA
964 005606 104022                          ERROR +^D18           ;ERROR: UBE INT. WHEN PSW AT SAME PRIORITY LEVEL
965 005610 004767 010616          JSR PC,TERRPC         ;TYPE PC OF ERROR MSG
966 005614 032777 007740 174704          BIT #7740,@BECR2     ;SEE IF ERROR CONDITION OCCURRED IN BECR2
967 005622 001407                          BEQ T08L04           ;BRANCH IF NO
968 005624 017767 174676 173362          MOV @BECR2,$REG0     ;SAVE ERROR CONDITIONS
969 005632 104017                          ERROR +^D15           ;ERROR: ERROR BITS IN BECR2 SET WHEN SHOULD=0
970 005634 004767 010572          JSR PC,TERRPC         ;TYPE PC OF ERROR MSG
971 005640 000445                          BR TST11             ;BRANCH TO NEXT TEST
972
973 005642 012777 005650 174662          T08L04: MOV #T08L05,@INTVEC ;SET UP INTVEC TO FIND BR LEVEL UBE MADE
974 005650 012706 001100          T08L05: MOV #STACK,SP  ;RESTORE STACK
975 005654 062767 000040 173334          ADD #40,$REG1        ;RAISE PRIORITY LEVEL BY 1
976 005662 005267 173326          INC $REG0              ;INDICATE NEW LEVEL OF PRIORITY
977 005666 016737 173324 177776          MOV $REG1,@#PSW      ;SET PSW PRIORITY
978 005674 005277 174624          INC @BECR1            ;HAVE UBE INTERRUPT AGAIN
979 005700 000240                          NOP                    ;IF UBE INT. HERE, INCREMENT PRIORITY
980 005702 004767 010316          JSR PC,RCATCH         ;RESTORE TRAP CATCHER
981 005706 104023                          ERROR +^D19           ;ERROR: UBE FALSELY INTERRUPTED AT HIGHER LEVEL
982 005710 004767 010516          JSR PC,TERRPC         ;TYPE PC OF ERROR MSG
983 005714 000417                          BR TST11             ;BRANCH TO NEXT TEST
984
985 005716 022626          WINT:  CMP (SP)+,(SP)+ ;RESTORE STACK AFTER INTERRUPT
986 005720 004767 010372          JSR PC,RVEC           ;RESTORE VECTOR AREA
987 005724 104024                          ERROR +^D20           ;ERROR: UBE INTERRUPTED TO WRONG VECTOR
988 005726 004767 010500          JSR PC,TERRPC         ;TYPE PC OF ERROR MSG
989 005732 004767 010360          T08L07: JSR PC,RVEC     ;RETURN VECTOR AREA WHEN FINISH BR TEST
990 005736 032777 004000 174562          T08L06: BIT #4000,@BECR2 ;WAS NO INT. SSYN ERROR BIT SET?
991 005744 001403                          BEQ TST11            ;BRANCH TO NEXT TEST IF NO
992 005746 104027                          ERROR +^D23           ;ERROR: NO INT. SSYN BIT FALSELY SET
  
```

993 005750 004767 010456
994
1001

JSR PC,TERRPC ;TYPE PC OF ERROR MSG

:TEST 11 TEST THE NO,NO SACK ERROR BIT DOESN'T SET
:
:*THE INHIBIT SACK BIT IS SET AND THE UBE IS TOLD TO
:*DO A FUN. 3.THE NO,NO SACK ERROR BIT IS THEN
:*CHECKED TO NOT HAVE SET.
:*****

005754 000004
1002 005756 012706 001100
1003 005762 012737 000340 177776
1004 005770 004767 010176
1005 005774 012777 000010 174524
1006 006002 012777 006003 174514

TST11: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,@#PSW ;LOCK OUT INTERRUPTS
JSR PC,CLRREG ;CLEAR ALL UBE REGS.
MOV #10,@BECR2 ;ENABLE INH SACK IN BECR2
MOV #6003,@BECR1 ;DO FUN 3 VIA BR4

1008 006010 005037 177776
1009 006014 000240

CLR @#PSW
NOP

;ALLOW INTERRUPTS
;ALLOW UBE TO GET BUSS. CPU SHOULD TIME OUT

2

1011									
1012	006016	005000			CLR R0				:INIT COUNTER
1013	006020	005200		1\$:	INC R0				:INC COUNTER
1014	006022	105700			TSTB R0				:DELAY AT LEAST 41 USEC
1015	006024	100375			BPL 1\$:BRANCH IF NO
1016	006026	032777	000200	174472	BIT #200,@BECR2				:WAS NO, NO SACK BIT SET?
1017	006034	001403			BEQ RTR				:BRANCH IF NO
1018	006036	104026			ERROR +^D2?				:ERROR: NO, NO SACK BIT FALSELY SET

1020 006040 004767 010366
1021 006044 004767 010122
1022
1031

RTR: JSR PC,TERRPC ;TYPE PC OF ERROR MSG
JSR PC,CLRREG ;CLEAR ALL UBE REG

*TEST 12 TEST DATI,DATIP,DATO,DATOB AND FUNCTION 1 WORKS
*
*ALL DATA TRANSFERS ARE DONE VIA BR TRANSFERS.
*EACH OPERATION (DATI, DATO, DATIP, DATOB) DOES ONE
*TRANSFER AND THE DATA IS THEN CHECKED.
*EACH TIME AN OPERATION IS STARTED THE READY
*BIT IS TESTED BY THE SUBROUTINE 'RDYS' TO SEE IF IT SETS.

006050 000004
1032 006052 012706 001100
1033 006056 012737 000340 177776
1034 006064 012767 052525 021770
1035 006072 004767 010074
1036 006076 012777 177777 174414
1037 006104 012777 030062 174410
1038 006112 012705 006620
1039 006116 012777 002003 174400
1040 006124 005037 177776
1041 006130 004767 000434
1042 006134 022777 052525 17435/
1043 006142 001421
1044 006144 017767 174346 173042
1045 006152 016767 021704 173036
1046 006160 012767 030062 173032
1047 006166 012767 052525 173026
1048 006174 104030
1049 006176 004767 010230
1050 006202 000167 000450
1051 006206 004767 007760
1052 006212 005067 021644
1053 006216 012777 177777 174274
1054 006224 012777 030062 174270
1055 006232 012777 052525 174256
1056 006240 012705 006630
1057 006244 012777 003003 174252
1058 006252 004767 000312
1059 006256 022767 052525 021576
1060 006264 001420
1061 006266 017767 174224 172720
1062 006274 016767 021562 172714
1063 006302 012767 030062 172710
1064 006310 012767 052525 172704
1065 006316 104031
1066 006320 004767 010106
1067 006324 000554
1068
1069 006326 004767 007640
1070 006332 012767 052525 021522
1071 006340 012777 177777 174152
1072 006346 012777 030062 174146
1073 006354 012705 006640
1074 006360 012777 002403 174136
1075 006366 004767 000176

TST12: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,@#PSW ;LOCK OUT INTERRUPTS
MOV #052525,BUFF1 ;PUT TEST DATA IN BUFFER
JSR PC,CLRREG ;CLEAR ALL UBE REG
MOV #177777,@BECC ;HAVE UBE DO 1 XFER
MOV #BUFF1,@BEBA ;LOAD UBE WITH BUFFER ADDRESS
MOV #ERR1,R5 ;INITIALIZE R5 FOR ERROR ADDRESS
MOV #2003,@BECC1 ;HAVE UBE DO DATI VIA BR=4 AND FUNCTION 1
CLR @#PSW ;ALLOW DATA XFER
JSR PC,RDYS ;GO CHECK FOR RDY TO SET
CMP #052525,@BEBD ;IS DATA OK?
BEQ T10L01 ;GO TEST DATO IF YES
MOV @BEBD,\$REG0 ;SAVE (BEBD)
MOV BUFF1,\$REG1 ;SAVE MEM DATA
MOV #BUFF1,\$REG2 ;SAVE MEM ADDRESS
MOV #52525,\$REG3 ;SAVE CORRECT DATA
ERROR +^D24 ;ERROR: DATI FAILED TO LOAD PROPER DATA
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
JMP TSTA ;GO TO NEXT TEST
T10L01: JSR PC,CLRREG ;CLEAR UBE REG
CLR BUFF1 ;CLEAR TEST AREA
MOV #177777,@BECC ;HAVE UBE DO 1 XFER
MOV #BUFF1,@BEBA ;LOAD UBE WITH BUFFER ADDRESS
MOV #052525,@BEBD ;LOAD UBE WITH DATA
MOV #ERR2,R5 ;INITIALIZE R5 FOR ERROR ADDRESS
MOV #3003,@BECC1 ;HAVE UBE DO DATO VIA BR=4 AND FUNCTION 1
JSR PC,RDYS ;GO CHECK FOR RDY TO SET
CMP #052525,BUFF1 ;WAS BUFFER LOADED PROPERLY?
BEQ T10L02 ;GO TEST DATIP IF YES
MOV @BEBD,\$REG0 ;SAVE (BEBD)
MOV BUFF1,\$REG1 ;SAVE MEM DATA
MOV #BUFF1,\$REG2 ;SAVE MEM ADDRESS
MOV #052525,\$REG3 ;SAVE CORRECT DATA
ERROR +^D25 ;ERROR: DATO FAILED TO LOAD PROPER DATA
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
BR TST13 ;BRANCH TO NEXT TEST
T10L02: JSR PC,CLRREG ;CLEAR UBE REG
MOV #052525,BUFF1 ;PUT TEST DATA IN BUFFER
MOV #177777,@BECC ;HAVE UBE DO 1 XFER
MOV #BUFF1,@BEBA ;LOAD UBE WITH BUFFER ADDRESS
MOV #ERR3,R5 ;INITIALIZE R5 FOR ERROR ADDRESS
MOV #2403,@BECC1 ;HAVE UBE DO DATIP VIA BR=4 AND FUNCTION 1
JSR PC,RDYS ;GO CHECK FOR RDY SET

```

1076 006372 022777 125252 174116      CMP #125252,@BEBD      ;HAS UBE SHIFTED DATA?
1077 006400 001004                      BNE T10L06            ;BRANCH IF NO
1078 006402 022767 125252 021452      CMP #125252,BUFF1    ;HAS MEM LOC BEEN SHIFTED?
1079 006410 001420                      BEQ T10L03            ;GO TEST DATOB IF YES
1080 006412 017767 174100 172574      T10L06: MOV @BEBD,$REG0    ;SAVE (BEBD)
1081 006420 016767 021436 172570      MOV BUFF1,$REG1      ;SAVE MEM DATA
1082 006426 012767 030062 172564      MOV #BUFF1,$REG2    ;SAVE MEM ADDRESS
1083 006434 012767 125252 172560      MOV #125252,$REG3    ;SAVE CORRECT DATA
1084 006442 104032                      ERROR +^D26          ;ERROR: DATIP FAILED TO LOAD PROPER DATA
1085 006444 004767 007762              JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1086 006450 000502                      BR TST13             ;:BRANCH TO NEXT TEST
1087
1088 006452 012767 000377 021402      T10L03: MOV #377,BUFF1    ;INITIALIZE BUFFER
1089 006460 012705 006650              MOV #ERR4,R5         ;INITIALIZE R5 FOR ERROR ADDRESS
1090 006464 012777 177400 174024      MOV #177400,@BEBD    ;LOAD HIGH BYTE OF UBE WITH 1'S
1091 006472 012777 030063 174022      MOV #BUFF1+1,@BEBA   ;LOAD HIGH BYTE BUFF ADDR. INTO UBE
1092 006500 012777 177777 174012      MOV #177777,@BECC    ;HAVE UBE DO 1 XFER
1093 006506 012777 003403 174010      MOV #3403,@BECCR1    ;HAVE UBE DO DATOB VIA BR=4 AND FUNCTION 1
1094 006514 004767 000050              JSR PC,RDYS          ;GO CHECK FOR RDY SET
1095 006520 022767 177777 021334      CMP #177777,BUFF1    ;TEST IF DATOB DONE CORRECTLY
1096 006526 001453                      BEQ TST13             ;:BRANCH IF YES TO NEXT TEST
1097
1098 006530 017767 173762 172456      MOV @BEBD,$REG0      ;SAVE (BEBD)
1099 006536 016767 021320 172452      MOV BUFF1,$REG1      ;SAVE NEW DATA
1100 006544 012767 030062 172446      MOV #BUFF1,$REG2     ;SAVE MEM ADDRESS
1101 006552 012767 177777 172442      MOV #177777,$REG3    ;SAVE CORRECT DATA
1102 006560 104033                      ERROR +^D27          ;ERROR: DATOB FAILED TO LOAD DATA PROPERLY
1103 006562 004767 007644              JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1104 006566 000433                      BR TST13             ;:BRANCH TO NEXT TEST
1105
1106 ;SUBROUTINE TO TEST IF RDY BIT SET
1107
1108 006570 005004                      RDYS: CLR R4           ;INITIALIZE R4
1109 006572 032777 000200 173724      T10L05: BIT #200,@BECCR1 ;IS RDY SET?
1110 006600 001006                      BNE T10L04           ;BRANCH IF YES
1111 006602 005204                      INC R4               ;UPDATE COUNT
1112 006604 032704 000020              BIT #20,R4           ;COUNT=16?
1113 006610 001770                      BEQ T10L05           ;IF NO, GO TEST RDY AGAIN
1114 006612 005726                      TST (SP)+            ;RETURN STACK PTER
1115 006614 000115                      JMP (R5)             ;GO INDICATE ERROR
1116 006616 000207                      T10L04: RTS PC        ;RETURN AND CHECK DATA
1117 006620 104034                      ERR1: ERROR +^D28     ;ERROR: DATI FAILED TO SET RDY
1118 006622 004767 007604              JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1119 006626 000413                      BR TST13             ;:GO TO NEXT TEST
1120 006630 104035                      ERR2: ERROR +^D29     ;ERROR: DATO FAILED TO SET RDY
1121 006632 004767 007574              JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1122 006636 000407                      BR TST13             ;:GO TO NEXT TEST
1123 006640 104036                      ERR3: ERROR +^D30     ;ERROR: DATIP FAILED TO SET RDY
1124 006642 004767 007564              JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1125 006646 000403                      BR TST13             ;:GO TO NEXT TEST
1126 006650 104037                      ERR4: ERROR +^D31     ;ERROR: DATOB FAILED TO SET RDY
1127 006652 004767 007554              JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1128
1129 006656                      TSTA:
1130
1131 ;*****
; *TEST 13 TEST INHIBIT DATA SHIFT ON DATIP

```



```

006656 000004
1132 006660 012706 001100
1133 006664 004767 007302
1134 006670 005037 177776
1135 006674 012767 052525 021160
1136 006702 012777 177777 173610
1137 006710 012777 030062 173604
1138 006716 012777 022403 173600
1139 006724 004767 007316
1140 006730 005704
1141 006732 001404
1142 006734 104036
1143 006736 004767 007470
1144 006742 000427
1145 006744 022777 052525 173544
1146 006752 001004
1147 006754 022767 052525 021100
1148 006762 001417
1149 006764 017767 173526 172222
1150 006772 016767 021064 172216
1151 007000 012767 030062 172212
1152 007006 012767 052525 172206
1153 007014 104040
1154 007016 004767 007410
1155
1156

```

```

*****
TST13: SCOPE
MOV #STACK,SP ;RESTORE STACK
JSR PC,CLRREG ;CLEAR UBE REG
CLR @PSW ;ALLOW INTERRUPTS
MOV #052525,BUFF1 ;PUT TEST DATA IN BUFFER
MOV #177777,@BECC ;HAVE UBE DO 1 XFER
MOV #BUFF1,@BEBA ;LOAD UBE WITH BUFFER ADDRESS
MOV #22403,@BECC1 ;HAVE UBE DO DATIP WITH INH DATA SHIFT
JSR PC,CRDY ;CHECK FOR RDY BIT
TST R4 ;DID RDY SET?
BEQ T11L01 ;BRANCH IF YES
ERROR +^D30 ;ERROR: DATIP FAILED TO SET RDY
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
BR TST14 ;BRANCH TO NEXT TEST
T11L01: CMP #052525,@BEBD ;IS (BEBD) OK?
BNE T11L02 ;BRANCH IF NO
CMP #052525,BUFF1 ;IS MEM OK?
BEQ TST14 ;BRANCH IF YES TO NEXT TEST
T11L02: MOV @BEBD,$REG0 ;SAVE (BEBD)
MOV BUFF1,$REG1 ;SAVE MEM DATA
MOV #BUFF1,$REG2 ;SAVE MEM ADDRESS
MOV #052525,$REG3 ;SAVE CORRECT DATA
ERROR +^D32 ;ERROR: INH. DATA SHIFT ON DATIP FAILED
JSR PC,TERRPC ;TYPE PC OF ERROR MSG

```

```

*****
*TEST 14 TEST DATOB ON DATIP
*****

```

```

007022 000004
1157 007024 012706 001100
1158 007030 004767 007136
1159 007034 005037 177776
1160 007040 012767 177525 021014
1161 007046 012777 030062 173446
1162 007054 012777 177777 173436
1163 007062 012777 042403 173434
1164 007070 004767 007152
1165 007074 022777 177253 173414
1166 007102 001004
1167 007104 022767 177653 020750
1168 007112 001417
1169 007114 017767 173376 172072
1170 007122 016767 020734 172066
1171 007130 012767 030062 172062
1172 007136 012767 177653 172056
1173 007144 104041
1174 007146 004767 007260
1175
1176
1188

```

```

*****
TST14: SCOPE
MOV #STACK,SP ;RESTORE STACK
JSR PC,CLRREG ;CLEAR UBE REG
CLR @PSW ;ALLOW INTERRUPTS
MOV #177525,BUFF1 ;LOAD TEST DATA IN BUFFER
MOV #BUFF1,@BEBA ;LOAD UBE WITH LOW BYTE ADDRESS
MOV #177777,@BECC ;HAVE UBE DO 1 XFER
MOV #42403,@BECC1 ;HAVE UBE DO DATOB ON DATIP
JSR PC,CRDY ;CHECK FOR RDY SET
CMP #177253,@BEBD ;CHECK (BEBD) OK
BNE T12L01 ;BRANCH IF NO
CMP #177653,BUFF1 ;CHECK BUFFER OK
BEQ TST15 ;BRANCH IF YES TO NEXT TEST
T12L01: MOV @BEBD,$REG0 ;SAVE (BEBD)
MOV BUFF1,$REG1 ;SAVE MEM DATA
MOV #BUFF1,$REG2 ;SAVE MEM ADDRESS
MOV #177653,$REG3 ;SAVE CORRECT DATA
ERROR +^D33 ;ERROR: DATOB ON DATIP FAILED
JSR PC,TERRPC ;TYPE PC OF ERROR MSG

```

```

*****
*TEST 15 TEST NO SSYN ERROR BIT WORK
*
*A DATI NPR IS DONE TO A MEM LOC (760000) THAT RETURNS
*NO SSYN. THE NO SSYN ERROR BIT AND BIT 15 OF BECC1
*ARE CHECKED TO SET. THE ERROR INTERRUPT IS THEN TESTED.
*AFTER THIS THE ERROR IS CLEARED BY THE CLEAR ERROR

```

```

:*ADDRESS. FINALLY THE FUN A,B BITS (BITS 10,11 OF BECR1)
:*ARE EXAMINED TO SEE IF THEY RESET WHEN AN ERROR
:*INTERRUPT OCCURS.
:* TEST NO SSYN ERROR BIT WORKS & FUN A,B BITS RESET ERROR INTRRUPT
:*****
  
```

```

TST15: SCOPE
1189 007152 000004          MOV #STACK,SP          ;RESTORE STACK
1190 007154 012706 001100  MOV #340,@#PSW         ;LOCK OUT INTERRUPTS
1191 007166 004767 007000  JSR PC,CLRREG         ;CLEAR UBE REG
1192 007172 012777 007320 253332  MOV #T23L01,@INTVEC   ;SET UP FOR INTERRUPTS
1193 007200 012777 160000 253314  MOV #160000,@BEBA     ;LOAD UBE WITH TEST ADDRESS WHICH RETURNS NO SSYN
1194 007206 012777 000003 253312  MOV #3,@BECR2        ;LOAD UBE WITH TEST ADDRESS WHICH RETURNS NO SSYN
1195 007214 012777 177777 253276  MOV #177777,@BECC    ;HAVE UBE DO 1 CYCLE
1196 007222 012777 002041 253274  MOV #2041,@BECR1     ;HAVE DATI NPR DONE
1197 007230 004767 007012  JSR PC,CRDY          ;WAIT TILL RDY SET
1198 007234 032777 000400 253264  BIT #400,@BECR2      ;WAS NSSYN ERROR BIT SET?
1199 007242 001004          BNE T23L02           ;BRANCH IF YES
1200 007244 104073          ERROR +^D59         ;ERROR: TEST OF NSSYN ERROR BIT FAILED
1201 007246 104074          ERROR +^D60         ;TO SET BIT 8 OF BECR2
1202 007250 004767 007156  JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1203 007254 032777 100000 253242 T23L02: BIT #100000,@BECR1 ;WAS ERROR BIT SET?
1204 007262 001004          BNE T23L03           ;BRANCH IF YES
1205 007264 104073          ERROR +^D59         ;ERROR: TEST OF NSSYN ERROR BIT FAILED
1206 007266 104075          ERROR +^D61         ;TO SET BIT 15 OF BECR1
1207 007270 004767 007136  JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1208 007274 005037 177776  T23L03: CLR @#PSW     ;ALLOW UBE TO INTERRUPT
1209 007300 000240          NOP                 ;UBE SHOULD INTERRUPT HERE
1210 007302 017767 173220 251704  MOV @BECR2,$REGO     ;SAVE BECR2
1211 007310 104073          ERROR +^D59         ;ERROR: TEST OF NSSYN ERROR BIT FAILED
1212 007312 104072          ERROR +^D58         ;TO INTERRUPT CPU
1213 007314 004767 007112  JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1214 007320 005077 173204  T23L01: CLR @BERE     ;CLEAR ERROR BITS
1215 007324 032777 000400 253174  BIT #400,@BECR2      ;WAS NSSYN ERROR BIT CLEARED?
1216 007332 001404          BEQ T23L05          ;BRANCH IF YES TO TEST FUN A, B BITS
1217 007334 104073          ERROR +^D59         ;ERROR: TEST OF NSSYN ERROR BIT FAILED
1218 007336 104076          ERROR +^D62         ;TO CLEAR BIT 8 OF BECR2
1219 007340 004767 007066  JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1220 007344 032777 002000 253152 T23L05: BIT #2000,@BECR1 ;WAS FUN A BIT RESET?
1221 007352 001404          BEQ T23L06          ;BRANCH IF YES
1222 007354 104073          ERROR +^D59         ;ERROR: TEST OF NSSYN ERROR BIT FAILED
1223 007356 104016          ERROR +^D14         ;TO CLEAR BIT 10 OF BECR1
1224 007360 004767 007046  JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1225 007364 012777 160000 253130 T23L06: MOV #160000,@BEBA ;LOAD UBE WITH TEST ADDRESS WHICH RETURNS NO SSYN
1226 007372 012777 000003 253126  MOV #3,@BECR2        ;LOAD UBE WITH TEST ADDRESS WHICH RETURNS NO SSYN
1227 007400 012777 177772 253112  MOV #177772,@BECC    ;DO 2 CYCLES
1228 007406 012777 007426 253116  MOV #T23L07,@INTVEC ;SET UP FOR INT
1229 007414 012777 004041 253102  MOV #4041,@BECR1     ;HAVE UBE DO FUN2 DATI VIA NPR
1230 007422 004767 006620  JSR PC,CRDY          ;WAIT TILL RDY SETS
1231 007426 032777 004000 253070 T23L07: BIT #4000,@BECR1 ;WAS FUN B BIT RESET
1232 007434 001404          BEQ T23L04          ;RESTORE TRAP
1233 007436 104073          ERROR +^D59         ;ERROR: TEST OF NSSYN ERROR BIT FAILED
1234 007440 104105          ERROR +^D69         ;TO CLEAR BIT 11 OF BECR1
1235 007442 004767 006764  JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1236 007446 004767 006552  T23L04: JSR PC,RCATCH ;RESTORE TRAP
1237 007452 005077 173052  CLR @BERE           ;CLEAR ALL ERROR CONDITIONS
1238
1239
  
```

1244

```
*****  
*TEST 16 TEST ADDRESS REG COUNTS BY 2 AND 1  
*  
*RO CONTAINS THE TEST DATA  
*****
```

```
TST16: SCOPE  
MOV #STACK,SP ;RESTORE STACK  
JSR PC,CLRREG ;CLEAR UBE REGS  
JSR PC,DINT ;DISREGARD UBE INTERRUPTS  
CLR @PSW ;ALLOW INTERRUPTS  
MOV #2,R0 ;INITIALIZE TEST COUNTER  
1245 007456 000004  
1246 007460 012706 001100  
1247 007464 004767 006502  
1248 007470 004767 006602  
1249 007500 012700 000002  
1250 007504 012777 177777 173006 T14L02: MOV #177777,@BECC ;HAVE UBE DO 1 XFER  
1251 007512 012777 002003 173004 MOV #2003,@BECR1 ;HAVE UBE DO DATI  
1252 007520 004767 006522 JSR PC,CRDY ;CHECK RDY SET  
1253 007524 020077 172772 CMP R0,@BEBA ;IS ADDRESS CORRECT?  
1254 007530 001057 BNE T14L01 ;BRANCH TO ERROR IF NO  
1255 007532 005200 INC R0 ;UPDATE R0  
1256 007534 005200 INC R0 ;UPDATE R0  
1257 007536 022700 000002 CMP #2,R0 ;HAVE ALL ADDRESSES BEEN TESTED?  
1258 007542 001360 BNE T14L02 ;LOOK AT NEXT ADDRESS IF NO  
1259 007544 012777 177776 172750 MOV #177776,@BEBA ;LOAD MAX ADDRESS IN LOWER 16 BITS UBE  
1260 007552 012777 000003 172746 MOV #3,@BECR2 ;LOAD A16,A17 OF UBE WITH 1  
1261 007560 012777 177777 172732 MOV #177777,@BECC ;HAVE UBE DO 1 XFER  
1262 007566 005277 172732 INC @BECR1 ;HAVE UBE DO DATI  
1263 007572 004767 006450 JSR PC,CRDY ;CHECK RDY SET  
1264 007576 032777 000003 172722 BIT #3,@BECR2 ;TEST A16,A17=0  
1265 007604 001042 BNE T14L03 ;BRANCH TO ERROR IF NO  
1266  
1267 ;NOW TEST ADDRESS COUNTS BY 1  
1268  
1269 007606 012777 030063 172706 MOV #BUFF1+1,@BEBA ;PUT ODD ADD OF BUFFER IN UBE  
1270 007614 012777 177777 172676 MOV #177777,@BECC ;HAVE UBE DO 1 XFER  
1271 007622 012777 003403 172674 MOV #3403,@BECR1 ;HAVE UBE DO DATOB  
1272 007630 004767 006412 JSR PC,CRDY ;CHECK RDY  
1273 007634 022777 030064 172660 CMP #BUFF1+2,@BEBA ;DID ADDRESS UPDATE BY 1?  
1274 007642 001434 BEQ T14L04 ;BRANCHIF YES TO RESTORE TRAPS  
1275 007644 017767 172652 171342 MOV @BEBA,$REG0 ;SAVE BAD ADDRESS  
1276 007652 012767 030064 171336 MOV #BUFF1+2,$REG1 ;SAVE GOOD ADDRESS  
1277 007660 104045 ERROR +^D37 ;ERROR: BEBA DID NOT COUNT BY 1  
1278 007662 004767 006544 JSR PC,TERRPC ;TYPE PC OF ERROR MSG  
1279 007666 000422 BR T14L04 ;GO TO RESTORE TRAPS  
1280 007670 017767 172626 171316 T14L01: MOV @BEBA,$REG0 ;SAVE BAD ADDRESS  
1281 007676 010067 171314 MOV R0,$REG1 ;SAVE CORRECT ADDRESS  
1282 007702 104043 ERROR +^D35 ;ERROR: BEBA DID NOT COUNT BY 2  
1283 007704 004767 006522 JSR PC,TERRPC ;TYPE PC OF ERROR MSG  
1284 007710 000411 BR T14L04 ;GO TO RESTORE TRAPS  
1285 007712 017700 172610 T14L03: MOV @BECR2,R0 ;GET ADDRESS BITS FROM UBE  
1286 007716 042700 177774 BIC #177774,R0 ;JUST LOOK AT A16,A17  
1287 007722 010067 171266 MOV R0,$REG0 ;SAVE ADDRESS  
1288 007726 104044 ERROR +^D36 ;ERROR: BEBA BITS A16,A17 DID NOT COUNT = 0  
1289 007730 004767 006476 JSR PC,TERRPC ;TYPE PC OF ERROR MSG  
1290 007734 004767 006264 T14L04: JSR PC,RCATCH ;RESTORE TRAPS AND GO TO NEXT TEST  
1291  
1299
```

```
*****  
*TEST 17 TEST BUS ADDRESS BITS WILL CHANGE  
*  
*THE UBE BUS ADDRESS BITS ARE CHECKED TO
```

;*SEE IF THEY CAN CHANGE FROM 0,1. SEVERAL DATIS
 :*ARE DONE FROM LOCATION 0, THE HIGHEST LOC IN THE FIRST
 :*8K AND FROM THE UBE SIMULTANEOUS (IO ADDRESS.

007740 000004
 1300 007742 012706 001100
 1301 007746 004767 006220
 1302 007752 004767 006320
 1303 007756 005037 177776
 1304
 1305
 1306
 1307 007762 012737 010012 000004
 1308 007770 012700 017776
 1309 007774 062700 004000
 1310 010000 005710
 1311 010002 022700 037776
 1312 010006 001372
 1313 010010 000402
 1314 010012 162700 004000
 1315
 1316 010016 012737 000006 000004
 1317 010024 011001
 1318 010026 010010
 1319 010030 012737 000000 000000
 1320 010036 012777 177777 172454
 1321 010044 012777 002003 172452
 1322 010052 004767 006170
 1323 010056 005777 172434
 1324 010062 001034
 1325 010064 010077 172432
 1326 010070 012777 177777 172422
 1327 010076 005277 172422
 1328 010102 004767 006140
 1329 010106 020077 172404
 1330 010112 001020
 1331 010114 016777 172414 172400
 1332 010122 012777 000003 172376
 1333 010130 012777 177777 172362
 1334 010136 005277 172362
 1335 010142 004767 006100
 1336 010146 005777 172344
 1337 010152 001411
 1338 010154 017767 172342 171032
 1339 010162 162767 000002 171024
 1340 010170 104042
 1341 010172 004767 006234
 1342 010176 004767 006022
 1343 010202 010110
 1344
 1351

TST17: SCOPE
 MOV #STACK,SP ;RESTORE STACK
 JSR PC,CLRREG ;CLEAR UBE REG
 JSR PC,DINT ;DISREGARD INTERRUPTS
 CLR @#PSW ;ALLOW DATA TRANSFERS

;SIZE MEMORY FROM 4K TO 8K

T13L02: MOV #T13L01,@#4 ;SET UP TIME OUT TRAP
 MOV #17776,R0 ;SET R0=LAST ADDRESS IN 1ST 4K OF MEM
 ADD #4000,R0 ;UPDATE R0 TO NEXT 1K OF MEM
 TST (R0) ;TEST IF 1K PRESENT. TIMES OUT IF NOT.
 CMP #37776,R0 ;AT 8K?
 BNE T13L02 ;LOOK AT NEXT 1K IF NOT

T13L01: SUB #4000,R0 ;GET ADDRESS OF LAST 1K OF MEM PRESENT

T13L03: MOV #6,@#4 ;RESTORE TRAP
 MOV (R0),R1 ;SAVE CONTENTS OF LAST LOC IN FIRST 8K
 MOV R0,(R0) ;PUT ADDRESS OF LOC IN MEM LOC
 MOV #0,@#0 ;PUT 0 IN LOC 0
 MOV #177777,@BECC ;HAVE UBE DO 1 XFER
 MOV #2003,@BECCR1 ;HAVE UBE DO DATI FROM MEM LOC 0
 JSR PC,CRDY ;CHECK FOR RDY SET
 TST @BEBD ;SEE IF UBE READ 0 FROM LOC 0
 BNE T13L04 ;BRANCH TO ERROR IF DATA NOT = 0
 MOV R0,@BEBA ;HAVE UBE ADDRESS HIGHEST MEMORY IN 4K-8K LOCATIONS
 MOV #177777,@BECC ;HAVE UBE DO 1 XFER
 INC @BECCR1 ;HAVE UBE DO DATI FROM HIGHEST MEMORY IN 4K-8K LOCATIONS
 JSR PC,CRDY ;CHECK FOR RDY SET
 CMP R0,@BEBD ;DID UBE READ FROM PROPER LOCATION?
 BNE T13L04 ;BRANCH IF DATA NOT = R0
 MOV BEGO,@BEBA ;HAVE UBE ADDRESS ITS GO ADDRESS
 MOV #3,@BECCR2 ;HAVE UBE ADDRESS ITS GO ADDRESS
 MOV #177777,@BECC ;HAVE UBE DO 1 XFER
 INC @BECCR1 ;HAVE UBE DO DATI FROM GO ADDRESS
 JSR PC,CRDY ;CHECK FOR RDY SET
 TST @BEBD ;DID UBE READ PROPER LOCATION?
 BEQ T13L05 ;BRANCH IF YES

T13L04: MOV @BEBA,\$REGO ;GET ADDRESS+2 TRIED TO READ FROM
 SUB #2,\$REGO ;CALC. ADDRESS TRIED TO READ FROM
 ERROR +*D34 ;ERROR: UBE DID DATI FROM WRONG LOCATION

T13L05: JSR PC,TERRPC ;TYPE PC OF ERROR MSG
 MOV R1,(R0) ;RESTORE TRAPS
 ;RESTORE CONTENTS OF LAST LOC OF FIRST 8K

 ;*TEST 20 TEST CYCLE COUNTS BY 1 AND INC WITH EACH INT
 ;*
 ;*THE BECC REG IS CYCLED FROM 0 TO 177777 BY INTERRUPTING THE
 ;*CPU. AFTER EACH INTERRUPT, THE REG IS COMPARED WITH R0 WHICH
 ;*CONTAINS THE PROPER DATA.
 ;*****

1352 010204 000004 001100
1353 010206 012706 000340 177776
1354 010212 012737 000340
1355 010220 004767 005746
1356 010224 005000
1357 010226 012777 010250 172276
1358 010234 012777 000003 172262
1359 010242 005037 177776
1360 010246 000240
1361 010250 022626
1362 010252 005200
1363 010254 005700
1364 010256 001423
1365 010260 020077 172234
1366 010264 001763
1367 010266 017767 172226 170720
1368 010274 010067 170716
1369 010300 104046
1370 010302 004767 006124
1371 010306 012737 000340 177776
1372 010314 012777 006003 172202
1373 010322 005037 177776
1374 010326 004767 005672
1380

TST20: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,@#PSW ;LOCK OUT INTERRUPTS
JSR PC,CLRREG ;CLEAR UBE REG
CLR R0 ;INITIALIZE TEST COUNTER
MOV #T15L01,@INTVEC ;SET UP INT VECTOR AREA
T15L03: MOV #3,@BECC1 ;HAVE UBE INT.VIA BR=4
CLR @#PSW ;ALLOW INTERRUPTS
NOP ;UBE WILL INTERRUPT HERE
T15L01: CMP (SP)+,(SP)+ ;RESTORE STACK AFTER INTERRUPT
INC R0 ;UPDATE TEST COUNTER
TST R0 ;IS R0=0?
BEQ T15L02 ;RESTORE TRAPS IF YES
CMP R0,@BECC ;DID CYCLE COUNT UPDATE PROPERLY?
BEQ T15L03 ;INCREMENT BECC IF YES
MOV @BECC,\$REG0 ;SAVE BAD DATA
MOV R0,\$REG1 ;SAVE GOOD DATA
ERROR +*D38 ;ERROR: INTERRUPT FAILED TO UPDATE BECC TO CORRECT VALUE
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
MOV #340,@#PSW ;LOCK OUT INTERRUPTS
MOV #6003,@BECC1 ;HAVE UBE CYCLE SO IT SETS RDY
CLR @#PSW ;ALLOW UBE TO CYCLE
T15L02: JSR PC,RCATCH ;RESTORE TRAPS

::*****
:*TEST 21 TEST INHIBIT INCREMENT OF BECC AND BEBA
:*
:*A DATI IS DONE VIA BR ARBITRATION AND THE BECC AND BEBA REGS
:*ARE CHECKED TO NOT INCREMENT.
::~*****

1381 010332 000004 001100
1382 010334 012706 000340 177776
1383 010340 012737 000340
1384 010346 004767 005620
1385 010352 012777 030062 172142
1386 010360 012777 177777 172132
1387 010366 012767 000001 017466
1388 010374 012777 000004 172124
1389 010402 012777 002003 172114
1390 010410 005037 177776
1391 010414 005777 172076
1392 010420 001775
1393 010422 022777 177777 172070
1394 010430 001010
1395 010432 022777 030062 172062
1396 010440 001407
1397 010442 104047
1398 010444 004767 005762
1399 010450 000403
1400 010452 104050
1401 010454 004767 005752
1402 010460 042777 000004 172040
1403 010466 004767 005554
1412

TST21: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,@#PSW ;LOCK OUT INTERRUPTS
JSR PC,CLRREG ;CLEAR UBE REG
MOV #BUFF1,@BEBA ;LOAD UBE WITH TEST ADDRESS
MOV #177777,@BECC ;LOAD TEST DATA INTO BECC
MOV #1,BUFF1 ;SETUP BUFFER DATA
MOV #4,@BECC2 ;HAVE UBE INH. INC. OF BECC AND BEBA
MOV #2003,@BECC1 ;HAVE UBE DO DATI FROM BUFFER AREA
CLR @#PSW ;ALLOW DATA XFER
T16L01: TST @BEBD ;WAS DATA XFERED?
BEQ T16L01 ;WAIT TILL DATA IN BEBD
CMP #177777,@BECC ;CHECK BECC WAS NOT UPDATED
BNE T16L02 ;BRANCH IF WAS TO ERROR
CMP #BUFF1,@BEBA ;CHECK BEBA WAS NOT UPDATED
BEQ T16L03 ;BRANCH IF WAS NOT UPDATED
ERROR +*D39 ;ERROR: BEBA INCREMENTED WHEN IT WAS INHIBITED
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
BR T16L03
T16L02: ERROR +*D40 ;ERROR: BECC INCREMENTED WHEN IT WAS INHIBITED
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
T16L03: BIC #4,@BECC2 ;ALLOW BEBA AND BECC TO COUNT
JSR PC,CRDY ;WAIT TILL UBE IS DONE

::*****
:*TEST 22 TEST INTERRUPT ENABLE & CCOVF WORKS
:*

;*THE UBE IS SETUP TO DO 4 DATO XFERS VIA BR ARBITRATION AND
 ;*INTERRUPT WHEN DONE. THE INTERRUPT IS CHECKED FOR
 ;*AND THEN A BUFFER AREA IS TESTED TO SEE IF EXACTLY
 ;*FOUR TRANSFERS WERE DONE.
 ;* TEST INTERRUPT ENABLE & CCOVF WORKS UBE WILL DO SEVERAL XFERS
 ;*****

010472	000004				TST22: SCOPE	
1413	010474	012706	001100		MOV #STACK,SP	;RESTORE STACK
1414	010500	012737	000340	177776	MOV #340,@#PSW	;LOCK OUT INTERRUPTS
1415	010506	004767	005460		JSR PC,CLRREG	;CLEAR UBE REG
1416	010512	012700	030062		MOV #BUFF1,R0	;GET BUFFER ADDRESS
1417	010516	005020			T17L01: CLR (R0)+	;CLEAR BUFFER AREA
1418	010520	020027	030102		CMP R0,#BUFF1+20	;AT END OF BUFFER?
1419	010524	001374			BNE T17L01	;BRANCH IF NO
1420	010526	012777	000377	171762	MOV #377,@BEED	;SET UP XFER TEST DATA
1421	010534	012777	030062	171760	MOV #BUFF1,@BEBA	;LOAD UBE WITH BUFF ADDRESS
1422	010542	012777	177774	171750	MOV #177774,@BECC	;SET UBE TO DO 4 XFERS
1423	010550	012777	010612	171754	MOV #T17L02,@INTVEC	;SET UP INT VECTOR
1424	010556	012777	003121	171740	MOV #3121,@BECR1	;HAVE UBE DO DATO VIA BR=7 AND INTERRUPT ON DONE
1425	010564	005037	177776		CLR @#PSW	;ALLOW XFERS
1426	010570	005000			CLR R0	;INITIALIZE COUNT
1427	010572	005200			T17L03: INC R0	;UPDATE COUNT TO WAIT FOR INTERRUPT
1428	010574	022700	001000		CMP #1000,R0	;WAITED LONG ENOUGH?
1429	010600	001374			BNE T17L03	;BRANCH IF NO
1430	010602	104051			ERROR +*D41	;ERROR: UBE FAILED TO INT. ON DONE
1431	010604	004767	005622		JSR PC,TERRPC	;TYPE PC OF ERROR MSG
1432	010610	000470			BR T17L09	;GO RESTORE TRAPS
1433	010612	012700	030062		T17L02: MOV #BUFF1,R0	;GET START OF BUFFER
1434	010616	005720			T17L05: TST (R0)+	;TEST FIRST 4 LOC WRITTEN
1435	010620	001433			BEQ T17L04	;BRANCH IF NOT WRITTEN TO ERROR
1436	010622	022700	030072		CMP #BUFF1+10,R0	;LOOKED AT ALL WRITTEN LOCS.
1437	010626	001373			BNE T17L05	;BRANCH IF NO
1438	010630	005720			T17L06: TST (R0)+	;TEST LAST 4 LOC WERE NOT WRITTEN
1439	010632	001027			BNE T17L10	;BRANCH TO ERROR IF WERE
1440	010634	022700	030102		CMP #BUFF1+20,R0	;AT END OF BUFFER?
1441	010640	001373			BNE T17L06	;NO, LOOK AT NEXT LOCATION
1442	010642	032777	000100	171654	BIT #100,@BECR1	;YES, TEST INT. ON DONE BIT=0
1443	010650	001041			BNE T17L07	;BRANCH TO ERROR IF NOT=0
1444	010652	032777	020000	171646	BIT #20000,@BECR2	;TEST CCOVF=1
1445	010660	001441			BEQ T17L08	;BRANCH TO ERROR IF=0
1446	010662	012777	006003	171634	MOV #6003,@BECR1	;SET GO BIT TO SEE IF CCOVF IS RESET
1447	010670	032777	020000	171630	BIT #20000,@BECR2	;TEST CCOVF=0
1448	010676	001435			BEQ T17L09	;GO RESTORE TRAPS IF YES
1449	010700	104052			ERROR +*D42	;ERROR: CCOVF NOT CLEARED BY GO
1450	010702	004767	005524		JSR PC,TERRPC	;TYPE PC OF ERROR MSG
1451	010706	000431			BR T17L09	;GO RESTORE TRAPS
1452	010710	005740			T17L04: TST -(R0)	;CALC. LAST ADD. WRITTEN
1453	010712	005740			T17L10: TST -(R0)	;CALC. LAST ADD. WRITTEN
1454	010714	022700	030062		CMP #BUFF1,R0	;WERE ANY ADD. WRITTEN?
1455	010720	003404			BLE T17L11	;BRANCH IF YES
1456	010722	104067			ERROR +*D55	;ERROR: UBE DID NOT DO DATO TO PROPER # OF LOC (4)
1457	010724	004767	005502		JSR PC,TERRPC	;TYPE PC OF ERROR MSG
1458	010730	000420			BR T17L09	;GO RESTORE TRAPS
1459	010732	012767	030062	170254	T17L11: MOV #BUFF1,\$REG0	;SAVE FIRST LOCATION WRITTEN
1460	010740	010067	170252		MOV R0,\$REG1	;SAVE LAST LOCATION WRITTEN
1461	010744	104053			ERROR +*D43	;ERROR: UBE DID NOT DO DATO TO PROPER # OF LOCATIONS (4)
1462	010746	004767	005460		JSR PC,TERRPC	;TYPE PC OF ERROR MSG

1463 010752 000407
1464 010754 104054
1465 010756 004767 005450
1466 010762 000403
1467 010764 104055
1468 010766 004767 005440
1469 010772 004767 005226
1470
1476

BR T17L09 ;:GO RESTORE TRAPS
T17L07: ERROR +^D44 ;:ERROR: INT. ON DONE BIT NOT CLEARED
JSR PC,TERRPC ;:TYPE PC OF ERROR MSG
BR T17L09 ;:GO RESTORE TRAPS
T17L08: ERROR +^D45 ;:ERROR: CCOVF NOT SET
JSR PC,TERRPC ;:TYPE PC OF ERROR MSG
T17L09: JSR PC,RCATCH ;:RESTORE TRAPS

*TEST 23 TEST DATA XFERS FROM BECC
*
*THE UBE IS SET UP TO DO 4 DATO XFERS VIA BR ARBITRATION FROM
*THE BECC REG TO A BUFFER AREA. THE AREA IS THEN CHECKED.

010776 000004
1477 011000 012706 001100
1478 011004 004767 005162
1479 011010 005037 177776
1480 011014 012700 030062
1481 011020 005020
1482 011022 020027 030102
1483 011026 001374
1484 011030 012777 030062 171464
1485 011036 012777 177774 171454
1486 011044 012777 013003 171452
1487 011052 032777 000200 171444
1488 011060 001774
1489 011062 012700 030062
1490 011066 012701 177774
1491 011072 022001
1492 011074 001005
1493 011076 005201
1494 011100 020027 030072
1495 011104 001372
1496 011106 000412
1497
1498 011110 005740
1499 011112 010067 170076
1500 011116 011067 170074
1501 011122 010167 170072
1502 011126 104056
1503 011130 004767 005276
1504
1515

TST23: SCOPE
MOV #STACK,SP ;:RESTORE STACK
JSR PC,CLRREG ;:CLEAR UBE REG
CLR @#PSW ;:ALLOW INTERRUPTS
MOV #BUFF1,R0 ;:GET BUFFER ADDRESS
T18L01: CLR (R0)+ ;:CLEAR BUFFER AREA
CMP R0,#BUFF1+20 ;:AT END OF BUFFER?
BNE T18L01 ;:BRANCH IF NO
MOV #BUFF1,@BEBA ;:LOAD STARTING ADDRESS INTO UBE
MOV #177774,@BECC ;:SETUP UBE TO DO 4 XFERS
MOV #13003,@BECC1 ;:HAVE UBE DO 4 XFERS FROM BECC
T18L02: BIT #200,@BECC1 ;:LOOK FOR RDY SET
BEQ T18L02 ;:BRANCH TILL SET
MOV #BUFF1,R0 ;:GET BUFFER ADDRESS
MOV #177774,R1 ;:INITIALIZE R1=TO FIRST DATA WORD
T18L04: CMP (R0)+,R1 ;:IS DATA OK?
BNE T18L03 ;:NO, GO TO ERROR
INC R1 ;:UPDATE FOR NEXT DATA
CMP R0,#BUFF1+10 ;:LOOKED AT ALL DATA?
BNE T18L04 ;:NO, LOOK AT NEXT WORD
BR TST24 ;:GO TO NEXT TEST
T18L03: TST -(R0) ;:CALC. ADDRESS OF FAILURES
MOV R0,\$REG0 ;:SAVE ADDRESS
MOV (R0),\$REG1 ;:SAVE BAD DATA
MOV R1,\$REG2 ;:SAVE GOOD DATA
ERROR +^D46 ;:ERROR: DATO FROM BECC NOT DONE PROPERLY
JSR PC,TERRPC ;:TYPE PC OF ERROR MSG

*TEST 24 TEST UBE CAN DO 2 XFERS PER BUS REQUEST
*
*THE UBE IS SET UP TO DO 2 DATO XFERS PER REQUEST VIA
*BR ARBITRATION. THE CYCLE COUNT IS SET TO DO A TOTAL OF
*FOUR XFERS. THE UBE IS TOLD TO GO. THE FIRST TIME
*THE CPU GETS THE BUS, AFTER THIS, THE PSW PRIORITY IS
*SET FOR 7 HOLDING OFF FURTHER UBE ACTION. A BUFFER
*AREA IS THEN CHECKED THAT THE UBE DID EXACTLY 2 XFERS
*PER REQUEST.

011134 000004
1516 011136 012706 001100
1517 011142 012737 000340 177776

TST24: SCOPE
MOV #STACK,SP ;:RESTORE STACK
MOV #340,@#PSW ;:LOCK ON INTERRUPTS

```

1518 011150 004767 005016 JSR PC,CLRREG ;CLEAR UBE REGS
1519 011154 012700 030062 MOV #BUFF1,R0 ;GET BUFFER ADDRESS
1520 011160 005020 T19L01: CLR (R0)+ ;CLEAR BUFFER AREA
1521 011162 020027 030102 CMP R0,#BUFF1+20 ;AT END OF BUFFER?
1522 011166 001374 BNE T19L01 ;CONTINUE TO CLEAR IF NO
1523 011170 012777 030062 171324 MOV #BUFF1,@BEBA ;LOAD BUFFER ADDRESS INTO UBE
1524 011176 012777 177774 171314 MOV #177774,@BECC ;SET UBE TO DO 4 XFERS
1525 011204 012777 000377 171304 MOV #377,@BEBD ;LOAD TEST DATA INTO UBE
1526 011212 012777 005003 171304 MOV #5003,@BECR1 ;HAVE UBE DO 2 DATO/REQUEST VIA BR=4
1527 011220 005037 177776 CLR @#PSW ;ALLOW UBE TO DO XFERS
1528 011224 000240 NOP ;UBE SHOULD DO 2 XFERS HERE
1529 011226 012737 000340 177776 MOV #340,@#PSW ;SET PRIORITY=7 TO STOP LAST 2 XFERS
1530 011234 012700 030062 MOV #BUFF1,R0 ;GET BUFF ADDRESS
1531 011240 005720 T19L03: TST (R0)+ ;WAS BUFF WRITTEN?
1532 011242 001411 BEQ T19L09 ;BRANCH TO ERROR IF NO
1533 011244 020027 030066 CMP R0,#BUFF1+4 ;LOOKED AT FIRST 2 LOCATIONS?
1534 011250 001373 BNE T19L03 ;BRANCH IF NO
1535 011252 005720 T19L04: TST (R0)+ ;TEST BUFF LOC NOT WRITTEN
1536 011254 001005 BNE T19L02 ;BRANCH TO ERROR IF WRITTEN
1537 011256 020027 030072 CMP R0,#BUFF1+10 ;LOOKED AT FOURTH LOC?
1538 011262 001373 BNE T19L04 ;BRANCH IF NO
1539 011264 000421 BR T19L05 ;GO TO END OF TEST
1540 011266 005740 T19L09: TST -(R0) ;CALC LAST ADDRESS WRITTEN
1541 011270 005740 T19L02: TST -(R0) ;CALC LAST ADDRESS WRITTEN
1542 011272 022700 030062 CMP #BUFF1,R0 ;WERE ANY ADDRESS WRITTEN?
1543 011276 101404 BLOS T19L07 ;BRANCH IF YES
1544 011300 104060 ERROR +^D48 ;ERROR: UBE DID NOT DO 2 XFERS/REQUEST
1545 011302 004767 005124 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
1546 011306 000410 BR T19L05 ;GO TO END OF TEST
1547 011310 012767 030062 167676 T19L07: MOV #BUFF1,$REGO ;SAVE FIRST ADDRESS WRITTEN
1548 011316 010067 167674 MOV R0,$REG1 ;SAVE LAST ADDRESS WRITTEN
1549 011322 104057 ERROR +^D47 ;ERROR: UBE DID NOT DO 2 XFERS FOR EACH REQUEST
1550 011324 004767 005102 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
1551 011330 005037 177776 T19L05: CLR @#PSW ;ALLOW LAST 2 XFERS
1552 011334 000240 NOP ;ALLOW UBE TO GET BUS
1553 011336 004767 004704 JSR PC,CRDY ;WAIT TILL UBE FINISHES XFERS
1554
1565

```

```

*****
*TEST 25 TEST UBE CAN DO 2 DATIP XFERS PER REQUEST
*
*THE UBE IS SET UP TO DO 2 DATIP XFERS PER REQUEST VIA
*BR ARBITRATION. THE CYCLE COUNT IS SET TO DO A TOTAL OF
*FOUR XFERS. THE UBE IS TOLD TO GO. THE FIRST TIME
*THE CPU GETS THE BUS, AFTER THIS, THE PSW PRIORITY IS
*SET FOR 7 HOLDING OFF FURTHER UBE ACTION. A BUFFER
*AREA IS THEN CHECKED THAT THE UBE DID EXACTLY 2 XFERS
*PER REQUEST.
*****

```

```

011342 000004
1566 011344 012706 001100 TST25: SCOPE ;RESTORE STACK
1567 011350 012737 000340 177776 MOV #340,@#PSW ;LOCK OUT INTERRUPTS
1568 011356 004767 004610 JSR PC,CLRREG ;CLEAR UBE REG
1569 011362 012700 030062 MOV #BUFF1,R0 ;GET BUFFER ADDRESS
1570 011366 012720 125252 T20L01: MOV #125252,(R0)+ ;LOAD TEST DATA
1571 011372 020027 030072 CMP R0,#BUFF1+10 ;LOADED FIRST 4 LOCS?
1572 011376 001373 BNE T20L01 ;BRANCH IF NO
1573 011400 012777 030062 171114 MOV #BUFF1,@BEBA ;LOAD BUFFER ADDRESS INTO UBE

```


1574	011406	012777	177774	171104		MOV #177774,@BECC	:SET UBE TO DO 4 CYCLES
1575	011414	012777	004403	171102		MOV #4403,@BECR1	:HAVE UBE DO 2 DATIP/REQUEST VIA BR=4
1576	011422	005037	177776			CLR @#PSW	:ALLOW UBE TO DO CYCLES
1577	011426	000240				NOP	:UBE SHOULD DO XFRS HERE
1578	011430	012737	000340	177776		MOV #340,@#PSW	:SET PRIORITY = 7 TO STOP LAST 2 CYCLES
1579	011436	012700	030062			MOV #BUFF1,R0	:GET BUFF ADDRESS
1580	011442	022720	052525		T20L03:	CMP #052525,(R0)+	:TEST BUFF LOCS WRITTEN
1581	011446	001012				BNE T20L02	:BRANCH TO ERROR IF NOT DONE PROPERLY
1582	011450	022700	030066			CMP #BUFF1+4,R0	:LOOKED AT 2 WRITTEN LOCS?
1583	011454	001372				BNE T20L03	:BRANCH IF NO
1584	011456	022720	125252		T20L04:	CMP #125252,(R0)+	:TEST BUFF LOCS NOT WRITTEN
1585	011462	001005				BNE T20L08	:BRANCH TO ERROR IF WRITTEN
1586	011464	020027	030072			CMP R0,#BUFF1+10	:LOOKED AT FOURTH LOC?
1587	011470	001372				BNE T20L04	:BRANCH IF NO
1588	011472	000421				BR T20L05	::GO TO END OF TEST
1589	011474	005740			T20L02:	TST -(R0)	:CALC LAST ADDRESS WRITTEN
1590	011476	005740			T20L08:	TST -(R0)	:CALC LAST ADDRESS WRITTEN
1591	011500	022700	030062			CMP #BUFF1,R0	:WERE ANY LOC WRITTEN?
1592	011504	101404				BLOS T20L06	:BRANCH IF YES
1593	011506	104061				ERROR +^D49	:ERROR: DID NOT DO 2 DATIP/REQUEST
1594	011510	004767	004716			JSR PC,TERRPC	:TYPE PC OF ERROR MSG
1595	011514	000410				BR T20L05	::GO TO END OF TEST
1596	011516	012767	030062	167470	T20L06:	MOV #BUFF1,\$REGO	:SAVE FIRST ADDRESS WRITTEN
1597	011524	010067	167466			MOV R0,\$REG1	:SAVE LAST ADDRESS WRITTEN
1598	011530	104062				ERROR +^D50	:ERROR: UBE DID NOT DO 2 DATIP/REQUEST
1599	011532	004767	004674			JSR PC,TERRPC	:TYPE PC OF ERROR MSG
1600	011536	005037	177776		T20L05:	CLR @#PSW	:ALLOW LAST 2 CYCLES
1601	011542	000240				NOP	:ALLOW UBE TO GET BUS
1602	011544	004767	004476			JSR PC,CRDY	:WAIT FOR UBE TO FINISH XFRS

```

1603
1613
::*****
:*TEST 26      TEST DATA XFRS VIA NPR AND INT ON DONE WORK
:*
:*THIS IS THE FIRST TEST WHERE THE NPR IS EXERCISED.  ONE
:*DATO NPR IS DONE TO A BUFFER AREA.  THE READY BIT IS
:*THEN CHECKED FOR SETTING.  NEXT, THE SAME OPERATION IS
:*REPEATED ONLY THE INTERRUPT ON DONE BIT IS SET.
:*THE PROGRAM TESTS FOR THE INTERRUPT AND THEN EXAMINES
:*THE BUFFER AREA TO SEE THAT ONLY ONE XFER WAS DONE.
::*****

```

1614	011550	000004			TST26:	SCOPE	
1615	011552	012706	001100			MOV #STACK,SP	:RESTORE STACK
1616	011556	012737	000340	177776		MOV #340,@#PSW	:LOCK OUT INTERRUPTS
1617	011564	004767	004402			JSR PC,CLRREG	:CLEAR UBE REG
1618	011570	005067	016266			CLR BUFF1	:CLEAR BUFFER LOC
1619	011574	012777	177777	170714		MOV #177777,@BEBD	:LOAD UBE DATA REG WITH TEST DATA
1620	011602	012777	030062	170712		MOV #BUFF1,@BEBA	:LOAD UBE ADDRESS REG WITH BUFF ADD.
1621	011610	012777	177777	170702		MOV #177777,@BECC	:SET UBE TO DO 1 CYCLE
1622	011616	012777	003041	170700		MOV #3041,@BECR1	:HAVE UBE DO DATO VIA NPR
1623	011624	000240				NOP	:ALLOW UBE TO SET BUS
1624	011626	004767	004414			JSR PC,CRDY	:CHECK RDY SET
1625	011632	005704				TST R4	:DID RDY SET?
1626	011634	001042				BNE T21L01	:BRANCH TO ERROR IF RDY DID NOT SET
1627	011636	005767	016220			TST BUFF1	:WAS DATO DONE?
1628	011642	001452				BEQ T21L02	:BRANCH TO ERROR IF NPR NOT DONE
1629	011644	005067	016212			CLR BUFF1	:CLEAR BUFF LOC
1629	011650	005067	016210			CLR BUFF1+2	:CLEAR BUFF LOC +2

1676 012102 032777 010000 167062
1677 012110 001002
1678 012112 104401 025457
1679
1680
1688

BIT #SW12,@SWR ;INHIBIT TYPEOUTS?
BNE TST30 ;:BRANCH IF YES
TYPE ,MSG4 ;EXITING TEST

: *TEST 30 TEST WRONG A LINE ERROR BIT DOES NOT SET

: *A DATI NPR IS DONE FROM THE UBE GO ADDRESS
: *THE ERROR BIT IS TESTED NOT TO HAVE SET AND NOT TO HAVE INTERRUPTED.
: *THE ADDRESS BITS 14,15,16,17 ARE NEXT TESTED SEPARATELY
: *AND THE ERROR BIT IS CHECKED NOT TO HAVE SET.

TST30: SCOPE

1689 012116 000004
1689 012120 012706 001100
1690 012124 012737 000340 177776
1691 012132 004767 004034
1692 012136 016777 170372 170356
1693 012144 012777 000003 170354
1694 012152 012777 177777 170340
1695 012160 012777 012230 170344
1696 012166 012777 002041 170330
1697 012174 004767 004046
1698 012200 032777 001000 170320
1699 012206 001404
1700 012210 104070
1701 012212 104071
1702 012214 004767 004212
1703 012220 005037 177776
1704 012224 000240
1705 012226 000410
1706 012230 017767 170272 166756
1707 012236 104070
1708 012240 104077
1709 012242 004767 004164
1710 012246 000447
1711 012250 004767 004022
1712 012254 005077 170242
1713 012260 012777 000001 170240
1714 012266 012777 177777 170224
1715 012274 062777 040000 170220
1716 012302 032777 140000 170212
1717 012310 001011
1718 012312 032777 000003 170206
1719 012320 001422
1720 012322 005277 170200
1721 012326 042777 000004 170172
1722 012334 012777 002041 170162
1723 012342 004767 003700
1724 012346 032777 001000 170152
1725 012354 001744
1726 012356 104070
1727 012360 104071
1728 012362 004767 004044
1729 012366 004767 003632
1730
1741

MOV #STACK,SP ;RESTORE STACK
MOV #340,@#PSW ;LOCK OUT INTERRUPTS
JSR PC,CLRREG ;CLEAR UBE REGS
MOV BEGO,@BEBA ;HAVE UBE ADDRESS ITS GO ADDRESS
MOV #3,@BECCR2 ;HAVE UBE ADDRESS ITS GO ADDRESS
MOV #177777,@BECC ;SET UP TO DO 1 CYCLE
MOV #T24L01,@INTVEC ;SET UP FOR INT.
MOV #2041,@BECCR1 ;HAVE DATI NPR DONE FROM GO ADDRESS
JSR PC,CRDY ;CHECK FOR RDY SET
BIT #1000,@BECCR2 ;WAS ADDRESS ERROR SET?
BEQ T24L02 ;BRANCH IF NO
ERROR +^D56 ;ERROR: TEST OF WRONG A LINES ERROR BIT FAILED
ERROR +^D57 ;BECCR2 BIT 9 FALSELY SET
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
T24L02: CLR @#PSW ;ALLOW ANY INTERRUPTS
NOP ;UBE SHOULD NOT INTERRUPT HERE
BR T24L06 ;GO TEST INDIVIDUAL ADDRESS BITS
T24L01: MOV @BECCR2,\$REGO ;SAVE BECCR2
ERROR +^D56 ;ERROR:TEST OF WRONG A LINES ERROR BIT FAILED
ERROR +^D63 ;FALSELY INTERRUPTED CPU
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
BR T24L03 ;GO RESTORE TRAP
T24L06: JSR PC,DINT ;DISREGARD INTERRUPTS
CLR @BEBA ;CLEAR ADDRESS 0-15
MOV #1,@BECCR2 ;TEST ADDRESS 16
T24L05: MOV #177777,@BECC ;DO 1 CYCLE
ADD #40000,@BEBA ;TEST NEXT ADDRESS
BIT #140000,@BEBA ;HAVE ADDRESS BITS 14,15 BEEN EXERCISED?
BNE T24L04 ;TEST NEXT ADDRESS IF NO
BIT #3,@BECCR2 ;HAVE ADDRESS BITS 16,17 BEEN EXERCISED?
BEQ T24L03 ;GO RESTORE TRAPS IF YES
INC @BECCR2 ;INC ADDRESS BITS 16,17
BIC #4,@BECCR2 ;CLEAR BIT 2 OF BECCR2 IF SET
T24L04: MOV #2041,@BECCR1 ;DO DATI NPR TO ADDRESS
JSR PC,CRDY ;WAIT TILL RDY SET
BIT #1000,@BECCR2 ;WAS WRONG ADDRESS LINES ERROR BIT SET?
BEQ T24L05 ;TEST NEXT ADDRESS IF NO
ERROR +^D56 ;ERROR: TEST OF WRONG A LINES ERROR BIT FAILED
ERROR +^D57 ;BECCR2 BIT 9 FALSELY SET
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
T24L03: JSR PC,RCATCH ;RESTORE TRAP CATCHER

```
;*TEST 31 TEST WRONG GRANTS OR NOT ONE ERROR BIT SET  
;*THE UBE IS SET UP TO DO ONE DATI XFER/REQUEST. ALL  
;*THE POSSIBLE COMBINATIONS OF BR AND NPR LEVELS ARE THEN  
;*EXERCISED. AFTER EACH, THE ERROR BITS AND INTERRUPTS ARE  
;*CHECKED FOR. FINALLY, A DATI NPR IS DONE FROM A BUFFER  
;*AREA WITH THE INTERRUPT ON DONE BIT SET. UPON INTERRUPT, THE  
;*ERROR BITS ARE CHECKED.  
;* TEST WRONG GRANT & NO GRANT OR NOT ONE GRANT ERR BITS DO NOT SET  
:*****
```

```
TST31: SCOPE  
1742 012372 000004  
1742 012374 012706 001100 MOV #STACK,SP ;RESTORE STACK  
1743 012400 004767 003566 JSR PC,CLRREG ;CLEAR UBE REG  
1744 012404 012777 002000 170112 MOV #2000,@BECR1 ;SET UP UBE TO DO 1 DATI XFER/REQ.  
1745 012412 012777 012512 170112 MOV #T25L01,@INTVEC ;SET UP FOR INTERRUPTS  
1746 012420 012737 000340 177776 T25L05: MOV #340,@#PSW ;LOCK OUT INTERRUPTS  
1747 012426 012777 177777 170064 MOV #177777,@BECC ;SET UBE TO DO 1 CYCLE  
1748 012434 012777 030062 170060 MOV #BUFF1,@EBA ;SET UBE TO ADDRESS BUFFER AREA  
1749 012442 062777 000003 170054 ADD #3,@BECR1 ;HAVE UBE DO NEXT LEVEL OF REQUEST  
1750 012450 005037 177776 CLR @#PSW ;ALLOW DATA XFERS VIA BR AND NPR LEVELS  
1751 012454 004767 003566 JSR PC,CRDY ;WAIT TILL RDY SET  
1752 012460 032777 000076 170036 BIT #76,@BECR1 ;HAVE ALL REQUEST LEVELS BEEN EXERCISED  
1753 012466 001425 BEQ T25L02 ;BRANCH IF YES  
1754 012470 032777 000040 170030 BIT #40,@BECR2 ;WAS WRONG GRANT ERROR BIT SET?  
1755 012476 001062 BNE T25L03 ;BRANCH TO ERROR IF SET  
1756 012500 032777 002000 170020 BIT #2000,@BECR2 ;WAS NO GRANT OR NOT ONE GRANT ERROR BIT SET?  
1757 012506 001066 BNE T25L04 ;BRANCH TO ERROR IF YES  
1758 012510 000743 BR T25L05 ;GO TEST NEXT LEVEL  
1759 012512 104100 T25L01: ERROR +^D64 ;ERROR: TEST OF WRONG GRANT OR NOT ONE GRANT FAILED  
1760 012514 017767 170006 166472 MOV @BECR2,$REGO ;SAVE ERROR BITS  
1761 012522 104077 ERROR +^D63 ;FALSELY INTERRUPTED CPU  
1762 012524 017767 167774 166462 MOV @BECR1,$REGO ;SAVE BECR1  
1763 012532 104104 ERROR +^D68 ;WITH BECR1=  
1764 012534 004767 003672 JSR PC,TERRPC ;TYPE PC OF ERROR MSG  
1765 012540 000460 BR T25L08 ;GO RESTORE TRAPS  
1766 012542 012777 012560 167762 T25L02: MOV #T25L06,@INTVEC ;SET UP NEW INT. AREA  
1767 012550 012777 002143 167746 MOV #2143,@BECR1 ;HAVE UBE DO 1 DATI NPR AND INT ON DONE  
1768 012556 000001 WAIT ;WAIT TO BE INTERRUPTED  
1769 012560 032777 000040 167740 T25L06: BIT #40,@BECR2 ;WAS WRONG GRANT ERROR BIT SET?  
1770 012566 001015 BNE T25L07 ;BRANCH TO ERROR IF WAS  
1771 012570 032777 002000 167730 BIT #2000,@BECR2 ;WAS NO GRANT OR NOT ONE GRANT BIT SET?  
1772 012576 001441 BEQ T25L08 ;GO RESTORE TRAPS IF WAS NOT  
1773 012600 104100 ERROR +^D64 ;ERROR: TEST OF WRONG GRANT OR NOT ONE GRANT FAILED  
1774 012602 017767 167716 166404 MOV @BECR1,$REGO ;SAVE BECR1  
1775 012610 104101 ERROR +^D65 ;NO GRANT OR NOT ONE GRANT ERROR BIT FALSELY SET  
1776 012612 104102 ERROR +^D66 ;WITH INT ON DONE = 1  
1777 012614 004767 003612 JSR PC,TERRPC ;TYPE PC OF ERROR MSG  
1778 012620 000430 BR T25L08 ;GO RESTORE TRAPS  
1779 012622 104100 T25L07: ERROR +^D64 ;ERROR: TEST OF WRONG GRANT OR NOT ONE GRANT FAILED  
1780 012624 017767 167674 166362 MOV @BECR1,$REGO ;SAVE BECR1  
1781 012632 104103 ERROR +^D67 ;WRONG GRANT ERROR BIT FALSELY SET  
1782 012634 104102 ERROR +^D66 ;WITH INT ON DONE = 1  
1783 012636 004767 003570 JSR PC,TERRPC ;TYPE PC OF ERROR MSG  
1784 012642 000417 BR T25L08 ;GO RESTORE TRAPS  
1785 012644 104100 T25L03: ERROR +^D64 ;ERROR: TEST OF WRONG GRANT OR NOT ONE GRANT FAILED  
1786 012646 017767 167652 166340 MOV @BECR1,$REGO ;SAVE BECR1  
1787 012654 104103 ERROR +^D67 ;WRONG GRANT ERROR BIT FALSELY SET
```

```

1788 012656 004767 003550          JSR PC,TERRPC          ;TYPE PC OF ERROR MSG
1789 012662 000407                    BR T25L08              ;GO RESTORE TRAPS
1790 012664 104100          T25L04: ERROR +^D64    ;ERROR: TEST OF WRONG GRANT OR NOT ONE GRANT FAILED
1791 012666 017767 167632 166320    MOV @BECR1,$REGO      ;SAVE BECR1
1792 012674 104101          ERROR +^D65           ;NO GRANT OR NOT ONE GRANT ERROR BIT FALSELY SET
1793 012676 004767 003530          JSR PC,TERRPC          ;TYPE PC OF ERROR MSG
1794 012702 004767 003316          T25L08: JSR PC,RCATCH  ;RESTORE TRAP CATCHER
1795
1804

:*****
:*TEST 32          TEST TIME DELAY AND BUSS LATENCY ERROR BITS
:*
:*THE BUS LATENCY ERROR BIT IS SET BY DOING A RELEASE
:*BUS IMMEDIATE FUNCTION AND SETTING THE TIME DELAY BIT.  THE
:*ERROR BIT AND BIT 15 OF BECR1 ARE CHECKED TO SET.  THE
:*ERROR INTERRUPT IS THEN CHECKED FOR AND THE ERROR CONDITION
:*IS TESTED TO CLEAR.
:*****
TST32:  SCOPE
1805 012706 000004          MOV #STACK,SP         ;RESTORE STACK
1806 012710 012706 001100          MOV #340,@#PSW        ;LOCK OUT INTERRUPTS
1807 012714 012737 000340 177776    JSR PC,CLRREG         ;CLEAR UBE REG
1808 012722 004767 003244          JSR PC,CLRREG         ;CLEAR UBE REG
1809 012726 012777 040000 167572    MOV #40000,@BECR2     ;SET TIME DELAY BIT
1810 012734 012777 013044 167570    MOV #T26L01,@INTVEC  ;SET UP FOR INTERRUPTS
1811 012742 012777 006003 167554    MOV #6003,@BECR1     ;DO RELEASE BUS IMMED.
1812 012750 005000          CLR R0                ;INITIALIZE R0
1813 012752 005200          T26L02: INC R0         ;DELAY TO WAIT FOR
1814 012754 022700 000400          CMP #400,R0           ;BUSS LATENCY ERROR BIT
1815 012762 032777 000100 167536    BNE T26L02            ;TO SET
1816 012770 001004          BIT #100,@BECR2      ;WAS BUSS LATENCY ERROR BIT SET?
1817 012772 104106          BNE T26L03            ;BRANCH IF YES
1818 012774 104107          ERROR +^D70          ;ERROR: TEST OF TIME DALAY AND BUSS LATENCY FAILED
1819 012776 004767 003430          ERROR +^D71          ;TO SET BIT 6 OF BECR2
1820 013002 032777 100000 167514    T26L03: JSR PC,TERRPC   ;TYPE PC OF ERROR MSG
1821 013010 001004          BIT #100000,@BECR1  ;WAS ERROR BIT SET?
1822 013012 104106          BNE T26L04            ;BRANCH IF YES
1823 013014 104075          ERROR +^D70 ;        ;ERROR: TEST OF TIME DELAY AND BUSS LATENCY FAILED
1824 013016 004767 003410          ERROR +^D61          ;TO SET BIT 15 OF BECR1
1825 013022 005037 177776          T26L04: JSR PC,TERRPC  ;TYPE PC OF ERROR MSG
1826 013026 000240          CLR @#PSW            ;ALLOW ERROR INTERRUPTS
1827 013030 000240          NOP                  ;** UBE SHOULD INTERRUPT
1828 013032 104106          NOP                  ;** UBE SHOULD INTERRUPT ON LEVEL 7
1829 013034 104072          ERROR +^D70          ;ERROR: TEST OF TIME DELAY AND BUSS LATENCY FAILED
1830 013036 004767 003370          ERROR +^D58          ;TO INTERRUPT CPU
1831 013042 000412          JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1832 013044 005077 167460          BR T26L05            ;GO TO END OF TEST
1833 013050 032777 000100 167450    T26L01: CLR @BERE     ;CLEAR ERROR CONDITION
1834 013056 001404          BIT #100,@BECR2     ;WAS ERROR CLEARED?
1835 013060 104106          BEQ T26L05           ;BRANCH IF YES
1836 013062 104110          ERROR +^D70          ;ERROR: TEST OF TIME DELAY AND BUSS LATENCY FAILED
1837 013064 004767 003342          ERROR +^D72          ;TO CLEAR BIT 6 OF BECR2
1838 013070 004767 003076          T26L05: JSR PC,TERRPC  ;TYPE PC OF ERROR MSG
1839 013074 004767 003176          JSR PC,CLRREG        ;CLEAR ALL UBE REG
1840 013100 012777 177777 167412    JSR PC,DINT          ;DISREGARD ERROR INTERRUPTS
1841 013106 012777 030062 167406    MOV #177777,@BECC    ;HAVE UBE DO DATI
1842 013114 012777 002041 167402    MOV #BUFF1,@BEBA     ;SO BUSS LATENCY REG
1843 013122 004767 003120          MOV #2041,@BECR1    ;HOLD FLOP CLEARED
1843 013122 004767 003120          JSR PC,CRDY         ;WAIT FOR RDY SET
    
```

1844 013126 005077 167376
 1845 013132 004767 003066
 1846
 1847

CLR @BERE ;CLEAR LATENCY ERROR IF SET
 JSR PC,RCATCH ;RESTORE TRAPS

 *TEST 33 TEST MULTIPLE INTERRUPTS SET RDY BIT

013136 000004
 1848 013140 012706 001100
 1849 013144 004767 003022
 1850 013150 004767 003122
 1851 013154 005037 177776
 1852 013160 012777 177776 167332
 1853 013166 012777 040000 167332
 1854 013174 012777 000003 167322
 1855 013202 004767 003040
 1856 013206 005704
 1857 013210 001403
 1858 013212 104124
 1859 013214 004767 003212
 1860 013220 004767 003000
 1861
 1871

TST33: SCOPE
 MOV #STACK,SP ;INITIALIZE STACK
 JSR PC,CLRREG ;CLEAR ALL UBE REG
 JSR PC,DINT ;DISREGARD INTERRUPTS
 CLR @PSW ;ALLOW INTERRUPTS
 MOV #177776,@BECC ;HAVE UBE DO 2 CYCLES
 MOV #40000,@BECC2 ;DO TIME DLY
 MOV #3,@BECC1 ;HAVE UBE INT. VIA BR4
 JSR PC,CRDY ;CHECK FOR RDY SET
 TST R4 ;WAS RDY SET?
 BEQ T31L01 ;BRANCH IF YES
 ERROR +^D84 ;ERROR:TEST OF MULTIPLE INTERRUPTS FAILED TO SET RDY
 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
 T31L01: JSR PC,RCATCH ;RESTORE TRAP CATCHER

 *TEST 34 TEST POWER DOWN SEQUENCE

*THE POWER DOWN TEST IS ONLY DONE IF SW4=1.
 *THE POWER DOWN IS TESTED FOR AND THEN THE POWER UP
 *IS TESTED. AN INTERNAL REG R0 COUNTS FOR A TIME >150
 *MS TO SEE IF THE CPU GETS POWERED UP. THE PROGRAM
 *THEN WAITS FOR A TIME >150MS TO SEE IF THE CPU
 *GETS POWERED DOWN AGAIN.

013224 000004
 1872 013226 012767 000001 166000
 1873 013234 032777 000020 165730
 1874 013242 001516
 1875 013244 012737 000340 177776
 1876 013252 012706 001100
 1877 013256 013746 000024
 1878 013262 013746 000026
 1879 013266 012737 013324 000024
 1880 013274 012737 000340 000026
 1881 013302 012777 000020 167216
 1882 013310 000240
 1883 013312 104111
 1884 013314 104112
 1885 013316 004767 003110
 1886 013322 000450
 1887 013324 022626 013370 000024
 1888 013334 005000
 1889 013336 005001
 1890 013340 005200
 1891 013342 005700
 1892 013344 001375
 1893 013346 005201
 1894 013350 022701 000004
 1895 013354 001371

TST34: SCOPE
 MOV #1,\$TIMES ;DO 1 ITERATION
 BIT #20,@SWR ;SEE IF POWER DOWN TO BE TESTED
 BEQ TST35 ;GO TO NEXT TEST IF SWR4 = 0
 MOV #340,@PSW ;LOCK OUT INTERRUPTS
 MOV #STACK,SP ;INITIALIZE STACK
 MOV @#24,-(SP) ;SAVE POWER FAIL VECTOR ON STACK
 MOV @#26,-(SP) ;SAVE POWER FAIL VECTOR ON STACK
 MOV #T27L01,@#24 ;SET UP FOR POWER FAIL
 MOV #340,@#26 ;SET UP FOR POWER FAIL
 MOV #20,@BECC2 ;HAVE UBE DO POWER FAIL
 NOP ;SHOULD POWER FAIL HERE
 ERROR +^D73 ;ERROR: TEST OF POWER DOWN BIT FAILED
 ERROR +^D74 ;TO POWER DOWN CPU
 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
 BR T27L02 ;RESTORE TRAPS
 T27L01: CMP (SP)+,(SP)+ ;RESTORE STACK
 MOV #T27L03,@#24 ;SET UP FOR POWER UP SEQUENCE
 CLR R0 ;INITIALIZE COUNTER
 CLR R1 ;INITIALIZE COUNTER
 T27L04: INC R0 ;COUNT FOR A TIME
 TST R0 ;GREATER THAN 150 MS
 BNE T27L04
 INC R1
 CMP #4,R1 ;IS TIME > 150 MS?
 BNE T27L04 ;BRANCH IF NO

```

1896 013356 104111          ERROR +*D73          ;ERROR: TEST OF POWER DOWN BIT FAILED
1897 013360 104113          ERROR +*D75          ;TO POWER UP CPU
1898 013362 004767 003044    JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1899 013366 000426          BR T27L02            ;RESTORE TRAPS
1900 013370 012737 013432 000024 T27L03: MOV #T27L05,@#24    ;SET UP TO POWER DOWN AGAIN
1901 013376 005000          CLR R0
1902 013400 005001          CLR R1
1903 013402 005200          T27L06: INC R0        ;COUNT FOR A TIME
1904 013404 005700          TST R0              ;GREATER THAN 150 MS
1905 013406 001375          BNE T27L06
1906 013410 005201          INC R1
1907 013412 022701 000004    CMP #4,R1           ;IS TIME > 150 MS?
1908 013416 001371          BNE T27L06          ;BRANCH IF NO
1909 013420 104111          ERROR +*D73          ;ERROR: TEST OF POWER DOWN BIT FAILED
1910 013422 104114          ERROR +*D76          ;TO REPOWER DOWN CPU
1911 013424 004767 003002    JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1912 013430 000405          BR T27L02            ;GO CHECK POWER DOWN BIT
1913 013432 022626          T27L05: CMP (SP)+,(SP)+ ;RESTORE STACK
1914 013434 012737 013444 000024 MOV #T27L02,@#24    ;SET UP TO POWER UP AGAIN
1915 013442 000001          WAIT                ;WAIT TO POWER UP AGAIN
1916 013444 032777 000020 167054 T27L02: BIT #20,@BECR2 ;WAS POWER DOWN BIT SET?
1917 013452 001004          BNE T27L07          ;BRANCH IF YES
1918 013454 104111          ERROR +*D73          ;ERROR: TEST OF POWER DOWN BIT FAILED
1919 013456 104115          ERROR +*D77          ;TO SET BIT 4 OF BECR2
1920 013460 004767 002746    JSR PC,TERRPC        ;TYPE PC OF ERROR MSG
1921 013464 012637 000026    T27L07: MOV (SP)+,@#26 ;RESTORE POWER FAIL VECTOR
1922 013470 012637 000024    MOV (SP)+,@#24
1923 013474 005077 167026    CLR @BECR2          ;CLEAR POWER DOWN BIT
1924
1930
;*****
;*TEST 35          TEST DCLO CLRS BECC,BEBA,BECR2,0,6,7,15
;*
;*THIS TEST IS ONLY DONE IF SW4=1.
;* TEST DCLO CLRS BECC,BEBA,BECR2,& BITS,0-6,7-15,OF BECR1
;*****
TST35: SCOPE
MOV #1,$TIMES          ;;DO 1 ITERATION
BIT #20,@SWR           ;SEE IF POWER DOWN TO BE TESTED
BNE T28L10            ;BRANCH IF SW4=1
JMP TSTB              ;GO TO NEXT TEST
T28L10: MOV #177777,@BECC ;HAVE UBE DO 1 CYCLE
MOV #3,@BECR2         ;SET ADDRESS BITS 16, 17
MOV #160000,@BEBA    ;LOAD UBE WITH ADDRESS THAT RETURNS NO SSYN
JSR PC,DINT          ;DISREGARD INTERRUPTS
MOV #2041,@BECR1     ;HAVE UBE DO DATI SO CCOVF=1 AND NSSYN ERROR = 1
CLR @PSW             ;ALLOW INTERRUPTS
WAIT                ;WAIT TILL ERROR INTERRUPT
MOV @#24,-(SP)       ;STORE POWER VECTOR ON STACK
MOV @#26,-(SP)       ;STORE POWER VECTOR ON STACK
MOV #177777,@BEBA    ;LOAD ADDRESS REG WITH ALL '1'
MOV #177777,@BECC    ;LOAD CYCLE COUNT REG WITH ALL '1'
MOV #77776,@BECR1   ;LOAD BECR1 WITH ONES
MOV #T28L01,@#24    ;SET UP FOR POWER DOWN
MOV #40037,@BECR2   ;LOAD BECR2 WITH ONES AND DO POWER DOWN
WAIT                ;CPU SHOULD POWER DOWN
T28L01: CMP (SP)+,(SP)+ ;RESTORE STACK
MOV #T28L05,@#24    ;SETUP FOR POWER UP
    
```

1951	013646	000001			WAIT		;CPU SHOULD POWER UP
1952	013650	042777	000020	166650	T28L05: BIC #20,@BECCR2		;CLEAR POWER DOWN BIT
1953	013656	016767	166636	165330	MOV BECC,\$REG0		;SAVE BECC ADDRESS
1954	013664	005067	165330		CLR \$REG2		;SAVE CORRECT DATA
1955	013670	005777	166624		TST @BECC		;(BECC)=0?
1956	013674	001026			BNE T28L02		;BRANCH IF NO
1957	013676	016767	166620	165310	MOV BEBA,\$REG0		;SAVE BEBA ADDRESS
1958	013704	005777	166612		TST @BEBA		;(BEBA)=0?
1959	013710	001020			BNE T28L02		;BRANCH IF NO
1960	013712	016767	166610	165274	MOV BECCR2,\$REG0		;SAVE BECCR2 ADDRESS
1961	013720	005777	166602		TST @BECCR2		;WAS BECCR2 CLEARED?
1962	013724	001012			BNE T28L02		;BRANCH IF NO
1963	013726	016767	166572	165260	MOV BECCR1,\$REG0		;SAVE BECCR1 ADDRESS
1964	013734	012767	000200	165256	MOV #200,\$REG2		;SAVE CORRECT DATA (BECCR1)
1965	013742	022777	000200	166554	CMP #200,@BECCR1		;WAS BECCR1 CLEARED?
1966	013750	001407			BEQ T28L03		;BRANCH IF YES
1967	013752	017767	165236	165236	T28L02: MOV @\$REG0,\$REG1		;SAVE BAD DATA
1968	013760	104116			ERROR +^D78		;ERROR: DCLO FAILED TO CLEAR REG
1969	013762	004767	002444		JSR PC,TERRPC		;TYPE PC OF ERROR MSG
1970	013766	000454			BR T28L04		;GO RESTORE VECTORS
1971	013770	012737	000340	177776	T28L03: MOV #340,@#PSW		;LOCK OUT INTERRUPTS
1972	013776	012777	040000	166522	MOV #40000,@BECCR2		;SET TIME DLY BIT
1973	014004	012777	006003	166512	MOV #6003,@BECCR1		;DO RELEASE BUSS IMMED. TO SET LATENCY ERROR BIT
1974	014012	032777	000100	166506	T28L06: BIT #100,@BECCR2		;TEST LATENCY ERROR BIT
1975	014020	001774			BEQ T28L06		;WAIT TILL IT SETS
1976	014022	005037	177776		CLR @#PSW		;ALLOW LATENCY ERROR INTERRUPT
1977	014026	000240			NOP		;ALLOW INTERRUPT TO BE IGNORED
1978	014030	012737	014046	000024	MOV #T28L08,@#24		;SET UP FOR POWER DOWN
1979	014036	052777	000020	166462	BIS #20,@BECCR2		;SET POWER DOWN BIT
1980	014044	000001			WAIT		;WAIT FOR POWER DOWN
1981	014046	022626			T28L08: CMP (SP)+,(SP)+		;RESTORE STACK
1982	014050	012737	014060	000024	MOV #T28L09,@#24		;SETUP FOR POWER UP
1983	014056	000001			WAIT		;CPU SHOULD POWER UP
1984	014060	005077	166442		T28L09: CLR @BECCR2		;CLEAR POWER DOWN BIT
1985	014064	005777	166436		TST @BECCR2		;WAS BUSS LATENCY ERROR BIT CLEARED?
1986	014070	001413			BEQ T28L04		;BRANCH IF YES
1987	014072	016767	166430	165114	MOV BECCR2,\$REG0		;SAVE REG ADDRESS
1988	014100	017767	166422	165110	MOV @BECCR2,\$REG1		;SAVE REG DATA
1989	014106	005067	165106		CLR \$REG2		;SAVE CORRECT DATA
1990	014112	104116			ERROR +^D78		;ERROR: DCLO FAILED TO CLEAR REG
1991	014114	004767	002312		JSR PC,TERRPC		;TYPE PC OF ERROR MSG
1992	014120	004767	002100		T28L04: JSR PC,RCATCH		;RESTORE TRAP CATCHER
1993	014124	012637	000026		MOV (SP)+,@#26		;RESTORE POWER VECTOR
1994	014130	012637	000024		MOV (SP)+,@#24		;RESTORE POWER VECTOR
1995							
1996	014134				TSTB:		
1997							
2004							

```

*****
;*TEST 36 TEST SIMULTANEOUS GO ADDRESS
;*
;*THE UBE IS SETUP TO INTERRUPT ON LEVEL 7 AND
;*THEN TOLD TO GO VIA THE SIMULTANEOUS GO. NO
;*INTERRUPT INDICATES AN ERROR.
*****

```

2005	014134	000004			TST36: SCOPE		
2006	014136	012706	001100		MOV #STACK,SP		;RESTORE STACK
	014142	012737	000340	177776	MOV #340,@#PSW		;LOCK OUT INTERRUPTS

2007	014150	004767	002016		JSR PC,CLRREG	;CLEAR ALL UBE REGS.
2008	014154	012777	014212	166350	MOV #T09L01,@INTVEC	;SETUP TO RECEIVE INTERRUPT
2009	014162	012777	000020	166334	MOV #20,@BECR1	;SETUP TO DO BR=7
2010	014170	005277	166340		INC @BEGO	;START SIMULTANEOUS GO
2011	014174	012737	000300	177776	MOV #300,@PSW	;ALLOW INTERRUPTS
2012	014202	000240			NOP	;UBE SHOULD INTERRUPT HERE
2013	014204	104025			ERRCR +^D21	;ERROR: SIMULTANEOUS GO FAILED
2014	014206	004767	002220		JSR PC,TERRPC	;TYPE PC OF ERROR MSG
2015	014212	004767	002006		T09L01: JSR PC,RCATCH	;RESTORE TRAP CATCHER
2016						
2017						

2029

```
*****  
: *TEST 37          DYNAMIC TEST OF UBE  
: *  
: *THIS TEST EXERCISES THE MOST HARDWARE IN THE  
: *UBE AT ONE TIME.  THE EXERCISOR IS SET UP TO DO EIGHT  
: *DATOB ON DATIP XFERS VIA NPR AND INTERRUPT ON DONE.  
: *AFTER INTERRUPTING, A BUFFER AREA IS EXAMINED TO SEE IF  
: *THE OPERATIONS WERE DONE PROPERLY.  THE ABOVE IS THEN  
: *REPEATED 100 TIMES.  
: *****
```

014216 000004
014220 012767 000001 165006

```
TST37:  SCOPE  
        MOV   #1,$TIMES      ;;DO 1 ITERATION
```

```

2031 014226 004767 001740 JSR PC,CLRREG ;CLEAR UBE REG
2032 014232 005002 CLR R2 ;INITIALIZE COUNT
2033 014234 005037 177776 CLR @#PSW ;ALLOW INTERRUPTS
2034 014240 012700 030062 T29L04: MOV #BUFF1,R0 ;GET BUFFER ADDRESS
2035 014244 012720 052525 T29L01: MOV #52525,(R0)+ ;LOAD BUFFER
2036 014250 020027 030104 CMP R0,#BUFF1+22 ;ENTIRE BUFFER LOADED?
2037 014254 001373 BNE T29L01 ;BRANCH IF NO
2038 014256 012777 014336 166246 MOV #T29L02,@INTVEC ;SET UP FOR INTERRUPTS
2039 014264 012777 030062 166230 MOV #BUFF1,@BEBA ;LOAD BUFF ADDRESS IN UBE
2040 014272 012777 177760 166220 MOV #177760,@BECC ;SET UBE TO DO 16 CYCLES
2041 014300 012777 042561 166216 MOV #42561,@BECCR1 ;DO DATOB ON DATIP, AND INT. VIA BR7 WHEN DONE
2042 014306 005000 CLR R0 ;INITIALIZE COUNTER
2043 014310 016767 013566 013564 T29L06: MOV BUFF1+20,BUFF1+20 ;DO BACKGROUND NOISE PATTERN
2044 014316 005200 INC R0 ;WAIT FOR COUNTER R0
2045 014320 005700 TST R0 ;TO OVERFLOW. IF DOES
2046 014322 001372 BNE T29L06 ;UBE FAILED TO INTERRUPT
2047 014324 104120 ERROR +^D80 ;ERROR: DYNAMIC TEST OF UBE FAILED
2048 014326 104072 ERROR +^D58 ;TO INTERRUPT CPU
2049 014330 004767 002076 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2050 014334 000432 BR T29L07 ;GO RESTORE TRAPS CATCHER
2051 014336 022626 T29L02: CMP (SP)+,(SP)+ ;RESTORE STACK
2052 014340 012700 030062 MOV #BUFF1,R0 ;GET BUFFER ADDRESS
2053 014344 022710 125652 T29L05: CMP #125652,(R0) ;WAS DATA SHIFTED PROPERLY?
2054 014350 001011 BNE T29L03 ;BRANCH TO ERROR IF NO
2055 014352 005720 TST (R0)+ ;INC R0 BY 2
2056 014354 022700 030102 CMP #BUFF1+20,R0 ;AT END OF BUFFER?
2057 014360 001371 BNE T29L05 ;BRANCH IF NO
2058 014362 005202 INC R2 ;UPDATE COUNT
2059 014364 020227 000100 CMP R2,#100 ;WAS UBE EXERCISED 100 TIMES?
2060 014370 001323 BNE T29L04 ;BRANCH IF NO
2061 014372 000413 BR T29L07 ;RESTORE TRAPS
2062 014374 010067 164614 T29L03: MOV R0,$REG0 ;SAVE ADDRESS
2063 014400 011067 164612 MOV (R0),$REG1 ;SAVE BAD DATA
2064 014404 012767 125652 164606 MOV #125652,$REG2 ;SAVE CORRECT DATA
2065 014412 104120 ERROR +^D80 ;ERROR: DYNAMIC TEST OF UBE FAILED
2066 014414 104121 ERROR +^D81 ;TO LOAD PROPER DATA
2067 014416 004767 002010 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2068 014422 004767 001576 T29L07: JSR PC,RCATCH ;RESTORE TRAP CATCHER
2069
2070 ;////////////////////////////////////
2071 ;RETURN ROUTINE TO TEST NEXT UBE BEFORE DO LAST TEST
2072 ;////////////////////////////////////
2073 014426 000004 SCOPE ;SCOPE FOR PREVIOUS TEST
2074 014430 004767 001536 NUBE: JSR PC,CLRREG ;CLEAR UBE SO NO INT.
2075 014434 000167 166736 NUBE1: JMP ACALC ;GO SEE IF MORE UBE
2076
2077 014440 012767 014462 164472 LAST: MOV #LAST1,$LPADR ;SETUP LOOP ADDRESS FOR LAST TEST
2078 014446 012767 014462 164466 MOV #LAST1,$LPERR ;SETUP LOOP ON ERROR ADDRESS FOR LAST TEST
2079 014454 105367 164454 DECB $TSTNM ;ADJUST TEST NUMBER
2080
2081
2082
2101
;*****
;*TEST 40 TEST PASSING OF GRANTS
;*
;*THIS TEST IS ONLY RUN IF THERE ARE MORE THAN ONE
;*UBE.IT IS COMPOSED OF TWO PARTS.THE FIRST PART CHECKS THAT

```

```

;*A HIGHER ELECTRICAL PRIORITY UBE WITH ALL BR LEVELS =1
;*AND GO BIT =0 WILL PASS A GRANT TO THE NEXT LOWER ONE.
;*THEN THIS SAME UBE IS CHECKED TO ALSO PASS A GRANT WHEN ALL BR=0
;*AND THE GO BIT IS ENABLED.
;* THE SECOND PART VERIFIES THAT A UBE WITH A HIGHER ELECTRICAL PRIORITY
;*BUT DOING A LOWER BR THAN A UBE OF LOWER ELECTRICAL
;*PRIORITY, WILL PASS THE GRANT TO THE UBE OF LOWER ELECTRICAL
;*PRIORITY.
;*
;*NOTE: THE UBE WITH THE LOWEST ELECTRICAL PRIORITY
;* ON THE BUS MUST BE SWAPPED WITH A HIGHER
;* ONE AND THEN THE ENTIRE PROGRAM RERUN INORDER
;* THAT ITS PASSING GRANT LOGIC IS TESTED.

```

```

*****
TST40: SCOPE
LAST1: TST BE2BD ;IS THERE MORE THAN ONE EXERCISOR?
        BNE T30L01 ;BRANCH IF YES
        BIT #SW12,@SWR ;INHIBIT TYPEOUTS?
        BNE 1$ ;BRANCH IF YES
        TYPE ,MSG11 ;PASSING OF GRANTS NOT TESTED WITH 1 EXERCISOR
        MOV #1,$TIMES ;DO 1 ITERATION IF THIS TEST NOT DONE
1$: JMP $EOP ;GO TO END OF TEST

;DETERMINE ELECTRICAL PRIORITY OF EXERCISORS
T30L01: MOV #STACK,SP ;INITIALIZE STACK
        MOV #T30L02,@BE1VEC ;SET UP UBE1 INTERRUPT HANDLER
        MOV BE1VEC,R0
        MOV #340,2(R0)
        MOV #T30L03,@BE2VEC ;SET UP UBE2 INTERRUPT HANDLER
        MOV BE2VEC,R0
        MOV #340,2(R0)
        TST BE3VEC ;ARE THERE 3 UBE?
        BEQ T30L21 ;BRANCH IF NO
        MOV #T30L04,@BE3VEC ;SET UP UBE3 INTERRUPT HANDLER
        MOV BE3VEC,R0
        MOV #340,2(R0)
        TST BE4VEC ;ARE THERE 4 UBE?
        BEQ T30L21 ;BRANCH IF NO
        MOV #T30L05,@BE4VEC ;SET UP UBE4 INTERRUPT HANDLER
        MOV BE4VEC,R0
        MOV #340,2(R0)
T30L21: MOV #BUFF1,R0 ;GET BUFFER ADDRESS
        CLR R1 ;INITIALIZE COUNT OF INTERRUPTS
        MOV #340,@#PSW ;SET PSW PRIORITY=7
        MOV #20,@BE1CR1 ;LOAD FIRST UBE TO DO INT. VIA BR7
        MOV #20,@BE2CR1 ;LOAD SECOND UBE TO DO INT. VIA BR7
        TST BE3CR1 ;TEST IF 3 EXERCISORS
        BEQ T30L07 ;BRANCH IF NO
        MOV #20,@BE3CR1 ;LOAD THIRD UBE TO DO INT. VIA BR7
        TST BE4CR1 ;TEST IF 4 EXERCISORS
        BEQ T30L07 ;BRANCH IF NO
        MOV #20,@BE4CR1 ;LOAD FOURTH UBE TO DO INT. VIA BR7
T30L07: INC @BEGC ;LET ALL EXERCISORS INTERRUPT
        CLR @#PSW ;ALLOW INTERRUPTS
        WAIT ;WAIT FOR 1ST INTERRUPT
T30L02: MOV #BE1BD,(R0)+ ;LOAD BUFFER WITH POINTER TO ADDRESS OF UBE

```

```

014460 000004
2102 014462 005767 166066
2103 014466 001013
2104 014470 032777 010000 164474
2105 014476 001005
2106 014500 104401 027356
2107 014504 012767 000001 164522
2108 014512 000167 001332
2109
2110
2111
2112 014516 012706 001100
2113 014522 012777 014730 166022
2114 014530 016700 166016
2115 014534 012760 000340 000002
2116 014542 012777 014744 166020
2117 014550 016700 166014
2118 014554 012760 000340 000002
2119 014562 005767 166020
2120 014566 001423
2121 014570 012777 014760 166010
2122 014576 016700 166004
2123 014602 012760 000340 000002
2124 014610 005767 166010
2125 014614 001410
2126 014616 012777 014774 166000
2127 014624 016700 165774
2128 014630 012760 000340 000002
2129 014636 012700 030062
2130 014642 005001
2131 014644 012737 000340 177776
2132 014652 012777 000020 165664
2133 014660 012777 000020 165674
2134 014666 005767 165706
2135 014672 001411
2136 014674 012777 000020 165676
2137 014702 005767 165710
2138 014706 001403
2139 014710 012777 000020 165700
2140 014716 005277 165612
2141 014722 005037 177776
2142 014726 000001
2143 014730 012720 002536

```

```

2144 014734 012777 006002 165602      MOV #6002,@BE1CR1      ;SETUP FIRST UBE TO DO A FUN 3
2145 014742 000421                    BR T30L06              ;GO SEE IF ALL UBE INTERRUPTED
2146 014744 012720 002554            T30L03: MOV #BE2BD,(R0)+ ;LOAD BUFFER WITH POINTER TO UBE ADDRESSES
2147 014750 012777 006002 165604      MOV #6002,@BE2CR1      ;SETUP SECOND UBE TO DO A FUN3
2148 014756 000413                    BR T30L06              ;GO SEE IF ALL UBE INTERRUPTED
2149 014760 012720 002572            T30L04: MOV #BE3BD,(R0)+ ;LOAD BUFFER WITH POINTER TO UBE ADDRESS
2150 014764 012777 006002 165606      MOV #6002,@BE3CR1      ;SETUP THIRD UBE TO DO A FUN3
2151 014772 000405                    BR T30L06              ;GO SEE IF ALL UBE INTERRUPTED
2152 014774 012720 002610            T30L05: MOV #BE4BD,(R0)+ ;LOAD BUFFER WITH POINTER TO UBE ADDRESS
2153 015000 012777 006002 165610      MOV #6002,@BE4CR1      ;SETUP FOURTH UBE TO DO A FUN3
2154 015006 022626                    T30L06: CMP (SP)+,(SP)+ ;RESTORE STACK
2155 015010 005201                    INC R1                 ;COUNT INTERRUPTS
2156 015012 020167 165610            CMP R1,UCNT           ;HAVE ALL EXERCISORS INTERRUPTED?
2157 015016 001403                    BEQ T30L22            ;BRANCH IF YES
2158 015020 005037 177776            CLR @#PSW             ;ALLOW NEXT UBE TO INTERRUPT
2159 015024 000001                    WAIT                  ;WAIT FOR INTERRUPT
2160 015026 024040                    T30L22: CMP -(R0),-(R0) ;DECREMENT R0 BY 4
2161 015030 011067 013036            MOV (R0),BUFF1+10    ;PUT NEXT TO LOWEST PRIORITY POINTER IN BUFF1+10
2162
2163 ;BUFFER NOW CONTAINS VECTORS IN ORDER OF ELECTRICAL PRIORITY
2164
2165 ;PART 1
2166
2167 015034 016700 165566            MOV UCNT,R0           ;GET COUNT OF UBE
2168 015040 005300                    DEC R0                 ;ADJUST COUNT
2169 015042 005001                    CLR R1                 ;CLEAR INDEX REG
2170 015044 016102 030062            T30L28: MOV BUFF1(R1),R2 ;GET PTER TO ADDRESS OF HIGHER PRIORITY UBE
2171 015050 012772 000036 000006      MOV #36,@6(R2)        ;SET ALL BR =1 IN THIS UBE
2172 015056 005721                    TST (R1)+             ;UPDATE INDEX
2173 015060 016103 030062            MOV BUFF1(R1),R3     ;GET PTER TO ADDRESS OF NEXT LOWER PRIORITY UBE
2174 015064 012773 015202 000014      MOV #T30L25,@14(R3)  ;SET UP FOR INT.
2175 015072 012773 000002 000006      T30L30: MOV #2,@6(R3) ;SETUP LOWER PRIORITY UBE FOR BR4
2176 015100 005273 000006            T30L26: INC @6(R3)    ;HAVE UBE INT.
2177 015104 005037 177776            CLR @#PSW             ;ALLOW INT.
2178 015110 000240                    NOP                   ;SHOULD INT. HERE
2179 015112 012737 000340 177776      MOV #340,@#PSW        ;LOCK OUT INT.
2180 015120 104122                    T30L29: ERROR +^DB2  ;ERROR:TEST OF PASSING GRANTS FAILED
2181 015122 032777 020000 164042      BIT #SW13,@SWR        ;INHIBIT ERROR TYPEOUTS?
2182 015130 001022                    BNE 1$                ;BRANCH IF YES
2183 015132 016367 000014 164054      MOV 14(R3),$REG0      ;SAVE INT. VECTOR
2184 015140 104401 027230                    TYPE ,MSG7            ;UBE WITH INT. VECTOR:
2185 015144 016746 164044            MOV $REG0,-(SP)      ;:SAVE $REG0 FOR TYPEOUT
2186 015150 104402                    TYPOC                 ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
2187 015152 017367 000006 164036      MOV @6(R3),$REG1     ;SAVE (BECR1)
2188 015160 104401 026207                    TYPE ,DH65           ;WITH BECR1=
2189 015164 016746 164026            MOV $REG1,-(SP)     ;:SAVE $REG1 FOR TYPEOUT
2190 015170 104402                    TYPOC                 ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
2189 015172 104401 027323                    TYPE ,MSG10          ;SHOULD HAVE INT.
2190 015176 000167 000474            1$: JMP T30L12        ;GO TO END OF TEST
2191 015202 006373 000006            T30L25: ASL @6(R3)    ;DO NEXT BR LEVEL
2192 015206 042773 000400 000006      BIC #400,@6(R3)      ;CLEAR SHIFTED RDY BIT
2193 015214 032773 000040 000006      BIT #40,@6(R3)       ;ALL BR TESTED?
2194 015222 001726                    BEQ T30L26            ;BRANCH IF NO
2195
2196 015224 012773 015250 000014      MOV #T30L27,@14(R3)  ;SETUP FOR INT.
2197 015232 012772 015120 000014      MOV #T30L29,@14(R2)  ;SETUP FOR ERROR INT.
2198 015240 012772 000001 000006      MOV #1,@6(R2)        ;HAVE HIGHER UBE TRY TO INT.

```

```

2199 015246 000711          BR T30L30          ;LET LOWER UBE INT.
2200
2201 015250 006373 000006    T30L27: ASL @6(R3)          ;DO NEXT LEVEL BR
2202 015254 042773 000400 000006    BIC #400,@6(R3)          ;CLEAR SHIFTED RDY
2203 015262 032773 000040 000006    BIT #40,@6(R3)          ;ALL BR TESTED?
2204 015270 001703          BEQ T30L26          ;BRANCH IF NO
2205 015272 012772 006003 000006    MOV #6003,@6(R2)        ;HAVE HIGHER UBE DO FUN3
2206 015300 005037 177776          CLR @#PSW              ;ALLOW REQUESTS
2207 015304 105772 000006    1$: TSTB @6(R2)          ;IS UBE DONE?
2208 015310 100375          BPL 1$                ;BRANCH IF NO
2209 015312 012737 000340 177776    MOV #340,@#PSW          ;SET LEVEL =7
2210 015320 005300          DEC R0                 ;ADJUST UBE COUNT
2211 015322 005700          TST R0                 ;ALL UBE TESTED?
2212 015324 001247          BNE T30L28            ;BRANCH IF NO
2213
2214          ;PART 2
2215
2216 015326 012700 000510          MOV #510,R0             ;GET FIRST POSSIBLE VECTOR AREA
2217 015332 012720 015506    T30L09: MOV #T30L08,(R0)+   ;SET UP VECTOR AREA TO HANDLE DOUBLE INTERRUPTS
2218 015336 012720 000340          MOV #340,(R0)+         ;SET PRIORITY = 7
2219 015342 022700 001000          CMP #1000,R0           ;AT END OF AREA?
2220 015346 001371          BNE T30L09             ;BRANCH IF NO
2221 015350 016700 012506          MOV BUFF1,R0           ;GET HIGHEST PRIORITY UBE ADDRESS POINTER
2222 015354 016701 012504          MOV BUFF1+2,R1        ;GET NEXT PRIORITY UBE ADDRESS POINTER
2223 015360 012770 000002 000006    T30L14: MOV #2,@6(R0)       ;HAVE HIGHER PRIORITY UBE DO BR4
2224 015366 012771 000004 000006    MOV #4,@6(R1)          ;HAVE NEXT LOWER ELEC. PRIORITY UBE DO BR5
2225 015374 012770 015506 000014    MOV #T30L08,@14(R0)    ;SET UP HIGHER PRIORITY UBE VECTOR FOR DOUBLE INT.
2226 015402 012771 015422 000014    MOV #T30L10,@14(R1)    ;SET UP FOR INTERRUPT FROM NEXT LOWER ELEC. PRIORITY UBE
2227 015410 005277 165120    T30L11: INC @BEGO         ;START INTERRUPT
2228 015414 005037 177776          CLR @#PSW              ;ALLOW INTERRUPTS
2229 015420 000001          WAIT
2230 015422 022626    T30L10: CMP (SP)+,(SP)+     ;RESTORE STACK
2231 015424 006371 000006          ASL @6(R1)             ;HAVE NEXT PRIORITY UBE INT. ONE LEVEL HIGHER
2232 015430 042771 000400 000006    BIC #400,@6(R1)        ;CLEAR SHIFTED RDY
2233 015436 032771 000040 000006    BIT #40,@6(R1)        ;TESTED ALL BR LEVELS?
2234 015444 001761          BEQ T30L11            ;BRANCH IF NO
2235 015446 020067 012420          CMP R0,BUFF1+10       ;TESTED ALL UBE POSSIBLE?
2236 015452 001511          BEQ T30L12            ;BRANCH IF YES TO CLEAR BECR1 AND RESTORE TRAPS
2237 015454 020067 012402          CMP R0,BUFF1          ;JUST TESTED FIRST UBE?
2238 015460 001005          BNE T30L13            ;BRANCH IF NO
2239 015462 016700 012376          MOV BUFF1+2,R0        ;TEST SECOND HIGHEST PRIORITY UBE
2240 015466 016701 012374          MOV BUFF1+4,R1        ;GET THIRD HIGHEST PRIORITY UBE
2241 015472 000732          BR T30L14              ;GO TEST SECOND HIGHEST PRIORITY UBE
2242 015474 016700 012366    T30L13: MOV BUFF1+4,R0       ;TEST THIRD HIGHEST PRIORITY UBE
2243 015500 016701 012364          MOV BUFF1+6,R1        ;GET FOURTH HIGHEST PRIORITY UBE
2244 015504 000725          BR T30L14              ;GO TEST THIRD HIGH PRIORITY UBE
2245 015506 022626    T30L08: CMP (SP)+,(SP)+     ;RESTORE STACK
2246 015510 016067 000014 163476    MOV 14(R0),$REG0       ;SAVE INTERRUPT VECTOR OF BAD UBE
2247 015516 012767 000004 163472    MOV #4,$REG1           ;SAVE BAD BR LEVEL
2248 015524 016167 000014 163466    MOV 14(R1),$REG2       ;SAVE NEXT HIGHER PRIORITY UBE VECTOR
2249 015532 032771 000004 000006    BIT #4,@6(R1)         ;WAS BR=5?
2250 015540 001404          BEQ T30L15            ;BRANCH IF NO
2251 015542 012767 000005 163452    MOV #5,$REG3          ;BR=5
2252 015550 000413          BR T30L17             ;GO INDICATE ERROR
2253 015552 032771 000010 000006    T30L15: BIT #10,@6(R1)   ;WAS BR=6?
2254 015560 001404          BEQ T30L16            ;BRANCH IF NO
2255 015562 012767 000006 163432    MOV #6,$REG3          ;INDICATE BR=6

```

2256	015570	000403				BR T30L17		:GO INDICATE ERROR
2257	015572	012767	000007	163422	T30L16:	MOV #7,\$REG3		:INDICATE BR=7
2258	015600	104122			T30L17:	ERROR +^D82		:ERROR: TEST OF PASSING GRANTS FAILED
2259	015602	032777	0200J0	163362		BIT #SW13,@SWR		:INHIBIT ERROR TYPEOUTS?
2260	015610	001032				BNE T30L12		:BRANCH IF YES
2261	015612	104401	027230			TYPE ,MSG7		:TYPE FAILING UBE VECTOR
2262	015616	016746	163372			MOV \$REG0,-(SP)		::SAVE \$REG0 FOR TYPEOUT
	015622	104402				TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
2263	015624	104401	027252			TYPE ,MSG8		:TYPE FAILING UBE BR LEVEL
2264	015630	016746	163362			MOV \$REG1,-(SP)		::SAVE \$REG1 FOR TYPEOUT
	015634	104402				TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
2265	015636	104401	027267			TYPE ,MSG9		
2266	015642	104401	027230			TYPE ,MSG7		:TYPE UBE USED TO TEST FAILING ONE
2267	015646	016746	163346			MOV \$REG2,-(SP)		::SAVE \$REG2 FOR TYPEOUT
	015652	104402				TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
2268	015654	104401	027252			TYPE ,MSG8		:TYPE BR LEVEL TESTING
2269	015660	016746	163336			MOV \$REG3,-(SP)		::SAVE \$REG3 FOR TYPEOUT
	015664	104402				TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
2270	015666	104401	027323			TYPE ,MSG10		
2271	015672	004767	000534			JSR PC,TERRPC		:TYPE PC OF ERROR MSG
2272	015676	012777	006003	164640	T30L12:	MOV #6003,@BE1CR1		:SETUP UBE TO DO A FUN3
2273	015704	012777	006003	164650		MOV #6003,@BE2CR1		:SETUP UBE TO DO A FUN3
2274	015712	005767	164662			TST BE3CR1		:ARE THERE 3 UBE?
2275	015716	001411				BEQ 1\$:BRANCH IF NO
2276	015720	012777	006003	164652		MOV #6003,@BE3CR1		:SETUP UBE TO DO A FUN3
2277	015726	005767	164664			TST BE4CR1		:ARE THERE 4 UBE?
2278	015732	001403				BEQ 1\$:BRANCH IF NO
2279	015734	012777	006003	164654		MOV #6003,@BE4CR1		:SETUP UBE TO DO A FUN3
2280	015742	005037	177776		1\$:	CLR @#PSW		:ALLOW ALL UBE TO DO FUN3
2281	015746	105777	164572		2\$:	TSTB @BE1CR1		:FIRST UBE DONE?
2282	015752	100375				BPL 2\$:BRANCH IF NO
2283	015754	105777	164602		3\$:	TSTB @BE2CR1		:SECOND UBE DONE?
2284	015760	100375				BPL 3\$:BRANCH IF NO
2285	015762	005767	164612			TST BE3CR1		:ARE THERE THREE UBE?
2286	015766	001411				BEQ 6\$:BRANCH IF NO
2287	015770	105777	164604		4\$:	TSTB @BE3CR1		:THIRD UBE DONE?
2288	015774	100375				BPL 4\$:BRANCH IF NO
2289	015776	005767	164614			TST BE4CR1		:ARE THERE 4 UBE?
2290	016002	001403				BEQ 6\$:BRANCH IF NO
2291	016004	105777	164606		5\$:	TSTB @BE4CR1		:FOURTH UBE DONE?
2292	016010	100375				BPL 5\$:BRANCH IF NO
2293								
2294								
2295								
2296	016012	012700	000510		6\$:	MOV #510,R0		:GET FIRST VECTOR ADDRESS
2297	016016	012701	000512			MOV #512,R1		
2298	016022	010120			T30L20:	MOV R1,(R0)+		:PUT ADDRESS OF NEXT LOC IN THIS ONE
2299	016024	005020				CLR (R0)+		:PUT HALT IN NEXT LOCATION
2300	016026	022121				CMP (R1)+,(R1)+		:INC R1 BY 4
2301	016030	020027	001000			CMP R0,#1000		:AT END OF VECTOR AREA?
2302	016034	001372				BNE T30L20		:BRANCH IF NO
2303	016036	005767	163070			TST \$PASS		:FIRST PASS OF PROGRAM?
2304	016042	001002				BNE \$EOP		:BRANCH IF NO
2305	016044	104401	020425			TYPE ,MSG2		:ALL EXERCISORS TESTED
2306								:NOTE:TO TEST PASSING OF GRANTS FOR
2307								:THE LAST UBE,IT SHOULD BE
2308								:SWAPPED WITH A UBE OF HIGHER

2309
2310
2311
2312

;ELECTRICAL PRIORITY

.SBTTL END OF PASS ROUTINE

;*INCREMENT THE PASS NUMBER (\$PASS)
;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
;*IF THERES A MONITOR GO TO IT
;*IF THERE ISN'T JUMP TO START1

016050
016050 000004
016052 005067 163056
016056 005067 163152
016062 005267 163044
016066 042767 100000 163036
016074 005327
016076 000001
016100 003022
016102 012737
016104 000001
016106 016076
016110 104401 016155
016114 016746 163012
016120 104405
016122 104401 016152
016126 013700 000042
016132 001405
016134 000005
016136 004710
016140 000240
016142 000240
016144 000240
016146
016146 000137
016150 003100
016152 377 377 000
016155 015 012 105
016160 116 104 040
016163 120 101 123
016166 123 040 043
016171 000

\$EOP: SCOPE
CLR \$STNM ;;ZERO THE TEST NUMBER
CLR \$TIMES ;;ZERO THE NUMBER OF ITERATIONS
INC \$PASS ;;INCREMENT THE PASS NUMBER
BIC #100000,\$PASS ;;DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;;LOOP?
\$EOPCT: .WORD 1
BGT \$DOAGN ;;YES
MOV (PC)+,@(PC)+ ;;RESTORE COUNTER
\$ENDCT: .WORD 1
\$EOPCT
TYPE \$ENDMG ;;TYPE 'END PASS #'
MOV \$PASS,-(SP) ;;SAVE \$PASS FOR TYPEOUT
TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
TYPE \$ENULL ;;TYPE A NULL CHARACTER
\$GET42: MOV @#42,R0 ;;GET MONITOR ADDRESS
BEQ \$DOAGN ;;BRANCH IF NO MONITOR
RESET ;;CLEAR THE WORLD
\$ENDAD: JSR PC,(R0) ;;GO TO MONITOR
NOP ;;SAVE ROOM
NOP ;;FOR
NOP ;;ACT11
\$DOAGN: JMP @(PC)+ ;;RETURN
\$RTNAD: .WORD START1
\$ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
\$ENDMG: .ASCIZ <15><12>/END PASS #/

2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326

016172 005077 164332
016176 005077 164324
016202 005077 164316
016206 005077 164310
016212 005077 164302
016216 005077 164274
016222 000207
016224 010546

////////////////////////////////////
;SUBROUTINE TO CLEAR ALL UBE REG
////////////////////////////////////
CLRREG: CLR @BERE ;CLEAR ERROR CONDITIONS
CLR @BECR2 ;CLEAR BECR2 REG
CLR @BECR1 ;CLEAR BECR1 REG, EXCEPT RDY
CLR @BEBA ;CLEAR BEBA REG
CLR @BECC ;CLEAR BECC REG
CLR @BEBD ;CLEAR BEBD REG
RTS PC ;RETURN
////////////////////////////////////
;SUBROUTINE TO RESTORE TRAP CATCHER TO UBE VECTOR AREA
////////////////////////////////////
RCATCH: MOV R5,-(SP) ;SAVE R5 ON STACK


```

2327 016226 016705 164300      MOV INTVEC,R5          ;GET INT. VECTOR
2328 016232 005725              TST(R5)+              ;CALC. INTVEC+2
2329 016234 010577 164272      MOV R5,@INTVEC        ;PUT INTVEC+2 IN INTVEC
2330 016240 005015              CLR (R5)              ;PUT HALT IN INTVEC+2
2331 016242 012605              MOV (SP)+,R5         ;RESTORE R5
2332 016244 000207              RTS PC
2333
2334
2335
2336      ;////////////////////////////////////
2337      ;SUBROUTINE TO CHECK IF RDY BIT SET
2338      ;////////////////////////////////////
2339 016246 005004      CRDY:  CLR R4
2340 016250 005005              CLR R5
2341 016252 005205      2$:   INC R5          ;UPDATE COUNT
2342 016254 105777 164244      TSTB @BECR1          ;SEE IF RDY SET
2343 016260 100405              BMI 1$              ;BRANCH IF SET
2344 016262 032705 000200      BIT #200,R5         ;WAITED >100 MICROSECS?
2345 016266 001771              BEQ 2$              ;CONTINUE TO LOOK FOR RDY IF R5 NOT =128
2346 016270 012704 000001      MOV #1,R4           ;SET R4=1 TO INDICATE ERROR
2347 016274 000207      1$:   RTS PC          ;RETURN
2348
2349      ;////////////////////////////////////
2350      ;SUBROUTINE TO DISREGARD UBE INTERRUPTS
2351      ;////////////////////////////////////
2352 016276 016705 164230      DINT:  MOV INTVEC,R5  ;GET INTVEC AND
2353 016302 005725              TST (R5)+           ;CALC. INTVEC+2
2354 016304 010577 164222      MOV R5,@INTVEC      ;PUT ADDRESS OF NEXT LOC IN THIS ONE
2355 016310 012715 000002      MOV #2,(R5)         ;PUT AN RTI IN INTVEC+2
2356 016314 000207              RTS PC
2357      ;////////////////////////////////////
2358      ;SUBROUTINE TO RESTORE VECTOR AREA 0-56, 174, AND 176 FROM STACK AREA AND PUT TRAP CATCHER I
2359      ;////////////////////////////////////
2360 016316 016705 162702      RVEC:  MOV $TMP0,R5   ;GET AREA WHERE VECTOR STORED
2361 016322 005004              CLR R4              ;SET R4 =TO FIRST LOC
2362 016324 014524      1$:   MOV -(R5),(R4)+     ;RESTORE VECTORS
2363 016326 022704 000060      CMP #60,R4         ;AT END OF AREA?
2364 016332 001374              BNE 1$              ;BRANCH IF NO
2365 016334 014537 000174      MOV -(R5),@#174    ;RESTORE SOFTWARE SWR
2366 016340 014537 000176      MOV -(R5),@#176    ;
2367 016344 012704 000060      MOV #60,R4         ;SET R4 FOR FIRST TRAP CATCHER
2368 016350 012705 000062      MOV #62,R5         ;SET R5=TO FIRST TRAP CATCHER ADDRESS
2369 016354 010524      2$:   MOV R5,(R4)+       ;PUT ADDRESS OF NEXT LOC IN THIS ONE
2370 016356 005024              CLR(R4)+            ;PUT HALT IN NEXT LOC
2371 016360 022525              CMP (R5)+,(R5)+    ;INC R5 BY 4
2372 016362 022704 000174      CMP #174,R4        ;AT END OF VECTOR AREA?
2373 016366 001372              BNE 2$              ;BRANCH IF NO
2374 016370 012704 000200      MOV #200,R4        ;AS ABOVE, PUT TRAP CATCHER IN AREA 200-776
2375 016374 012705 000202      MOV #202,R5
2376 016400 010524      3$:   MOV R5,(R4)+
2377 016402 005024              CLR (R4)+
2378 016404 022525              CMP (R5)+,(R5)+
2379 016406 022704 001000      CMP #1000,R4
2380 016412 001372              BNE 3$
2381 016414 012737 000137 000200      MOV #137,@#200     ;RESTORE JMP @#START TO LOC 200
2382 016422 012737 002632 000202      MOV #START,@#202
2383 016430 000207              RTS PC              ;RETURN
    
```

```

2384 :////////////////////
2385 :SUBROUTINE TO TYPE PC OF ERROR MESSAGE
2386 :////////////////////
2387 016432 032777 020000 162532 TERRPC: BIT #SW13,@SWR ;INHIBITS ERROR TYPOUTS?
2388 016440 001013 BNE 1$ ;BRANCH IF YES
2389 016442 104401 027703 TYPE ,MSG15 ;PC OF ERROR MSG WAS:
2390 016446 016746 162476 MOV $ERRPC,-(SP) ;SAVE $ERRPC FOR TYPEOUT
016452 104402 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
2391 016454 104401 030026 TYPE ,MSG17 ;TEST NUMBER WAS:
2392 016460 016746 162450 MOV $TSTNM,-(SP) ;SAVE $TSTNM FOR TYPEOUT
2393 016464 104403 TYPOS ;GO TYPE -OCTAL ASCII
2394 016466 002 .BYTE 2 ;TYPE 2 DIGITS
2395 016467 000 .BYTE 0 ;SUPPRESS LEADING ZEROS
2396 016470 000207 1$: RTS PC
2397
2398

```

.SBTTL SCOPE HANDLER ROUTINE

```

:*****
:*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
:*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
:*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
:*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
:*SW14=1 LOOP ON TEST
:*SW11=1 INHIBIT ITERATIONS
:*SW09=1 LOOP ON ERROR
:*CALL
:* SCOPE ;:SCOPE=IOT

```

```

016472 $SCOPE:
016472 032777 040000 162472 1$: BIT #BIT14,@SWR ;:LOOP ON PRESENT TEST?
016500 001101 BNE $OVER ;:YES IF SW14=1
016502 000416 ;#####START OF CODE FOR THE XOR TESTER#####
$XTSTR: BR 6$ ;:IF RUNNING ON THE 'XOR' TESTER CHANGE
016504 013746 000004 MOV @#ERRVEC,-(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
016510 012737 016530 000004 MOV #5$,@#ERRVEC ;:SET FOR TIMEOUT
016516 005737 177060 TST @#177060 ;:TIME OUT ON XOR?
016522 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
016526 000453 BR $SVLAD ;:GO TO THE NEXT TEST
016530 022626 5$: CMP (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
016532 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
016536 000413 BR 7$ ;:LOOP ON THE PRESENT TEST
016540 6$:;#####END OF CODE FOR THE XOR TESTER#####
016540 105767 162371 2$: TSTB $ERFLG ;:HAS AN ERROR OCCURRED?
016544 001421 BEQ 3$ ;:BR IF NO
016546 126767 162375 162361 CMPB $ERMAX,$ERFLG ;:MAX. ERRORS FOR THIS TEST OCCURRED?
016554 101015 BHI 3$ ;:BR IF NO
016556 032777 001000 162406 BIT #BIT09,@SWR ;:LOOP ON ERROR?
016564 001404 BEQ 4$ ;:BR IF NO
016566 016767 162350 162344 7$: MOV $LPERR,$LPADR ;:SET LOOP ADDRESS TO LAST SCOPE
016574 000443 BR $OVER
016576 105067 162333 4$: CLRB $ERFLG ;:ZERO THE ERROR FLAG
016602 005067 162426 CLR $TIMES ;:CLEAR THE NUMBER OF ITERATIONS TO MAKE
016606 000415 BR 1$ ;:ESCAPE TO THE NEXT TEST
016610 032777 004000 162354 3$: BIT #BIT11,@SWR ;:INHIBIT ITERATIONS?
016616 001011 BNE 1$ ;:BR IF YES
016620 005767 162306 TST $PASS ;:IF FIRST PASS OF PROGRAM

```

```

016624 001406          BEQ      1$          ;; INHIBIT ITERATIONS
016626 005267 162304   INC      $ICNT       ;; INCREMENT ITERATION COUNT
016632 026767 162376 162276  CMP     $TIMES,$ICNT  ;; CHECK THE NUMBER OF ITERATIONS MADE
016640 002021          BGE     $OVER        ;; BR IF MORE ITERATION REQUIRED
016642 012767 000001 162266 1$:  MOV     #1,$ICNT     ;; REINITIALIZE THE ITERATION COUNTER
016650 016767 000044 162356   MOV     $MXCNT,$TIMES ;; SET NUMBER OF ITERATIONS TO DO
016656 105267 162252   $SVLAD: INCB    $STNM    ;; COUNT TEST NUMBERS
016662 011667 162252   MOV     (SP),$LPADR  ;; SAVE SCOPE LOOP ADDRESS
016666 011667 162250   MOV     (SP),$LPERR  ;; SAVE ERROR LOOP ADDRESS
016672 005067 162340   CLR     $ESCAPE     ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
016676 112767 000001 162243   MOV     #1,$ERMAX   ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
016704 016777 162224 162262 $OVER:  MOV     $STNM,@DISPLAY ;; DISPLAY TEST NUMBER
016712 016716 162222   MOV     $LPADR,(SP) ;; FUDGE RETURN ADDRESS
016716 000002          RTI              ;; FIXES PS
016720 000012   $MXCNT: 10.        ;; MAX. NUMBER OF ITERATIONS
2399   .SBTTL  ERROR HANDLER ROUTINE

```

```

*****
*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
*AND GO TO $ERRTYP ON ERROR
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW15=1      HALT ON ERROR
*SW13=1      INHIBIT ERROR TYPEOUTS
*SW10=1      BELL ON ERROR
*SW09=1      LOOP ON ERROR
*CALL
*          ERROR  N          ;;ERROR=EMT AND N=ERROR ITEM NUMBER

```

```

016722          $ERROR:
016722 105267 162207   7$:  INCB    $ERFLG     ;; SET THE ERROR FLAG
016726 001775          BEQ      7$          ;; DON'T LET THE FLAG GO TO ZERO
016730 016777 162200 162236   MOV     $STNM,@DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
016736 032777 002000 162226   BIT     #BIT10,@SWR   ;; BELL ON ERROR?
016744 001402          BEQ      1$          ;; NO - SKIP
016746 104401 001240   TYPE    $BELL       ;; RING BELL
016752 005267 162166   1$:  INC     $ERTTL     ;; COUNT THE NUMBER OF ERRORS
016756 011667 162166   MOV     (SP),$ERRPC  ;; GET ADDRESS OF ERROR INSTRUCTION
016762 162767 000002 162160   SUB     #2,$ERRPC
016770 117767 162154 162150   MOV     @ERRPC,$ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
016776 032777 020000 162166   BIT     #BIT13,@SWR  ;; SKIP TYPEOUT IF SET
017004 001004          BNE     20$         ;; SKIP TYPEOUTS
017006 004767 000056   JSR    PC,$ERRTYP   ;; GO TO USER ERROR ROUTINE
017012 104401 001245   TYPE    $CRLF
017016          20$:
017016 005777 162150   2$:  TST     @SWR        ;; HALT ON ERROR
017022 100001          BPL     3$          ;; SKIP IF CONTINUE
017024 000000          HALT
017026 032777 001000 162136 3$:  BIT     #BIT09,@SWR  ;; LOOP ON ERROR SWITCH SET?
017034 001402          BEQ     4$          ;; BR IF NO
017036 016716 162100   MOV     $LPERR,(SP)  ;; FUDGE RETURN FOR LOOPING
017042 005767 162170   4$:  TST     $ESCAPE     ;; CHECK FOR AN ESCAPE ADDRESS
017046 001402          BEQ     5$          ;; BR IF NONE
017050 016716 162162   MOV     $ESCAPE,(SP) ;; FUDGE RETURN ADDRESS FOR ESCAPE
017054          5$:
017054 022737 016136 000042   CMP     #$ENDAD,@#42 ;; ACT-11 AUTO-ACCEPT?
017062 001001          BNE     6$          ;; BRANCH IF NO

```

017064 000000
 017066 000002
 2400

017070
 017070 104401 001245
 017074 010046
 017076 005000
 017100 153700 001146
 017104 001004

017106 016746 162036

017112 104402
 017114 000425
 017116 005300
 017120 006300
 017122 006300
 017124 006300
 017126 062700 001250
 017132 012067 000004
 017136 001404
 017140 104401
 017142 000000
 017144 104401 001245
 017150 012067 000004
 017154 001404
 017156 104401
 017160 000000
 017162 104401 001245
 017166 011000
 017170 001004
 017172 012600
 017174 104401 001245
 017200 000207
 017202
 017202 013046
 017204 104402
 017206 005710
 017210 001770
 017212 104401 017220
 017216 000771
 017220 040 040 000

```

        HALT                ;;YES
6$:     RTI                 ;;RETURN
        .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

        ;*****
        ;*THIS ROUTINE USES THE 'ITEM CONTROL BYTE' ($ITEMB) TO DETERMINE WHICH
        ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE 'ERROR TABLE' ($ERRTB),
        ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
        ;*****

        $ERRTYP:
        TYPE      ,$CRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
        MOV       RO,-(SP)    ;;SAVE RO
        CLR       RO          ;;PICKUP THE ITEM INDEX
        BISB     @#$ITEMB,RO
        BNE      1$          ;;IF ITEM NUMBER IS ZERO, JUST
                               ;;TYPE THE PC OF THE ERROR
        MOV      $ERRPC,-(SP) ;;SAVE $ERRPC FOR TYPEOUT
                               ;;ERROR ADDRESS
                               ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
        TYPCC
        BR       6$          ;;GET OUT
        1$:     DEC       RO    ;;ADJUST THE INDEX SO THAT IT WILL
        ASL     RO          ;;WORK FOR THE ERROR TABLE
        ASL     RO
        ASL     RO
        ADD     # $ERRTB,RO  ;;FORM TABLE POINTER
        MOV     (RO)+,2$    ;;PICKUP 'ERROR MESSAGE' POINTER
        BEQ     3$          ;;SKIP TYPEOUT IF NO POINTER
        TYPE   'ERROR MESSAGE'
        2$:     .WORD    0    ;;'ERROR MESSAGE' POINTER GOES HERE
        TYPE   ,$CRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
        3$:     MOV     (RO)+,4$ ;;PICKUP 'DATA HEADER' POINTER
        BEQ     5$          ;;SKIP TYPEOUT IF 0
        TYPE   'DATA HEADER'
        4$:     .WORD    0    ;;'DATA HEADER' POINTER GOES HERE
        TYPE   ,$CRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
        5$:     MOV     (RO),RO ;;PICKUP 'DATA TABLE' POINTER
        BNE     7$          ;;GO TYPE THE DATA
        6$:     MOV     (SP)+,RO ;;RESTORE RO
        TYPE   ,$CRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
        RTS     PC          ;;RETURN
        7$:     MOV     @ (RO)+,-(SP) ;;SAVE @ (RO)+ FOR TYPEOUT
        TYPCC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
        TST     (RO)      ;;IS THERE ANOTHER NUMBER?
        BEQ     6$        ;;BR IF NO
        TYPE   ,8$        ;;TYPE TWO(2) SPACES
        BR     7$        ;;LOOP
        8$:     .ASCIZ  / /  ;;TWO(2) SPACES
        .EVEN
        .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

        ;*****
        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
        ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
        ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
        ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
    
```

2401

```

        :*REPLACED WITH SPACES.
        :*CALL:
        :*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
        :*      TYPDS      ;;GO TO THE ROUTINE

017224      $TYPDS:
017224      010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
017226      010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
017230      010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
017232      010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
017234      010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
017236      012746      020200      MOV      #20200,-(SP)      ;;SET BLANK SWITCH AND SIGN
017242      016605      000020      MOV      20(SP),R5      ;;GET THE INPUT NUMBER
017246      100004      BPL      1$      ;;BR IF INPUT IS POS.
017250      005405      NEG      R5      ;;MAKE THE BINARY NUMBER POS.
017252      112766      000055      000001      MOVVB   #'-,1(SP)      ;;MAKE THE ASCII NUMBER NEG.
017260      005000      1$:      CLR      R0      ;;ZERO THE CONSTANTS INDEX
017262      012703      017440      MOV      #SDBLK,R3      ;;SETUP THE OUTPUT POINTER
017266      112723      000040      MOVVB   #' ,(R3)+      ;;SET THE FIRST CHARACTER TO A BLANK
017272      005002      2$:      CLR      R2      ;;CLEAR THE BCD NUMBER
017274      016001      017430      MOV      $DTBL(R0),R1      ;;GET THE CONSTANT
017300      160105      3$:      SUB      R1,R5      ;;FORM THIS BCD DIGIT
017302      002402      BLT      4$      ;;BR IF DONE
017304      005202      INC      R2      ;;INCREASE THE BCD DIGIT BY 1
017306      000774      BR       3$
017310      060105      4$:      ADD      R1,R5      ;;ADD BACK THE CONSTANT
017312      005702      TST      R2      ;;CHECK IF BCD DIGIT=0
017314      001002      BNE      5$      ;;FALL THROUGH IF 0
017316      105716      TSTB    (SP)      ;;STILL DOING LEADING 0'S?
017320      100407      BMI      7$      ;;BR IF YES
017322      106316      5$:      ASLB    (SP)      ;;MSD?
017324      103003      BCC      6$      ;;BR IF NO
017326      116663      000001      177777      MOVVB   1(SP),-1(R3)      ;;YES--SET THE SIGN
017334      052702      000060      6$:      BIS      #'0,R2      ;;MAKE THE BCD DIGIT ASCII
017340      052702      000040      7$:      BIS      #' ,R2      ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
017344      110223      MOVVB   R2,(R3)+      ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
017346      005720      TST      (R0)+      ;;JUST INCREMENTING
017350      020027      000010      CMP      R0,#10      ;;CHECK THE TABLE INDEX
017354      002746      BLT      2$      ;;GO DO THE NEXT DIGIT
017356      003002      BGT      8$      ;;GO TO EXIT
017360      010502      MOV      R5,R2      ;;GET THE LSD
017362      000764      BR       6$      ;;GO CHANGE TO ASCII
017364      105726      8$:      TSTB    (SP)+      ;;WAS THE LSD THE FIRST NON-ZERO?
017366      100003      9$:      BPL      9$      ;;BR IF NO
017370      116663      177777      177776      MOVVB   -1(SP),-2(R3)      ;;YES--SET THE SIGN FOR TYPING
017376      105013      CLRB    (R3)      ;;SET THE TERMINATOR
017400      012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
017402      012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
017404      012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
017406      012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
017410      012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
017412      104401      017440      TYPE    ,SDBLK      ;;NOW TYPE THE NUMBER
017416      016666      000002      000004      MOV      2(SP),4(SP)      ;;ADJUST THE STACK
017424      012616      MOV      (SP)+,(SP)
017426      000002      RTI
017430      023420      $DTBL: 10000.
017432      001750      1000.
    
```

017434 000144
 017436 000012
 017440
 2402

100.
 10.
 \$DBLK: .BLKW 4
 .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
*

```

017450	105767	161535	\$TYPE:	TSTB	\$TPFLG	::IS THERE A TERMINAL?
017454	100002			BPL	1\$::BR IF YES
017456	000000			HALT		::HALT HERE IF NO TERMINAL
017460	000407			BR	3\$::LEAVE
017462	010046		1\$:	MOV	RO,-(SP)	::SAVE RO
017464	017600	000002		MOV	@2(SP),RO	::GET ADDRESS OF ASCIZ STRING
017470	112046		2\$:	MOVB	(RO)+,-(SP)	::PUSH CHARACTER TO BE TYPED ONTO STACK
017472	001005			BNE	4\$::BR IF IT ISN'T THE TERMINATOR
017474	005726			TST	(SP)+	::IF TERMINATOR POP IT OFF THE STACK
017476	012600		60\$:	MOV	(SP)+,RO	::RESTORE RO
017500	062716	000002	3\$:	ADD	#2,(SP)	::ADJUST RETURN PC
017504	000002			RTI		::RETURN
017506	122716	000011	4\$:	CMPB	#HT,(SP)	::BRANCH IF <HT>
017512	001430			BEQ	8\$	
017514	122716	000200		CMPB	#CRLF,(SP)	::BRANCH IF NOT <CRLF>
017520	001006			BNE	5\$	
017522	005726			TST	(SP)+	::POP <CR><LF> EQUIV
017524	104401			TYPE		::TYPE A CR AND LF
017526	001245			\$CRLF		
017530	105067	000200		CLRB	\$CHARCNT	::CLEAR CHARACTER COUNT
017534	000755			BR	2\$::GET NEXT CHARACTER
017536	004767	000056	5\$:	JSR	PC,\$TYPEC	::GO TYPE THIS CHARACTER
017542	126726	161442	6\$:	CMPB	\$FILLC,(SP)+	::IS IT TIME FOR FILLER CHARS.?
017546	001350			BNE	2\$::IF NO GO GET NEXT CHAR.
017550	016746	161432		MOV	\$NULL,-(SP)	::GET # OF FILLER CHARS. NEEDED
						::AND THE NULL CHAR.
017554	105366	000001	7\$:	DECB	1(SP)	::DOES A NULL NEED TO BE TYPED?
017560	002770			BLT	6\$::BR IF NO--GO POP THE NULL OFF OF STACK
017562	004767	000032		JSR	PC,\$TYPEC	::GO TYPE A NULL
017566	105367	000142		DECB	\$CHARCNT	::DO NOT COUNT AS A COUNT
017572	000770			BR	7\$::LOOP

;HORIZONTAL TAB PROCESSOR

017574	112716	000040	8\$:	MOVB	#,(SP)	::REPLACE TAB WITH SPACE
017600	004767	000014	9\$:	JSR	PC,\$TYPEC	::TYPE A SPACE
017604	132767	000007 000122		BITB	#7,\$CHARCNT	::BRANCH IF NOT AT

TYPE ROUTINE

```

017612 001372          BNE      9$          ;;TAB STOP
017614 005726          TST      (SP)+        ;;POP SPACE OFF STACK
017616 000724          BR       2$          ;;GET NEXT CHARACTER
017620 105777 161356   $TYPEC: TSTB   @$TPS        ;;WAIT UNTIL PRINTER IS READY
017624 100375          BPL      $TYPEC
017626 116677 000002 161350 MOVB   2(SP),@$TPB    ;;LOAD CHAR TO BE TYPED INTO DATA REG.
017634 105777 161336   TSTB   @$TKS        ;;SEE IF KEYBOARD IS TALKING.
017640 100021          BPL      2$          ;;BRANCH IF IT ISN'T.
017642 017746 161332   MOV     @$TKB,-(SP)   ;;PUSH CHARACTER ONTO STACK.
017646 042716 177600   BIC    #177600,(SP)  ;;BIT CLEAR TOP BYTE AND PARITY BIT.
017652 022726 000023   CMP    #23,(SP)+    ;;SEE IF THIS IS A ^S.
017656 001012          BNE      2$          ;;BRANCH TO CONTINUE IF IT ISN'T.
017660 105777 161312   3$:    TSTB   @$TKS        ;;WAIT FOR ANOTHER INPUT.
017664 100375          BPL      3$          ;;BRANCH BACK IF NOT READY.
017666 017746 161306   MOV     @$TKB,-(SP)  ;;PUSH NEXT CHARACTER ON STACK.
017672 042716 177600   BIC    #177600,(SP)  ;;BIT CLEAR TOP BYTE AND PARITY BIT.
017676 022726 000021   CMP    #21,(SP)+    ;;SEE IF THIS IS A ^Q.
017702 001366          BNE      3$          ;;BRANCH BACK FOR MORE WAIT IF NOT.
017704 122766 000015 000002 2$:    CMPB   #CR,2(SP)    ;;IS CHARACTER A CARRIAGE RETURN?
017712 001003          BNE      1$          ;;BRANCH IF NO
017714 105067 000014   CLRB   $CHARCNT     ;;YES--CLEAR CHARACTER COUNT
017720 000406          BR       $TYPEX
017722 122766 000012 000002 1$:    CMPB   #LF,2(SP)    ;;IS CHARACTER A LINE FEED?
017730 001402          BEQ     $TYPEX       ;;BRANCH IF YES
017732 105227          INCB   (PC)+        ;;COUNT THE CHARACTER
017734 000000          $CHARCNT: .WORD 0  ;;CHARACTER COUNT STORAGE
017736 000207          $TYPEX: RTS      PC

```

2403

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*   MOV     NUM,-(SP)          ;;NUMBER TO BE TYPED
*   TYPOS   ;;CALL FOR TYPEOUT
*   .BYTE   N                  ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*   .BYTE   M                  ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS
*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*$TYPOS OR $TYPOC
*CALL:
*   MOV     NUM,-(SP)          ;;NUMBER TO BE TYPED
*   TYPON   ;;CALL FOR TYPEOUT
*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
*CALL:
*   MOV     NUM,-(SP)          ;;NUMBER TO BE TYPED
*   TYPOC   ;;CALL FOR TYPEOUT
017740 017646 000000          $TYPOS: MOV     @(SP),-(SP)    ;;PICKUP THE MODE
017744 116667 000001 000211 MOVB   1(SP),$OFILL  ;;LOAD ZERO FILL SWITCH
017752 112667 000207          MOVB   (SP)+,$OMODE+1 ;;NUMBER OF DIGITS TO TYPE
017756 062716 000002          ADD    #2,(SP)      ;;ADJUST RETURN ADDRESS

```

```

017762 000406          BR          $TYPON
017764 112767 000001 000171 $TYPOC: MOVB   #1,$OFILL      ;;SET THE ZERG FILL SWITCH
017772 112767 000006 000165          MOVB   #6,$OMODE+1  ;;SET FOR SIX(6) DIGITS
020000 112767 000005 000154 $TYPON: MOVB   #5,$OCNT  ;;SET THE ITERATION COUNT
020006 010346          MOV     R3,-(SP)    ;;SAVE R3
020010 010446          MOV     R4,-(SP)    ;;SAVE R4
020012 010546          MOV     R5,-(SP)    ;;SAVE R5
020014 116704 000145          MOVB   $OMODE+1,R4  ;;GET THE NUMBER OF DIGITS TO TYPE
020020 005404          NEG     R4
020022 062704 000006          ADD     #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
020026 110467 000132          MOVB   R4,$OMODE  ;;SAVE IT FOR USE
020032 116704 000125          MOVB   $OFILL,R4  ;;GET THE ZERO FILL SWITCH
020036 016605 000012          MOV     12(SP),R5  ;;PICKUP THE INPUT NUMBER
020042 005003          CLR     R3        ;;CLEAR THE OUTPUT WORD
020044 006105          1$:   ROL     R5        ;;ROTATE MSB INTO 'C'
020046 000404          BR          3$
020050 006105          2$:   ROL     R5        ;;FORM THIS DIGIT
020052 006105          ROL     R5
020054 006105          ROL     R5
020056 010503          MOV     R5,R3
020060 006103          3$:   ROL     R3        ;;GET LSB OF THIS DIGIT
020062 105367 000076          DECB   $OMODE    ;;TYPE THIS DIGIT?
020066 100016          BPL     7$        ;;BR IF NO
020070 042703 177770          BIC     #177770,R3  ;;GET RID OF JUNK
020074 001002          BNE     4$        ;;TEST FOR 0
020076 005704          TST     R4        ;;SUPPRESS THIS 0?
020100 001403          BEQ     5$        ;;BR IF YES
020102 005204          4$:   INC     R4        ;;DON'T SUPPRESS ANYMORE 0'S
020104 052703 000060          BIS     #'0,R3    ;;MAKE THIS DIGIT ASCII
020110 052703 000040          5$:   BIS     #' ,R3  ;;MAKE ASCII IF NOT ALREADY
020114 110367 000040          MOVB   R3,8$     ;;SAVE FOR TYPING
020120 104401 020160          TYPE   ,8$      ;;GO TYPE THIS DIGIT
020124 105367 000032          7$:   DECB   $OCNT  ;;COUNT BY 1
020130 003347          BGT     2$        ;;BR IF MORE TO DO
020132 002402          BLT     6$        ;;BR IF DONE
020134 005204          INC     R4        ;;INSURE LAST DIGIT ISN'T A BLANK
020136 000744          BR      2$        ;;GO DO THE LAST DIGIT
020140 012605          6$:   MOV     (SP)+,R5  ;;RESTORE R5
020142 012604          MOV     (SP)+,R4  ;;RESTORE R4
020144 012603          MOV     (SP)+,R3  ;;RESTORE R3
020146 016666 000002 000004          MOV     2(SP),4(SP)  ;;SET THE STACK FOR RETURNING
020154 012616          MOV     (SP)+,(SP)
020156 000002          RTI
020160 000          8$:   .BYTE  0        ;;RETURN
020161 000          .BYTE  0        ;;STORAGE FOR ASCII DIGIT
020162 000          .BYTE  0        ;;TERMINATOR FOR TYPE ROUTINE
020163 000          $OCNT: .BYTE  0  ;;OCTAL DIGIT COUNTER
020164 000000          $OFILL: .BYTE  0  ;;ZERO FILL SWITCH
          $OMODE: .WORD 0  ;;NUMBER OF DIGITS TO TYPE
          .SBTTL TRAP DECODER

```

2404

```

*****
*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
*GO TO THAT ROUTINE.

```

```

020166 010046          $TRAP: MOV     R0,-(SP)  ;;SAVE R0

```



```

020170 016600 000002      MOV      2(SP),R0      ;;GET TRAP ADDRESS
020174 005740             TST      -(R0)        ;;BACKUP BY 2
020176 111000             MOVVB   (R0),R0       ;;GET RIGHT BYTE OF TRAP
020200 006300             ASL     R0            ;;POSITION FOR INDEXING
020202 016000 020222      MOV     $TRPAD(R0),R0 ;;INDEX TO TABLE
020206 000200             RTS     R0           ;;GO TO ROUTINE
    
```

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

```

020210 011646 000004 000002 $TRAP2: MOV    (SP),-(SP)  ;;MOVE THE PC DOWN
020212 016666             MOV    4(SP),2(SP)    ;;MOVE THE PSW DOWN
020220 000002             RTI                    ;;RESTORE THE PSW
    
```

.SBTTL TRAP TABLE

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

```

:          ROUTINE
:          -----
020222 020210 $TRPAD: .WORD  $TRAP2
020224 017450      $TYPE  ;;CALL=TYPE      TRAP+1(104401) TTY TYPEOUT ROUTINE
020226 017764      $TYPOC  ;;CALL=TYPOC     TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
020230 017740      $TYPOS  ;;CALL=TYPOS     TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
020232 020000      $TYPON  ;;CALL=TYPON     TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
020234 017224      $TYPDS  ;;CALL=TYPDS     TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
    
```

2405

.SBTTL POWER DOWN AND UP ROUTINES

```

:*****
:POWER DOWN ROUTINE
020236 012737 020376 000024 $PWRDN: MOV    # $ILLUP,@#PWRVEC ;;SET FOR FAST UP
020244 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
020252 010046             MOV    R0,-(SP)      ;;PUSH R0 ON STACK
020254 010146             MOV    R1,-(SP)      ;;PUSH R1 ON STACK
020256 010246             MOV    R2,-(SP)      ;;PUSH R2 ON STACK
020260 010346             MOV    R3,-(SP)      ;;PUSH R3 ON STACK
020262 010446             MOV    R4,-(SP)      ;;PUSH R4 ON STACK
020264 010546             MOV    R5,-(SP)      ;;PUSH R5 ON STACK
020266 017746 160700      MOV    @SWR,-(SP)    ;;PUSH @SWR ON STACK
020272 010667 000104      MOV    SP,$SAVR6    ;;SAVE SP
020276 012737 020310 000024      MOV    # $PWRUP,@#PWRVEC ;;SET UP VECTOR
020304 000000             HALT
020306 000776             BR     -2           ;;HANG UP
    
```

```

:*****
:POWER UP ROUTINE
020310 012737 020376 000024 $PWRUP: MOV    # $ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
020316 016706 000060      MOV    $SAVR6,SP    ;;GET SP
020322 005067 000054      CLR    $SAVR6      ;;WAIT LOOP FOR THE TTY
020326 005267 000050      1$:  INC    $SAVR6    ;;WAIT FOR THE INC
020332 001375             BNE   1$           ;;OF WORD
020334 012677 160632      MOV    (SP)+,@SWR   ;;POP STACK INTO @SWR
020340 012605             MOV    (SP)+,R5     ;;POP STACK INTO R5
020342 012604             MOV    (SP)+,R4     ;;POP STACK INTO R4
    
```

```

020344 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
020346 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
020350 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
020352 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
020354 012737 020236 000024      MOV      #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
020362 012737 000340 000026      MOV      #340,@#PWRVEC+2 ;;PRIO:7
020370 104401      TYPE      ;;REPORT THE POWER FAILURE
020372 020404      $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
020374 000002      RTI
020376 000000      $ILLUP: HALT      ;;THE POWER UP SEQUENCE WAS STARTED
020400 000776      BR      -2      ;; BEFORE THE POWER DOWN WAS COMPLETE
020402 000000      $SAVR6: 0      ;;PUT THE SP HERE
020404      015      012      120
020407      117      127      105
020412      122      000
    
```

2406

```

      .EVEN
      ;;*****
      ;;*****
    
```

2407 020414 015 012 105
 020417 115 101 120

```

MSG1: .ASCIZ<15><12>/EMAP: /
    
```

2408 020422 072 040 000
 020425 015 012 101
 020430 114 114 040
 020433 105 130 105
 020436 122 103 111
 020441 123 117 122
 020444 123 040 124
 020447 105 123 124
 020452 105 104 015

```

MSG2: .ASCII<15><12>/ALL EXERCISORS TESTED/<15><12>
    
```

2409 020456 040 040 040
 020461 116 117 124
 020464 105 072 124
 020467 117 040 124
 020472 105 123 124
 020475 040 120 101
 020500 123 123 111
 020503 116 107 040
 020506 117 106 040
 020511 107 122 101
 020514 116 124 123
 020517 040 106 117
 020522 122 040 124
 020525 110 105 040
 020530 114 101 123
 020533 124 040 125
 020536 102 105 015

```

      .ASCII/ NOTE:TO TEST PASSING OF GRANTS FOR THE LAST UBE/<15><12>
    
```

2410 020542 040 040 040
 020545 040 040 040
 020550 040 040 111
 020553 124 040 123
 020556 110 117 125
 020561 114 104 040
 020564 102 105 040
 020567 123 127 101
 020572 120 120 105

```

      .ASCII/ IT SHOULD BE SWAPPED WITH A UBE/<15><12>
    
```

	020575	104	040	127	
	020600	111	124	110	
	020603	040	101	040	
	020606	125	102	105	
	020611	015	012		
2411	020613	040	040	040	.ASCIZ/ OF HIGHER ELECTRICAL PRIORITY/<15><12>
	020616	040	040	040	
	020621	040	040	117	
	020624	106	040	110	
	020627	111	107	110	
	020632	105	122	040	
	020635	105	114	105	
	020640	103	124	122	
	020643	111	103	101	
	020646	114	040	120	
	020651	122	111	117	
	020654	122	111	124	
	020657	131	015	012	
	020662	000			
2412	020663	015	012	102	MSG5: .ASCIZ<15><12>*BUS PARITY NOT TESTED ON 11/05 OR 11/20 MACHINES*<15><12>
	020666	125	123	040	
	020671	120	101	122	
	020674	111	124	131	
	020677	040	116	117	
	020702	124	040	124	
	020705	105	123	124	
	020710	105	104	040	
	020713	117	116	040	
	020716	061	061	057	
	020721	060	065	040	
	020724	117	122	040	
	020727	061	061	057	
	020732	062	060	040	
	020735	115	101	103	
	020740	110	111	116	
	020743	105	123	015	
	020746	012	000		
2413	020750	116	117	040	EM1: .ASCIZ/NO RESPONSE TO REG ADDRESSES OR NO DEVICE PRESENT/
	020753	122	105	123	
	020756	120	117	116	
	020761	123	105	040	
	020764	124	117	040	
	020767	122	105	107	
	020772	040	101	104	
	020775	104	122	105	
	021000	123	123	105	
	021003	123	040	117	
	021006	122	040	116	
	021011	117	040	104	
	021014	105	126	111	
	021017	103	105	040	
	021022	120	122	105	
	021025	123	105	116	
	021030	124	000		
2414	021032	106	101	124	EM2: .ASCIZ/FATAL ERROR:REG FAILED TO CLEAR/
	021035	101	114	040	
	021040	105	122	122	

	021043	117	122	072		
	021046	122	105	107		
	021051	040	106	101		
	021054	111	114	105		
	021057	104	040	124		
	021062	117	040	103		
	021065	114	105	101		
	021070	122	000			
2415	021072	122	105	107	DH2:	.ASCIZ*REG ADD/REG CONTENTS *
	021075	040	101	104		
	021100	104	057	122		
	021103	105	107	040		
	021106	103	117	116		
	021111	124	105	116		
	021114	124	123	040		
	021117	000				
2416						.EVEN
2417	021120	001214	001216	000000	DT2:	.WORD \$REG0,\$REG1,0
2418	021126	106	101	124	EM3:	.ASCIZ/FATAL ERROR:CPU DID NOT RECEIVE SSYN/
	021131	101	114	040		
	021134	105	122	122		
	021137	117	122	072		
	021142	103	120	125		
	021145	040	104	111		
	021150	104	040	116		
	021153	117	124	040		
	021156	122	105	103		
	021161	105	111	126		
	021164	105	040	123		
	021167	123	131	116		
	021172	000				
2419	021173	120	103	040	DH3:	.ASCIZ/PC WAS/
	021176	127	101	123		
	021201	000				
2420						.EVEN
2421	021202	001214	000000		DT3:	.WORD \$REG0,0
2422	021206	106	101	124	EM4:	.ASCIZ/FATAL ERROR:REG FAILED TO FLOAT A '1'/'
	021211	101	114	040		
	021214	105	122	122		
	021217	117	122	072		
	021222	122	105	107		
	021225	040	106	101		
	021230	111	114	105		
	021233	104	040	124		
	021236	117	040	106		
	021241	114	117	101		
	021244	124	040	101		
	021247	040	047	061		
	021252	047	000			
2423	021254	122	105	107	DH4:	.ASCIZ*REG ADD/DATA IS/DATA SHOULD BE*
	021257	040	101	104		
	021262	104	057	104		
	021265	101	124	101		
	021270	040	111	123		
	021273	057	104	101		
	021276	124	101	040		
	021301	123	110	117		

	021304	125	114	104	
	021307	040	102	105	
	021312	000			
2424					.EVEN
2425	021314	001214	001216	001220	DT4: .WORD \$REG0,\$REG1,\$REG2,0
	021322	000000			
2426	021324	106	101	124	EM5: .ASCIZ/FATAL ERROR:REG FAILED TO FLOAT A '0'/'
	021327	101	114	040	
	021332	105	122	122	
	021335	117	122	072	
	021340	122	105	107	
	021343	040	106	101	
	021346	111	114	105	
	021351	104	040	124	
	021354	117	040	106	
	021357	114	117	101	
	021362	124	040	101	
	021365	040	047	060	
	021370	047	000		
2427	021372	106	101	124	EM6: .ASCIZ/FATAL ERROR:CONTROL REG HELD WRONG DATA/
	021375	101	114	040	
	021400	105	122	122	
	021403	117	122	072	
	021406	103	117	116	
	021411	124	122	117	
	021414	114	040	122	
	021417	105	107	040	
	021422	110	105	114	
	021425	104	040	127	
	021430	122	117	116	
	021433	107	040	104	
	021436	101	124	101	
	021441	000			
2428	021442	106	101	124	EM7: .ASCIZ/FATAL ERROR:DUAL ADDRESSING ERROR/
	021445	101	114	040	
	021450	105	122	122	
	021453	117	122	072	
	021456	104	125	101	
	021461	114	040	101	
	021464	104	104	122	
	021467	105	123	123	
	021472	111	116	107	
	021475	040	105	122	
	021500	122	117	122	
	021503	000			
2429	021504	122	105	107	DH7: .ASCIZ*REG ADD/REG ADD WERE SIMULATANEOUSLY WRITTEN*
	021507	040	101	104	
	021512	104	057	122	
	021515	105	107	040	
	021520	101	104	104	
	021523	040	127	105	
	021526	122	105	040	
	021531	123	111	115	
	021534	125	114	101	
	021537	124	101	116	
	021542	105	117	125	
	021545	123	114	131	

	021550	040	127	122	
	021553	111	124	124	
	021556	105	116	000	
2430					
2431	021562	001214	001216	000000	DT7: .EVEN
2432	021570	105	122	122	.WORD \$REG0,\$REG1,0
	021573	117	122	072	EM8: .ASCIZ/ERROR: SETTING PB PARITY FAILED TO CAUSE CPU TO TRAP/
	021576	040	123	105	
	021601	124	124	111	
	021604	116	107	040	
	021607	120	102	040	
	021612	120	101	122	
	021615	111	124	131	
	021620	040	106	101	
	021623	111	114	105	
	021626	104	040	124	
	021631	117	040	103	
	021634	101	125	123	
	021637	105	040	103	
	021642	120	125	040	
	021645	124	117	040	
	021650	124	122	101	
	021653	120	000		
2433	021655	105	122	122	EM9: .ASCIZ/ERROR: GO BIT FAILED TO LOAD '1'/'
	021660	117	122	072	
	021663	040	107	117	
	021666	040	102	111	
	021671	124	040	106	
	021674	101	111	114	
	021677	105	104	040	
	021702	124	117	040	
	021705	114	117	101	
	021710	104	040	047	
	021713	061	047	000	
2434	021716	105	122	122	EM10: .ASCIZ/ERROR: GO BIT FAILED TO LOAD '0'/'
	021721	117	122	072	
	021724	040	107	117	
	021727	040	102	111	
	021732	124	040	106	
	021735	101	111	114	
	021740	105	104	040	
	021743	124	117	040	
	021746	114	117	101	
	021751	104	040	047	
	021754	060	047	000	
2435	021757	106	101	124	EM11: .ASCIZ/FATAL ERROR: GO BIT FAILED TO CLEAR/
	021762	101	114	040	
	021765	105	122	122	
	021770	117	122	072	
	021773	040	107	117	
	021776	040	102	111	
	022001	124	040	106	
	022004	101	111	114	
	022007	105	104	040	
	022012	124	117	040	
	022015	103	114	105	
	022020	101	122	000	

2436	022023	106	101	124	EM12:	.ASCIZ/FATAL ERROR: READY BIT FAILED TO SET/
	022026	101	114	040		
	022031	105	122	122		
	022034	117	122	072		
	022037	040	122	105		
	022042	101	104	131		
	022045	040	102	111		
	022050	124	040	106		
	022053	101	111	114		
	022056	105	104	040		
	022061	124	117	040		
	022064	123	105	124		
	022067	000				
2437	022070	124	117	040	EM14:	.ASCIZ/TO CLEAR BIT 10 OF BECR1/
	022073	103	114	105		
	022076	101	122	040		
	022101	102	111	124		
	022104	040	061	060		
	022107	040	117	106		
	022112	040	102	105		
	022115	103	122	061		
	022120	000				
2438	022121	105	122	122	EM15:	.ASCIZ/ERROR: ERROR BITS IN BECR2 SET WHEN SHOULD BE CLEAR/
	022124	117	122	072		
	022127	040	105	122		
	022132	122	117	122		
	022135	040	102	111		
	022140	124	123	040		
	022143	111	116	040		
	022146	102	105	103		
	022151	122	062	040		
	022154	123	105	124		
	022157	040	127	110		
	022162	105	116	040		
	022165	123	110	117		
	022170	125	114	104		
	022173	040	102	105		
	022176	040	103	114		
	022201	105	101	122		
	022204	000				
2439	022205	103	117	116	DH15:	.ASCIZ/CONTENTS OF BECR2/
	022210	124	105	116		
	022213	124	123	040		
	022216	117	106	040		
	022221	102	105	103		
	022224	122	062	C00		
2440	022227	106	101	124	EM16:	.ASCIZ/FATAL ERROR: READY BIT FAILED TO CLEAR OR GO FAILED TO SET/
	022232	101	114	040		
	022235	105	122	122		
	022240	117	122	072		
	022243	040	122	105		
	022246	101	104	131		
	022251	040	102	111		
	022254	124	040	106		
	022257	101	111	114		
	022262	105	104	040		
	022265	124	117	040		

	022270	103	114	105	
	022273	101	122	040	
	022276	117	122	040	
	022301	107	117	040	
	022304	106	101	111	
	022307	114	105	104	
	022312	040	124	117	
	022315	040	123	105	
	022320	124	000		
2441	022322	105	122	122	EM17: .ASCIZ/ERROR: UBE FAILED TO INTERRUPT/
	022325	117	122	072	
	022330	040	125	102	
	022333	105	040	106	
	022336	101	111	114	
	022341	105	104	040	
	022344	124	117	040	
	022347	111	116	124	
	022352	105	122	122	
	022355	125	120	124	
	022360	000			
2442	022361	102	122	040	DH17: .ASCIZ*BR IS / PRIORITY IS*
	022364	111	123	040	
	022367	040	057	040	
	022372	120	122	111	
	022375	117	122	111	
	022400	124	131	040	
	022403	111	123	000	
2443	022406	105	122	122	EM18: .ASCIZ/ERROR: UBE INTERRUPTED WHEN PSW AT SAME PRIORITY LEVEL/
	022411	117	122	072	
	022414	040	125	102	
	022417	105	040	111	
	022422	116	124	105	
	022425	122	122	125	
	022430	120	124	105	
	022433	104	040	127	
	022436	110	105	116	
	022441	040	120	123	
	022444	127	040	101	
	022447	124	040	123	
	022452	101	115	105	
	022455	040	120	122	
	022460	111	117	122	
	022463	111	124	131	
	022466	040	114	105	
	022471	126	105	114	
	022474	000			
2444	022475	125	102	105	DH18: .ASCIZ/UBE BR WAS/
	022500	040	102	122	
	022503	040	127	101	
	022506	123	000		
2445	022510	105	122	122	EM19: .ASCIZ/ERROR: UBE FALSELY INTERRUPTED AT A HIGHER LEVEL/
	022513	117	122	072	
	022516	040	125	102	
	022521	105	040	106	
	022524	101	114	123	
	022527	105	114	131	
	022532	040	111	116	

	022535	124	105	122	
	022540	122	125	120	
	022543	124	105	104	
	022546	040	101	124	
	022551	040	101	040	
	022554	110	111	107	
	022557	110	105	122	
	022562	040	114	105	
	022565	126	105	114	
	022570	000			
2446	022571	110	111	107	DH19: .ASCIZ/HIGHER LEVEL WAS/
	022574	110	105	122	
	022577	040	114	105	
	022602	126	105	114	
	022605	040	127	101	
	022610	123	000		
2447	022612	105	122	122	EM20: .ASCIZ/ERROR: USE INTERRUPTED TO WRONG VECTOR/
	022615	117	122	072	
	022620	040	125	102	
	022623	105	040	111	
	022626	116	124	105	
	022631	122	122	125	
	022634	120	124	105	
	022637	104	040	124	
	022642	117	040	127	
	022645	122	117	116	
	022650	107	040	126	
	022653	105	103	124	
	022656	117	122	000	
2448					
2449	022661	105	122	122	EM21: .ASCIZ/ERROR: SIMULTANEOUS GO FAILED/
	022664	117	122	072	
	022667	040	123	111	
	022672	115	125	114	
	022675	124	101	116	
	022700	105	117	125	
	022703	123	040	107	
	022706	117	040	106	
	022711	101	111	114	
	022714	105	104	000	
2450	022717	105	122	122	EM22: .ASCIZ/ERROR: NO, NO SACK BIT FALSELY SET/
	022722	117	122	072	
	022725	040	116	117	
	022730	054	040	116	
	022733	117	040	123	
	022736	101	103	113	
	022741	040	102	111	
	022744	124	040	106	
	022747	101	114	123	
	022752	105	114	131	
	022755	040	123	105	
	022760	124	000		
2451	022762	105	122	122	EM23: .ASCIZ/ERROR: NO INT. SSYN BIT FALSELY SET/
	022765	117	122	072	
	022770	040	116	117	
	022773	040	111	116	
	022776	124	056	040	

	023001	123	123	131	
	023004	116	040	102	
	023007	111	124	040	
	023012	106	101	114	
	023015	123	105	114	
	023020	131	040	123	
	023023	105	124	000	
2452	023026	105	122	122	EM24: .ASCIZ/ERROR: DATI FAILED TO LOAD PROPER DATA/
	023031	117	122	072	
	023034	040	104	101	
	023037	124	111	040	
	023042	106	101	111	
	023045	114	105	104	
	023050	040	124	117	
	023053	040	114	117	
	023056	101	104	040	
	023061	120	122	117	
	023064	120	105	122	
	023067	040	104	101	
2453	023072	124	101	000	
	023075	102	105	102	DH24: .ASCIZ*BEED /MEM DATA/MEM ADD/DATA SHOULD BE IN MEM*
	023100	104	040	057	
	023103	115	105	115	
	023106	040	104	101	
	023111	124	101	057	
	023114	115	105	115	
	023117	040	101	104	
	023122	104	057	104	
	023125	101	124	101	
	023130	040	123	110	
	023133	117	125	114	
	023136	104	040	102	
	023141	105	040	111	
	023144	116	040	115	
	023147	105	115	000	
2454					.EVEN
2455	023152	001214	001216	001220	DT24: .WORD \$REG0,\$REG1,\$REG2,\$REG3,0
	023160	001222	000000		
2456	023164	104	101	124	EM25: .ASCIZ/DATO FAILED TO LOAD PROPER DATA/
	023167	117	040	106	
	023172	101	111	114	
	023175	105	104	040	
	023200	124	117	040	
	023203	114	117	101	
	023206	104	040	120	
	023211	122	117	120	
	023214	105	122	040	
	023217	104	101	124	
	023222	101	000		
2457	023224	104	101	124	EM26: .ASCIZ/DATIP FAILED TO LOAD PROPER DATA/
	023227	111	120	040	
	023232	106	101	111	
	023235	114	105	104	
	023240	040	124	117	
	023243	040	114	117	
	023246	101	104	040	
	023251	120	122	117	

	023254	120	105	122	
	023257	040	104	101	
	023262	124	101	000	
2458	023265	104	101	124	EM27: .ASCIZ/DATOB FILED TO LOAD PROPER DATA/
	023270	117	102	040	
	023273	106	111	114	
	023276	105	104	040	
	023301	124	117	040	
	023304	114	117	101	
	023307	104	040	120	
	023312	122	117	120	
	023315	105	122	040	
	023320	104	101	124	
	023323	101	000		
2459	023325	104	101	124	EM28: .ASCIZ/DATI FAILED TO SET RDY/
	023330	111	040	106	
	023333	101	111	114	
	023336	105	104	040	
	023341	124	117	040	
	023344	123	105	124	
	023347	040	122	104	
	023352	131	000		
2460	023354	104	101	124	EM29: .ASCIZ/DATO FAILED TO SET RDY/
	023357	117	040	106	
	023362	101	111	114	
	023365	105	104	040	
	023370	124	117	040	
	023373	123	105	124	
	023376	040	122	104	
	023401	131	000		
2461	023403	104	101	124	EM30: .ASCIZ/DATIP FAILED TO SET RDY/
	023406	111	120	040	
	023411	106	101	111	
	023414	114	105	104	
	023417	040	124	117	
	023422	040	123	105	
	023425	124	040	122	
	023430	104	131	000	
2462	023433	104	101	124	EM31: .ASCIZ/DATOB FAILED TO SET RDY/
	023436	117	102	040	
	023441	106	101	111	
	023444	114	105	104	
	023447	040	124	117	
	023452	040	123	105	
	023455	124	040	122	
	023460	104	131	000	
2463	023463	105	122	122	EM32: .ASCIZ/ERROR: INH. DATA SHIFT ON DATIP FAILED/
	023466	117	122	072	
	023471	040	111	116	
	023474	110	056	040	
	023477	104	101	124	
	023502	101	040	123	
	023505	110	111	106	
	023510	124	040	117	
	023513	116	040	104	
	023516	101	124	111	
	023521	120	040	106	

	023524	101	111	114	
	023527	105	104	000	
2464	023532	105	122	122	EM33: .ASCIZ/ERROR: DATOB ON DATIP FAILED/
	023535	117	122	072	
	023540	040	104	101	
	023543	124	117	102	
	023546	040	117	116	
	023551	040	104	101	
	023554	124	111	120	
	023557	040	106	101	
	023562	111	114	105	
	023565	104	000		
2465	023567	105	122	122	EM34: .ASCIZ/ERROR: UBE DID DATI FROM WRONG LOCATION/
	023572	117	122	072	
	023575	040	125	102	
	023600	105	040	104	
	023603	111	104	040	
	023606	104	101	124	
	023611	111	040	106	
	023614	122	117	115	
	023617	040	127	122	
	023622	117	116	107	
	023625	040	114	117	
	023630	103	101	124	
	023633	111	117	116	
	023636	000			
2466	023637	115	105	115	DH34: .ASCIZ/MEM LOC WANTED/
	023642	040	114	117	
	023645	103	040	127	
	023650	101	116	124	
	023653	105	104	000	
2467	023656	105	122	122	EM35: .ASCIZ/ERROR: BEBA LOWER 16 BITS DID NOT COUNT BY 2/
	023661	117	122	072	
	023664	040	102	105	
	023667	102	101	040	
	023672	114	117	127	
	023675	105	122	040	
	023700	061	066	040	
	023703	102	111	124	
	023706	123	040	104	
	023711	111	104	040	
	023714	116	117	124	
	023717	040	103	117	
	023722	125	116	124	
	023725	040	102	131	
	023730	040	062	000	
2468	023733	050	122	105	DH35: .ASCIZ*(REG) /DATA SHOULD BE*
	023736	107	051	040	
	023741	057	104	101	
	023744	124	101	040	
	023747	123	110	117	
	023752	125	114	104	
	023755	040	102	105	
	023760	000			
2469	023761	105	122	122	EM36: .ASCIZ/ERROR: BEBA BIT A16,17 DID NOT COUNT=0/
	023764	117	122	072	
	023767	040	102	105	

	024504	050	064	051	
	024507	000			
2478	024510	101	104	104	DH43: .ASCIZ*ADD FROM/TO WERE WRITTEN*
	024513	040	106	122	
	024516	117	115	057	
	024521	124	117	040	
	024524	127	105	122	
	024527	105	040	127	
	024532	122	111	124	
	024535	124	105	116	
	024540	000			
2479	024541	105	122	122	EM44: .ASCIZ/ERROR: INT. ON DONE BIT NOT CLEARED/
	024544	117	122	072	
	024547	040	111	116	
	024552	124	056	040	
	024555	117	116	040	
	024560	104	117	116	
	024563	105	040	102	
	024566	111	124	040	
	024571	116	117	124	
	024574	040	103	114	
	024577	105	101	122	
	024602	105	104	000	
2480	024605	105	122	122	EM45: .ASCIZ/ERROR: CCOVF NOT SET/
	024610	117	122	072	
	024613	040	103	103	
	024616	117	126	106	
	024621	040	116	117	
	024624	124	040	123	
	024627	105	124	000	
2481	024632	105	122	122	EM46: .ASCIZ/ERROR: DATO FROM BECC NOT DONE PROPERLY/
	024635	117	122	072	
	024640	040	104	101	
	024643	124	117	040	
	024646	106	122	117	
	024651	115	040	102	
	024654	105	103	103	
	024657	040	116	117	
	024662	124	040	104	
	024665	117	116	105	
	024670	040	120	122	
	024673	117	120	105	
	024676	122	114	131	
	024701	000			
2482	024702	101	104	104	DH46: .ASCIZ*ADD /DATA /DATA SHOULD BE*
	024705	040	040	040	
	024710	057	104	101	
	024713	124	101	040	
	024716	040	040	057	
	024721	104	101	124	
	024724	101	040	123	
	024727	110	117	125	
	024732	114	104	040	
	024735	102	105	000	
2483	024740	105	122	122	EM47: .ASCIZ/ERROR: UBE DID NOT DO 2 DATO FOR EACH REQUEST/
	024743	117	122	072	
	024746	040	125	102	

CZ
 SYI
 STI
 STI
 STI
 STI
 -
 ERI
 VII
 DYI
 ELI
 CZI

	025470	040	124	105	
	025473	123	124	000	
2492	025476	105	122	122	EM56: .ASCIZ/ERROR: TEST OF WRONG A LINES ERROR BIT FAILED/
	025501	117	122	072	
	025504	040	124	105	
	025507	123	124	040	
	025512	117	106	040	
	025515	127	122	117	
	025520	116	107	040	
	025523	101	040	040	
	025526	114	111	116	
	025531	105	123	040	
	025534	105	122	122	
	025537	117	122	040	
	025542	102	111	124	
	025545	040	106	101	
	025550	111	114	105	
	025553	104	000		
2493	025555	102	105	103	EM57: .ASCIZ/BECR2 BIT 9 FALSELY SET/
	025560	122	062	040	
	025563	102	111	124	
	025566	040	071	040	
	025571	106	101	114	
	025574	123	105	114	
	025577	131	040	123	
2494	025602	105	124	000	
	025605	124	117	040	EM58: .ASCIZ/TO INTERRUPT CPU/
	025610	111	116	124	
	025613	105	122	122	
	025616	125	120	124	
	025621	040	103	120	
2495	025624	125	000		
	025626	105	122	122	EM59: .ASCIZ/ERROR: TEST OF NSSYN ERROR BIT FAILED/
	025631	117	122	072	
	025634	040	124	105	
	025637	123	124	040	
	025642	117	106	040	
	025645	116	123	123	
	025650	131	116	040	
	025653	105	122	122	
	025656	117	122	040	
	025661	102	111	124	
	025664	040	106	101	
	025667	111	114	105	
2496	025672	104	000		
	025674	124	117	040	EM60: .ASCIZ/TO SET BIT 8 OF BECR2/
	025677	123	105	124	
	025702	040	102	111	
	025705	124	040	070	
	025710	040	117	106	
	025713	040	102	105	
	025716	103	122	062	
2497	025721	000			
	025722	124	117	040	EM61: .ASCIZ/TO SET BIT 15 OF BECR1/
	025725	123	105	124	
	025730	040	102	111	
	025733	124	040	061	

CZ
SY
SY
C
CL

CR
CR

CR

D

DB

DD

DH

DH

DH

DH

DH

DH

DH

DH

DH

DH

DH

DI

DI

DI

DS

DT

D1

D1

D1

D1

E

EN

EN

EN

EN

EN

EN

EN

EN

EN

EN

2502	026207	040	127	111	DH65:	.ASCIZ/ WITH BECR1 = /
	026212	124	110	040		
	026215	102	105	103		
	026220	122	061	040		
	026223	075	040	000		
2503	026226	127	111	124	EM66:	.ASCIZ/WITH INT. ON DONE = 1/
	026231	110	040	111		
	026234	116	124	056		
	026237	040	117	116		
	026242	040	104	117		
	026245	116	105	040		
	026250	075	040	061		
	026253	000				
2504	026254	127	122	117	EM67:	.ASCIZ/WRONG GRANT ERROR BIT FALSELY SET/
	026257	116	107	040		
	026262	107	122	101		
	026265	116	124	040		
	026270	105	122	122		
	026273	117	122	040		
	026276	102	111	124		
	026301	040	106	101		
	026304	114	123	105		
	026307	114	131	040		
	026312	123	105	124		
	026315	000				
2505	026316	124	117	040	EM69:	.ASCIZ/TO CLEAR BIT 11 OF BECR1/
	026321	103	114	105		
	026324	101	122	040		
	026327	102	111	124		
	026332	040	061	061		
	026335	040	117	106		
	026340	040	102	105		
	026343	103	122	061		
	026346	000				
2506	026347	105	122	122	EM70:	.ASCIZ/ERROR: TEST OF TIME DELAY AND BUS LATENCY FAILED/
	026352	117	122	072		
	026355	040	124	105		
	026360	123	124	040		
	026363	117	106	040		
	026366	124	111	115		
	026371	105	040	104		
	026374	105	114	101		
	026377	131	040	101		
	026402	116	104	040		
	026405	102	125	123		
	026410	040	114	101		
	026413	124	105	116		
	026416	103	131	040		
	026421	106	101	111		
	026424	114	105	104		
	026427	000				
2507	026430	124	117	040	EM71:	.ASCIZ/TO SET BIT 6 OF BECR2/
	026433	123	105	124		
	026436	040	102	111		
	026441	124	040	066		
	026444	040	117	106		
	026447	040	102	105		

CZ
SY
EM
EM
EM
EM
EM
EM
EM
EM
EM
EM
EM
EM
ER

ER
ER
ER
ER
ER
F
FI
FI
FI
GN

HT
IA
IN

IC
IT
LA
LA
LF
LC
LC
MS

	026715	040	103	114	
	026720	105	101	122	
	026723	040	122	105	
	026726	107	000		
2515	026730	105	122	122	EM80: .ASCIZ/ERROR: DYNAMIC TEST OF UBE FAILED/
	026733	117	122	072	
	026736	040	104	131	
	026741	116	101	115	
	026744	111	103	040	
	026747	124	105	123	
	026752	124	040	117	
	026755	106	040	125	
	026760	102	105	040	
	026763	106	101	111	
	026766	114	105	104	
	026771	000			
2516	026772	124	117	040	EM81: .ASCIZ/TO LOAD PROPER DATA/
	026775	114	117	101	
	027000	104	040	120	
	027003	122	117	120	
	027006	105	122	040	
	027011	104	101	124	
	027014	101	000		
2517	027016	105	122	122	EM82: .ASCIZ/ERROR: TEST OF PASSING GRANTS FAILED/
	027021	117	122	072	
	027024	040	124	105	
	027027	123	124	040	
	027032	117	106	040	
	027035	120	101	123	
	027040	123	111	116	
	027043	107	040	107	
	027046	122	101	116	
	027051	124	123	040	
	027054	106	101	111	
	027057	114	105	104	
	027062	000			
2518	027063	105	122	122	EM83: .ASCIZ/ERROR:FALSE INTERRUPT WHEN DO RELEASE BUS IMMED./
	027066	117	122	072	
	027071	106	101	114	
	027074	123	105	040	
	027077	111	116	124	
	027102	105	122	122	
	027105	125	120	124	
	027110	040	127	110	
	027113	105	116	040	
	027116	104	117	040	
	027121	122	105	114	
	027124	105	101	123	
	027127	105	040	102	
	027132	125	123	040	
	027135	111	115	115	
	027140	105	104	056	
	027143	000			
2519	027144	105	122	122	EM84: .ASCIZ/ERROR:TEST OF MULTIPLE INTERRUPTS FAILED TO SET RDY/
	027147	117	122	072	
	027152	124	105	123	
	027155	124	040	117	

	027160	106	040	115	
	027163	125	114	124	
	027166	111	120	114	
	027171	105	040	111	
	027174	116	124	105	
	027177	122	122	125	
	027202	120	124	123	
	027205	040	106	101	
	027210	111	114	105	
	027213	104	040	124	
	027216	117	040	123	
	027221	105	124	040	
	027224	122	104	131	
	027227	000			
2520	027230	125	102	105	MSG7: .ASCIZ/UBE WITH VECTOR: /
	027233	040	127	111	
	027236	124	110	040	
	027241	126	105	103	
	027244	124	117	122	
	027247	072	040	000	
2521	027252	040	101	116	MSG8: .ASCIZ/ AND BR AT: /
	027255	104	040	102	
	027260	122	040	101	
	027263	124	072	040	
	027266	000			
2522	027267	040	106	101	MSG9: .ASCIZ/ FALSELY INTERRUPTED WHEN/<15><12>
	027272	114	123	105	
	027275	114	131	040	
	027300	111	116	124	
	027303	105	122	122	
	027306	125	120	124	
	027311	105	104	040	
	027314	127	110	105	
	027317	116	015	012	
	027322	000			
2523	027323	040	123	110	MSG10: .ASCIZ/ SHOULD HAVE INTERRUPTED/<15><12>
	027326	117	125	114	
	027331	104	040	110	
	027334	101	126	105	
	027337	040	111	116	
	027342	124	105	122	
	027345	122	125	120	
	027350	124	105	104	
	027353	015	012	000	
2524	027356	015	012	120	MSG11: .ASCIZ<15><12>/PASSING OF GRANTS NOT TESTED WITH ONE EXERCISOR/<15><12>
	027361	101	123	123	
	027364	111	116	107	
	027367	040	117	106	
	027372	040	107	122	
	027375	101	116	124	
	027400	123	040	116	
	027403	117	124	040	
	027406	124	105	123	
	027411	124	105	104	
	027414	040	127	111	
	027417	124	110	040	
	027422	117	116	105	

	027425	040	105	130	
	027430	105	122	103	
	027433	111	123	117	
	027436	122	015	012	
	027441	000			
2525	027442	015	012	111	MSG12: .ASCII<15><12>/IF MORE THAN ONE UBE PRESENT JUMPER W1/<15><12>
	027445	106	040	115	
	027450	117	122	105	
	027453	040	124	110	
	027456	101	116	040	
	027461	117	116	105	
	027464	040	125	102	
	027467	105	040	120	
	027472	122	105	123	
	027475	105	116	124	
	027500	040	112	125	
	027503	115	120	105	
	027506	122	040	127	
	027511	061	015	012	
2526	027514	123	110	117	.ASCIZ/SHOULD BE INSERTED IN ALL UBE EXCEPT LAST/<15><12>
	027517	125	114	104	
	027522	040	102	105	
	027525	040	111	116	
	027530	123	105	122	
	027533	124	105	104	
	027536	040	111	116	
	027541	040	101	114	
	027544	114	040	125	
	027547	102	105	040	
	027552	105	130	103	
	027555	105	120	124	
	027560	040	114	101	
	027563	123	124	015	
	027566	012	000		
2527	027570	015	012	124	MSG13: .ASCIZ<15><12>/TESTING UBE WITH BEDB ADDRESS: /
	027573	105	123	124	
	027576	111	116	107	
	027601	040	125	102	
	027604	105	040	127	
	027607	111	124	110	
	027612	040	102	105	
	027615	104	102	040	
	027620	101	104	104	
	027623	122	105	123	
	027626	123	072	040	
	027631	000			
2528	027632	015	012	040	MSG14: .ASCIZ<15><12>/ NOTE:DISREGARD BIT 13 =1 OF BECR2/<15><12>
	027635	040	040	116	
	027640	117	124	105	
	027643	072	104	111	
	027646	123	122	105	
	027651	107	101	122	
	027654	104	040	102	
	027657	111	124	040	
	027662	061	063	040	
	027665	075	061	040	
	027670	117	106	040	

	027673	102	105	103	
	027676	122	062	015	
	027701	012	000		
2529	027703	015	012	120	MSG15: .ASCIZ<15><12>/PC OF ERROR MESSAGE WAS: /
	027706	103	040	117	
	027711	106	040	105	
	027714	122	122	117	
	027717	122	040	115	
	027722	105	123	123	
	027725	101	107	105	
	027730	040	127	101	
	027733	123	072	040	
	027736	000			
2530	027737	015	012	040	MSG16: .ASCIZ<15><12>/ UNIBUS EXERCISER MODULE DIAGNOSTIC--CZKUB-C/<15><12><15><12>
	027742	040	040	040	
	027745	040	125	116	
	027750	111	102	125	
	027753	123	040	105	
	027756	130	105	122	
	027761	103	111	123	
	027764	105	122	040	
	027767	115	117	104	
	027772	125	114	105	
	027775	040	104	111	
	030000	101	107	116	
	030003	117	123	124	
	030006	111	103	055	
	030011	055	103	132	
	030014	113	125	102	
	030017	055	103	015	
	030022	012	015	012	
	030025	000			
2531	030026	040	040	040	MSG17: .ASCIZ/ TEST NUMBER WAS: /
	030031	040	040	040	
	030034	040	040	040	
	030037	124	105	123	
	030042	124	040	116	
	030045	125	115	102	
	030050	105	122	040	
	030053	127	101	123	
	030056	072	040	000	

2532
2533
2534
2535
2536
2537 030062
2538 000001

```
          .EVEN  
://///////  
:BUFFER WORK AREA  
://///////  
BUFF1: .BLKW 11  
          .END
```

SYMBOL TABLE

A 003172
 ACALC 003376
 B 001126
 BEBA 002522
 BEBD 002516
 BECC 002520
 BECR1 002524
 BECR2 002526
 BEGO 002534
 BERE 002530
 BE1BA 002542
 BE1BD 002536
 BE1CC 002540
 BE1CR1 002544
 BE1CR2 002546
 BE1RE 002550
 BE1VEC 002552
 BE2BA 002560
 BE2BD 002554
 BE2CC 002556
 BE2CR1 002562
 BE2CR2 002564
 BE2RE 002566
 BE2VEC 002570
 BE3BA 002576
 BE3BD 002572
 BE3CC 002574
 BE3CR1 002600
 BE3CR2 002602
 BE3RE 002604
 BE3VEC 002606
 BE4BA 002614
 BE4BD 002610
 BE4CC 002612
 BE4CR1 002616
 BE4CR2 002620
 BE4RE 002622
 BE4VEC 002624
 BIT0 = 000001
 BIT00 = 000001
 BIT01 = 000002
 BIT02 = 000004
 BIT03 = 000010
 BIT04 = 000020
 BIT05 = 000040
 BIT06 = 000100
 BIT07 = 000200
 BIT08 = 000400
 BIT09 = 001000
 BIT1 = 000002
 BIT10 = 002000
 BIT11 = 004000
 BIT12 = 010000
 BIT13 = 020000
 BIT14 = 040000
 BIT15 = 100000
 BIT2 = 000004

BIT3 = 000010
 BIT4 = 000020
 BIT5 = 000040
 BIT6 = 000100
 BIT7 = 000200
 BIT8 = 000400
 BIT9 = 001000
 BPTVEC= 000014
 BUFF1 030062
 C 003500
 CLRREG 016172
 CR = 000015
 CRDY 016246
 CRLF = 000200
 D 003514
 DB = 170000
 DDISP = 177570
 DH15 022205
 DH17 022361
 DH18 022475
 DH19 022571
 DH2 021072
 DH24 023075
 DH3 021173
 DH34 023637
 DH35 023733
 DH36 024030
 DH4 021254
 DH43 024510
 DH46 024702
 DH65 026207
 DH7 021504
 DINT 016276
 DISPLA 001174
 DISPRE 000174
 DSWR = 177570
 DT2 021120
 DT24 023152
 DT3 021202
 DT4 021314
 DT7 021562
 E 003532
 EMAP 002510
 EMTVEC= 000030
 EM1 020750
 EM10 021716
 EM11 021757
 EM12 022023
 EM14 022070
 EM15 022121
 EM16 022227
 EM17 022322
 EM18 022406
 EM19 022510
 EM2 021032
 EM20 022612
 EM21 022661

EM22 022717
 EM23 022762
 EM24 023026
 EM25 023164
 EM26 023224
 EM27 023265
 EM28 023325
 EM29 023354
 EM3 021126
 EM30 023403
 EM31 023433
 EM32 023463
 EM33 023532
 EM34 023567
 EM35 023656
 EM36 023761
 EM37 024040
 EM38 024077
 EM39 024167
 EM4 021206
 EM40 024245
 EM41 024323
 EM42 024365
 EM43 024424
 EM44 024541
 EM45 024605
 EM46 024632
 EM47 024740
 EM49 025016
 EM5 021324
 EM51 025075
 EM52 025126
 EM53 025161
 EM54 025233
 EM56 025476
 EM57 025555
 EM58 025605
 EM59 025626
 EM6 021372
 EM60 025674
 EM61 025722
 EM62 025751
 EM63 026001
 EM64 026031
 EM65 026127
 EM66 026226
 EM67 026254
 EM69 026316
 EM7 021442
 EM70 026347
 EM71 026430
 EM72 026456
 EM73 026506
 EM74 026553
 EM75 026575
 EM76 026615
 EM77 026642

EM78 026670
 EM8 021570
 EM80 026730
 EM81 026772
 EM82 027016
 EM83 027063
 EM84 027144
 EM9 021655
 ERROR = 104000
 ERRVEC= 000004
 ERR1 006620
 ERR2 006630
 ERR3 006640
 ERR4 006650
 F 003570
 FINT1 005172
 FINT3 005600
 FIRST 003632
 HT = 000011
 IADD 003266
 INTVEC 002532
 IOTVEC= 000020
 ITRAP 004770
 LAST 014440
 LAST1 014462
 LF = 000012
 LOOP1 003150
 LOOP2 003156
 MSG1 020414
 MSG10 027323
 MSG11 027356
 MSG12 027442
 MSG13 027570
 MSG14 027632
 MSG15 027703
 MSG16 027737
 MSG17 030026
 MSG2 020425
 MSG3 025331
 MSG4 025457
 MSG5 020663
 MSG7 027230
 MSG8 027252
 MSG9 027267
 MTRAP 003234
 NO 002630
 NUBE 014430
 NUBE1 014434
 PIRQ = 177772
 PIRQVE= 000240
 PRO = 000000
 PR1 = 000040
 PR2 = 000100
 PR3 = 000140
 PR4 = 000200
 PR5 = 000240
 PR6 = 000300

PR7 = 000340
 PS = 177776
 PSW = 177776
 PTRAP 004754
 PWRVEC= 000024
 RCATCH 016224
 RDYS 006570
 RESVEC= 000010
 RSTART 001100
 RTR 006044
 RVEC 016316
 R6 = %000006
 R7 = %000007
 SCOPE = 000004
 SPTR 002514
 STACK = 001100
 START 002632
 START1 003100
 STKLMT= 177774
 STRAP 004030
 SWR 001172
 SWREG 000176
 SW0 = 000001
 SW00 = 000001
 SW01 = 000002
 SW02 = 000004
 SW03 = 000010
 SW04 = 000020
 SW05 = 000040
 SW06 = 000100
 SW07 = 000200
 SW08 = 000400
 SW09 = 001000
 SW1 = 000002
 SW10 = 002000
 SW11 = 004000
 SW12 = 010000
 SW13 = 020000
 SW14 = 040000
 SW15 = 100000
 SW2 = 000004
 SW3 = 000010
 SW4 = 000020
 SW5 = 000040
 SW6 = 000100
 SW7 = 000200
 SW8 = 000400
 SW9 = 001000
 TBITVE= 000014
 TERRPC 016432
 TKVEC = 000060
 TMAP 002512
 TPVEC = 000064
 TRAPVE= 000034
 TRTVEC= 000014
 TSTA 006656
 TSTB 014134

SYMBOL TABLE

TST1	003630	T05L03	004522	T17L11	010732	T28L01	013636	\$DOAGN	016146
TST10	005224	T05L04	004506	T18L01	011020	T28L02	013752	\$DTBL	017430
TST11	005754	T05L05	004652	T18L02	011052	T28L03	013770	\$ENDAD	016136
TST12	006050	T05L06	004620	T18L03	011110	T28L04	014120	\$ENDCT	016104
TST13	006656	T05L07	004562	T18L04	011072	T28L05	013650	\$ENDMG	016155
TST14	007022	T05L08	004540	T19L01	011160	T28L06	014012	\$ENULL	016152
TST15	007152	T06L01	005012	T19L02	011270	T28L08	014046	\$EOP	016050
TST16	007456	T07L03	005116	T19L03	011240	T28L09	014060	\$EOPCT	016076
TST17	007740	T07L04	005162	T19L04	011252	T28L10	013524	\$ERFLG	001135
TST2	004060	T07L05	005220	T19L05	011330	T29L01	014244	\$ERMAX	001147
TST20	010204	T07L06	005210	T19L07	011310	T29L02	014336	\$ERROR	016722
TST21	010332	T07L07	005114	T19L09	011266	T29L03	014374	\$ERRPC	001150
TST22	010472	T07L08	005202	T20L01	011366	T29L04	014240	\$ERRTB	001250
TST23	010776	T08L01	005304	T20L02	011474	T29L05	014344	\$ERRTY	017070
TST24	011134	T08L02	005552	T20L03	011442	T29L06	014310	\$ERTTL	001144
TST25	011342	T08L03	005510	T20L04	011456	T29L07	014422	\$ESCAP	001236
TST26	011550	T08L04	005642	T20L05	011536	T30L01	014516	\$FILLC	001210
TST27	012002	T08L05	005650	T20L06	011516	T30L02	014730	\$FILLS	001207
TST3	004266	T08L06	005736	T20L08	011476	T30L03	014744	\$GDADR	001152
TST30	012116	T08L07	005732	T21L01	011742	T30L04	014760	\$GDDAT	001156
TST31	012372	T08L08	005252	T21L02	011770	T30L05	014774	\$GET42	016126
TST32	012706	T08L09	005536	T21L03	011724	T30L06	015006	\$HD =	000001
TST33	013136	T09L01	014212	T21L04	011776	T30L07	014716	\$JCNT	001136
TST34	013224	T10L01	006206	T22L01	012076	T30L08	015506	\$JILLUP	020376
TST35	013500	T10L02	006326	T23L01	007320	T30L09	015332	\$INTAG	001167
TST36	014134	T10L03	006452	T23L02	007254	T30L10	015422	\$ITEMB	001146
TST37	014216	T10L04	006616	T23L03	007274	T30L11	015410	\$LF	001246
TST4	004364	T10L05	006572	T23L04	007446	T30L12	015676	\$LPADR	001140
TST40	014460	T10L06	006412	T23L05	007344	T30L13	015474	\$LPERR	001142
TST5	004462	T11L01	006744	T23L06	007364	T30L14	015360	\$MXCNT	016720
TST6	004656	T11L02	006764	T23L07	007426	T30L15	015552	\$NULL	001206
TST7	005032	T12L01	007114	T24L01	012230	T30L16	015572	\$NWTST=	000001
TYPDS =	104405	T13L01	010012	T24L02	012220	T30L17	015600	\$OCNT	020162
TYPE =	104401	T13L02	007774	T24L03	012366	T30L20	016022	\$OMODE	020164
TYPOC =	104402	T13L03	010016	T24L04	012334	T30L21	014636	\$OVER	016704
TYPON =	104404	T13L04	010154	T24L05	012266	T30L22	015026	\$PASS	001132
TYPOS =	104403	T13L05	010176	T24L06	012250	T30L25	015202	\$POWER	020404
T01L01	003676	T14L01	007670	T25L01	012512	T30L26	015100	\$PWRDN	020236
T01L02	003754	T14L02	007504	T25L02	012542	T30L27	015250	\$PWRMG	020372
T01L03	003712	T14L03	007712	T25L03	012644	T30L28	015044	\$PWRUP	020310
T01L04	003760	T14L04	007734	T25L04	012664	T30L29	015120	\$QUES	001244
T01L05	004042	T15L01	010250	T25L05	012420	T30L30	015072	\$REGAD	001212
T01L06	003746	T15L02	010326	T25L06	012560	T31L01	013220	\$REG0	001214
T02L01	004214	T15L03	010234	T25L07	012622	UCNT	002626	\$REG1	001216
T02L02	004144	T16L01	010414	T25L08	012702	WINT	005716	\$REG2	001220
T02L03	004122	T16L02	010452	T26L01	013044	\$AUTOB	001166	\$REG3	001222
T02L04	004204	T16L03	010460	T26L02	012752	\$BDADR	001154	\$RTNAD	016150
T03L01	004346	T17L01	010516	T26L03	013002	\$BDDAT	001160	\$SAVR6	020402
T03L02	004356	T17L02	010612	T26L04	013022	\$BELL	001240	\$SCOPE	016472
T03L03	004312	T17L03	010572	T26L05	013070	\$CHARC	017734	\$SETUP=	000037
T03L04	004306	T17L04	010710	T27L01	013324	\$CMTAG	001132	\$STUP =	177777
T04L01	004444	T17L05	010616	T27L02	013444	\$CM1 =	000004	\$SVLAD	016656
T04L02	004454	T17L06	010630	T27L03	013370	\$CM2 =	000010	\$SVPC =	001132
T04L03	004410	T17L07	010754	T27L04	013340	\$CM3 =	000004	\$SWR =	167000
T04L04	004404	T17L08	010764	T27L05	013432	\$CM4 =	000004	\$SWRMK=	000000
T05L01	004572	T17L09	010772	T27L06	013402	\$CRLF	001245	\$TIMES	001234
T05L02	004624	T17L10	010712	T27L07	013464	\$DBLK	017440	\$TXB	001200

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
A		003172	#61-511 61-524
ACALC		003376	#61-545 61-557 61-565 69-2075
B		001126	59-25 #59-27
BEBA		002522	#61-455 *61-533 *61-547 62-714 62-742 65-1037 65-1054 65-1072 65-1091 65-1137 65-1161 65-1193 65-1225 65-1253 65-1259 65-1269 65-1273 65-1275 65-1280 65-1325 65-1331 65-1338 65-1384 65-1394 65-1421 65-1484 65-1523 65-1573 65-1619 65-1631 66-1692 66-1712 66-1715 66-1716 66-1748 66-1841 66-1936 66-1943 66-1957 66-1958 69-2039 69-2319
BEBD		002516	#61-453 *61-531 *61-545 61-569 61-575 61-600 62-699 62-727 62-760 62-761 65-1042 65-1044 65-1055 65-1061 65-1076 65-1080 65-1090 65-1098 65-1145 65-1149 65-1165 65-1169 65-1323 65-1329 65-1336 65-1390 65-1420
BECC		002520	65-1525 65-1618 69-2321 #61-454 *61-532 *61-546 65-1036 65-1053 65-1071 65-1092 65-1136 65-1162 65-1195 65-1227 65-1250 65-1261 65-1270 65-1320 65-1326 65-1333 65-1364 65-1366 65-1385 65-1392 65-1422 65-1485 65-1524 65-1574 65-1620 65-1632 65-1666 66-1694 66-1714 66-1747 66-1840 66-1852 66-1934 66-1944 66-1953 66-1955 69-2040 69-2320
BECR1		002524	#61-456 *61-534 *61-548 61-602 61-618 61-651 62-768 62-775 62-779 62-789 62-852 62-853 62-860 62-862 62-922 62-928 62-933 62-938 62-949 62-956 62-958 62-959 62-978 62-1006 65-1039 65-1057 65-1074 65-1093 65-1109 65-1138 65-1163 65-1196 65-1203 65-1220 65-1229 65-1231 65-1251 65-1262 65-1271 65-1321 65-1327 65-1334 65-1357 65-1371 65-1388 65-1424 65-1442 65-1446 65-1486 65-1487 65-1526 65-1575 65-1621 65-1633 65-1646 65-1667 65-1670 65-1674 66-1696 66-1722 66-1744 66-1749 66-1752 66-1762 66-1767 66-1774 66-1780 66-1786 66-1791 66-1810 66-1820 66-1842 66-1854 66-1938 66-1945 66-1963 66-1965 66-1973 66-2009 69-2041 69-2318 69-2342
BECR2		002526	#61-457 *61-535 *61-549 61-598 61-609 61-611 61-620 61-622 61-623 62-667 62-669 62-684 62-686 62-777 62-781 62-787 62-791 62-823 62-824 62-836 62-966 62-968 62-990 62-1005 64-1016 65-1194 65-1198 65-1210 65-1215 65-1226 65-1260 65-1264 65-1285 65-1332 65-1387 65-1401 65-1444 65-1447 66-1693 66-1698 66-1706 66-1713 66-1718 66-1720 66-1721 66-1724 66-1754 66-1756 66-1760 66-1769 66-1771 66-1808 66-1815 66-1833 66-1853 66-1880 66-1916 66-1923 66-1935 66-1947 66-1952 66-1960 66-1961 66-1972 66-1974 66-1979 66-1984 66-1985 66-1987 66-1988 69-2317 #61-460 *61-537 61-577 65-1331 66-1692 66-2010 69-2140 69-2227
BEGO		002534	#61-460 *61-537 61-577 65-1331 66-1692 66-2010 69-2140 69-2227
BERE		002530	#61-458 *61-536 *61-550 61-599 61-621 61-652 65-1214 65-1237 66-1832 66-1844 69-2316
BE1BA		002542	#61-463
BE1BD		002536	#61-461 61-539 61-543 69-2143
BE1CC		002540	#61-462
BE1CR1		002544	#61-464 69-2132 69-2144 69-2272 69-2281
BE1CR2		002546	#61-465
BE1RE		002550	#61-466
BE1VEC		002552	#61-467 69-2113 69-2114
BE2BA		002560	#61-470
BE2BD		002554	#61-468 69-2102 69-2146
BE2CC		002556	#61-469
BE2CR1		002562	#61-471 69-2133 69-2147 69-2273 69-2283
BE2CR2		002564	#61-472
BE2RE		002566	#61-473
BE2VEC		002570	#61-474 69-2116 69-2117

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES								
BE3BA		002576	#61-477								
BE3BD		002572	#61-475	69-2149							
BE3CC		002574	#61-476								
BE3CR1		002600	#61-478	69-2134	69-2136	69-2150	69-2274	69-2276	69-2285	69-2287	
BE3CR2		002602	#61-479								
BE3RE		002604	#61-480								
BE3VEC		002606	#61-481	69-2119	69-2121	69-2122					
BE4BA		002614	#61-484								
BE4BD		002610	#61-482	69-2152							
BE4CC		002612	#61-483								
BE4CR1		002616	#61-485	69-2137	69-2139	69-2153	69-2277	69-2279	69-2289	69-2291	
BE4CR2		002620	#61-486								
BE4RE		002622	#61-487								
BE4VEC		002624	#61-488	69-2124	69-2126	69-2127					
BIT0	=	000001	#59-17								
BIT00	=	000001	#59-17	59-17							
BIT01	=	000002	#59-17	59-17							
BIT02	=	000004	#59-17	59-17							
BIT03	=	000010	#59-17	59-17							
BIT04	=	000020	#59-17	59-17							
BIT05	=	000040	#59-17	59-17							
BIT06	=	000100	#59-17	59-17							
BIT07	=	000200	#59-17	59-17							
BIT08	=	000400	#59-17	59-17							
BIT09	=	001000	#59-17	59-17	69-2398	69-2399					
BIT1	=	000002	#59-17								
BIT10	=	002000	#59-17	69-2399							
BIT11	=	004000	#59-17	69-2398							
BIT12	=	010000	#59-17								
BIT13	=	020000	#59-17	69-2399							
BIT14	=	040000	#59-17	69-2398							
BIT15	=	100000	#59-17								
BIT2	=	000004	#59-17								
BIT3	=	000010	#59-17								
BIT4	=	000020	#59-17								
BIT5	=	000040	#59-17								
BIT6	=	000100	#59-17								
BIT7	=	000200	#59-17								
BIT8	=	000400	#59-17								
BIT9	=	001000	#59-17								
BPTVEC	=	000014	#59-17								
BUFF1		030062	*65-1034	65-1037	65-1045	65-1046	*65-1052	65-1054	65-1059	65-1062	65-1063
			*65-1070	65-1072	65-1078	65-1081	65-1082	*65-1088	65-1091	65-1095	65-1099
			65-1100	*65-1135	65-1137	65-1147	65-1150	65-1151	*65-1160	65-1161	65-1167
			65-1170	65-1171	65-1269	65-1273	65-1276	65-1384	*65-1386	65-1394	65-1416
			65-1418	65-1421	65-1433	65-1436	65-1440	65-1454	65-1459	65-1480	65-1482
			65-1484	65-1489	65-1494	65-1519	65-1521	65-1523	65-1530	65-1533	65-1537
			65-1542	65-1547	65-1569	65-1571	65-1573	65-1579	65-1582	65-1586	65-1591
			65-1596	*65-1617	65-1619	65-1626	*65-1628	*65-1629	65-1631	65-1639	66-1748
			66-1841	69-2034	69-2036	69-2039	69-2043	*69-2043	69-2052	69-2056	69-2129
			*69-2161	69-2170	69-2173	69-2221	69-2222	69-2235	69-2237	69-2239	69-2240
			69-2242	69-2243	#69-2537						

SYMBOL	CROSS REFERENCE VALUE	REFERENCES
EM20	022612	61-126 #69-2447
EM21	022661	61-131 #69-2449
EM22	022717	61-136 #69-2450
EM23	022762	61-141 #69-2451
EM24	023026	61-146 #69-2452
EM25	023164	61-151 #69-2456
EM26	023224	61-156 #69-2457
EM27	023265	61-161 #69-2458
EM28	023325	61-166 #69-2459
EM29	023354	61-171 #69-2460
EM3	021126	61-41 #69-2418
EM30	023403	61-176 #69-2461
EM31	023433	61-181 #69-2462
EM32	023463	61-186 #69-2463
EM33	023532	61-191 #69-2464
EM34	023567	61-196 #69-2465
EM35	023656	61-201 #69-2467
EM36	023761	61-206 #69-2469
EM37	024040	61-211 #69-2471
EM38	024077	61-216 #69-2472
EM39	024167	61-221 #69-2473
EM4	021206	61-46 #69-2422
EM40	024245	61-226 #69-2474
EM41	024323	61-231 #69-2475
EM42	024365	61-236 #69-2476
EM43	024424	61-241 61-301 #69-2477
EM44	024541	61-246 #69-2479
EM45	024605	61-251 #69-2480
EM46	024632	61-256 #69-2481
EM47	024740	61-261 61-266 #69-2483
EM49	025016	61-271 61-276 #69-2484
EM5	021324	61-51 #69-2426
EM51	025075	61-281 #69-2485
EM52	025126	61-286 #69-2486
EM53	025161	61-291 #69-2487
EM54	025233	61-296 #69-2488
EM56	025476	61-306 #69-2492
EM57	025555	61-311 #69-2493
EM58	025605	61-316 #69-2494
EM59	025626	61-321 #69-2495
EM6	021372	61-56 #69-2427
EM60	025674	61-326 #69-2496
EM61	025722	61-331 #69-2497
EM62	025751	61-336 #69-2498
EM63	026001	61-341 #69-2499
EM64	026031	61-346 #69-2500
EM65	026127	61-351 #69-2501
EM66	026226	61-356 #69-2503
EM67	026254	61-361 #69-2504
EM69	026316	61-371 #69-2505
EM7	021442	61-61 #69-2428
EM70	026347	61-376 #69-2506

SYMBOL	VALUE	REFERENCES								
EM71	026430	61-381	#69-2507							
EM72	026456	61-386	#69-2508							
EM73	026506	61-391	#69-2509							
EM74	026553	61-396	#69-2510							
EM75	026575	61-401	#69-2511							
EM76	026615	61-406	#69-2512							
EM77	026642	61-411	#69-2513							
EM78	026670	61-416	#69-2514							
EM8	021570	61-66	#69-2432							
EM80	026730	61-426	#69-2515							
EM81	026772	61-431	#69-2516							
EM82	027016	61-436	#69-2517							
EM83	027063	61-441	#69-2518							
EM84	027144	61-446	#69-2519							
EM9	021655	61-71	#69-2433							
ERROR	= 104000	#59-17	61-525	61-608	61-630	62-666	62-707	62-735	62-772	62-825
		62-865	62-868	62-872	62-876	62-952	62-964	62-969	62-981	62-987
		62-992	64-1018	65-1048	65-1065	65-1084	65-1102	65-1117	65-1120	65-1123
		65-1126	65-1142	65-1153	65-1173	65-1200	65-1201	65-1205	65-1206	65-1211
		65-1212	65-1217	65-1218	65-1222	65-1223	65-1233	65-1234	65-1277	65-1282
		65-1288	65-1340	65-1368	65-1396	65-1399	65-1430	65-1449	65-1456	65-1461
		65-1464	65-1467	65-1502	65-1544	65-1549	65-1593	65-1598	65-1636	65-1641
		65-1644	65-1650	65-1672	66-1700	66-1701	66-1707	66-1708	66-1726	66-1727
		66-1759	66-1761	66-1763	66-1773	66-1775	66-1776	66-1779	66-1781	66-1782
		66-1785	66-1787	66-1790	66-1792	66-1817	66-1818	66-1822	66-1823	66-1828
		66-1829	66-1835	66-1836	66-1858	66-1882	66-1883	66-1896	66-1897	66-1909
		66-1910	66-1918	66-1919	66-1968	66-1990	66-2013	69-2047	69-2048	69-2065
		69-2066	69-2180	69-2258						
ERRVEC	= 000004	#59-17	61-492	61-492	61-492	69-2398	69-2398	69-2398	69-2398	
ERR1	006620	65-1038	#65-1117							
ERR2	006630	65-1056	#65-1120							
ERR3	006640	65-1073	#65-1123							
ERR4	006650	65-1089	#65-1126							
F	003570	#61-576	61-578							
FINT1	005172	62-849	#62-872							
FINT3	005600	62-919	#62-962							
FIRST	003632	61-581	61-582	#61-594						
GNS	= *****	59-20	59-20	69-2404	69-2404	69-2404	69-2404	69-2404	69-2404	69-2404
		69-2404	69-2404	69-2404						
HT	= 000011	#59-17	69-2402	69-2402						
IADD	003266	61-518	#61-531							
INTVEC	002532	#61-459	*61-538	*61-551	62-849	62-850	62-919	62-941	62-973	65-1192
		65-1228	65-1356	65-1423	65-1630	66-1695	66-1745	66-1766	66-1809	66-2008
		69-2038	69-2327	69-2329	69-2352	69-2354				
IOTVEC	= 000020	#59-17	61-492	61-492						
ITRAP	004770	62-816	#62-830							
LAST	014440	61-558	61-561	#69-2077						
LAST1	014462	69-2077	69-2078	#69-2102						
LF	= 000012	#59-17	69-2402	69-2402						
LOOP1	003150	#61-505	61-510							
LOOP2	003156	#61-507	61-522							
MSG1	020414	61-515	#69-2407							

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES										
MSG10		027323	69-2189	69-2270	#69-2523								
MSG11		027356	69-2106	#69-2524									
MSG12		027442	61-496	#69-2525									
MSG13		027570	61-568	#69-2527									
MSG14		027632	61-613	62-671	#69-2528								
MSG15		027703	69-2389	#69-2529									
MSG16		027737	61-495	#69-2530									
MSG17		030026	69-2391	#69-2531									
MSG2		020425	69-2305	#69-2408									
MSG3		025331	65-1664	#69-2489									
MSG4		025457	66-1678	#69-2491									
MSG5		020663	62-833	#69-2412									
MSG7		027230	69-2184	69-2261	69-2266	#69-2520							
MSG8		027252	69-2263	69-2268	#69-2521								
MSG9		027267	69-2265	#69-2522									
MTRAP		003234	61-501	#61-520									
NO		002630	#61-490	61-541	*61-543	61-574	*61-579						
NUBE		014430	62-673	62-709	62-737	62-774	62-867	62-879	#69-2074				
NUBE1		014434	61-635	#69-2075									
PIRQ	=	177772	#59-17										
PIRQVE	=	000240	#59-17										
PRO	=	000000	#59-17										
PR1	=	000040	#59-17										
PR2	=	000100	#59-17										
PR3	=	000140	#59-17										
PR4	=	000200	#59-17										
PR5	=	000240	#59-17										
PR6	=	000300	#59-17										
PR7	=	000340	#59-17										
PS	=	177776	#59-17	59-17									
PSW	=	177776	#59-17	61-595	61-648	62-698	62-726	62-758	62-820	62-847	62-855		
				62-904	62-923	62-927	62-932	62-942	62-948	62-977	62-1003	63-1008	
				65-1033	65-1040	65-1134	65-1159	65-1190	65-1208	65-1248	65-1303	65-1353	
				65-1358	65-1370	65-1372	65-1382	65-1389	65-1414	65-1425	65-1479	65-1517	
				65-1527	65-1529	65-1551	65-1567	65-1576	65-1578	65-1600	65-1615	65-1634	
				65-1647	65-1668	66-1690	66-1703	66-1746	66-1750	66-1806	66-1825	66-1851	
				66-1874	66-1939	66-1971	66-1976	66-2006	66-2011	69-2033	69-2131	69-2141	
				69-2158	69-2177	69-2179	69-2206	69-2209	69-2228	69-2280			
PTRAP		004754	62-821	#62-827									
PWRVEC	=	000024	#59-17	61-492	61-492	69-2405	69-2405	69-2405	69-2405	69-2405	69-2405	69-2405	
RCATCH		016224	62-864	62-878	62-880	62-980	65-1236	65-1290	65-1342	65-1373	65-1469		
			65-1652	66-1729	66-1794	66-1845	66-1860	66-1992	66-2015	69-2068	#69-2326		
			65-1041	65-1058	65-1075	65-1094	#65-1108						
RDYS		006570	#59-17										
RESVEC	=	000010	#59-17										
RSTART		001100	#59-22										
RTR		006044	64-1017	#65-1021									
RVEC		016316	59-26	62-951	62-963	62-986	62-989	#69-2360					
R6	=	%000006	#59-17	59-24	*61-492	*61-492	61-492						
R7	=	%000007	#59-17										
SCOPE	=	000004	#59-17	61-593	61-646	62-696	62-724	62-756	62-807	62-845	62-902		
				62-1001	65-1031	65-1131	65-1156	65-1188	65-1244	65-1299	65-1351	65-1380	
				65-1412	65-1476	65-1515	65-1565	65-1613	65-1659	66-1688	66-1741	66-1804	

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
T01L05		004042	61-615 #61-633
T01L06		003746	61-610 61-612 #61-614
T02L01		004214	62-662 #62-674
T02L02		004144	#62-663 62-681
T02L03		004122	#61-654 62-687
T02L04		004204	62-668 62-670 #62-672
T03L01		004346	62-703 #62-710
T03L02		004356	62-711 #62-714
T03L03		004312	#62-701 62-713
T03L04		004306	#62-700 62-715
T04L01		004444	62-731 #62-738
T04L02		004454	62-739 #62-742
T04L03		004410	#62-729 62-741
T04L04		004404	#62-728 62-743
T05L01		004572	62-765 62-767 #62-777
T05L02		004624	62-778 #62-786
T05L03		004522	#62-764 62-783 62-785
T05L04		004506	#62-761 62-792 62-794
T05L05		004652	62-790 #62-793
T05L06		004620	62-780 #62-784
T05L07		004562	62-769 #62-775
T05L08		004540	#62-770 62-776
T06L01		005012	62-829 62-831 #62-834
T07L03		005116	#62-857 62-861
T07L04		005162	62-859 #62-868
T07L05		005220	62-863 #62-880
T07L06		005210	62-870 #62-878
T07L07		005114	#62-856 62-874
T07L08		005202	62-854 #62-876
T08L01		005304	#62-915 62-918
T08L02		005552	62-941 #62-955
T08L03		005510	#62-945 62-960
T08L04		005642	62-967 #62-973
T08L05		005650	62-973 #62-974
T08L06		005736	62-954 #62-990
T08L07		005732	62-957 #62-989
T08L08		005252	#62-908 62-910
T08L09		005536	62-944 #62-951
T09L01		014212	66-2008 #66-2015
T10L01		006206	65-1043 #65-1051
T10L02		006326	65-1060 #65-1069
T10L03		006452	65-1079 #65-1088
T10L04		006616	65-1110 #65-1116
T10L05		006572	#65-1109 65-1113
T10L06		006412	65-1077 #65-1080
T11L01		006744	65-1141 #65-1145
T11L02		006764	65-1146 #65-1149
T12L01		007114	65-1166 #65-1169
T13L01		010012	65-1307 #65-1314
T13L02		007774	#65-1309 65-1312
T13L03		010016	65-1313 #65-1316
T13L04		010154	65-1324 65-1330 #65-1338

SYMBOL	VALUE	REFERENCES							
T13L05	010176	65-1337	#65-1342						
T14L01	007670	65-1254	#65-1280						
T14L02	007504	#65-1250	65-1258						
T14L03	007712	65-1265	#65-1285						
T14L04	007734	65-1274	65-1279	65-1284	#65-1290				
T15L01	010250	65-1356	#65-1360						
T15L02	010326	65-1363	#65-1373						
T15L03	010234	#65-1357	65-1365						
T16L01	010414	#65-1390	65-1391						
T16L02	010452	65-1393	#65-1399						
T16L03	010460	65-1395	65-1398	#65-1401					
T17L01	010516	#65-1417	65-1419						
T17L02	010612	65-1423	#65-1433						
T17L03	010572	#65-1427	65-1429						
T17L04	010710	65-1435	#65-1452						
T17L05	010616	#65-1434	65-1437						
T17L06	010630	#65-1438	65-1441						
T17L07	010754	65-1443	#65-1464						
T17L08	010764	65-1445	#65-1467						
T17L09	010772	65-1432	65-1448	65-1451	65-1458	65-1463	65-1466	#65-1469	
T17L10	010712	65-1439	#65-1453						
T17L11	010732	65-1455	#65-1459						
T18L01	011020	#65-1481	65-1483						
T18L02	011052	#65-1487	65-1488						
T18L03	011110	65-1492	#65-1498						
T18L04	011072	#65-1491	65-1495						
T19L01	011160	#65-1520	65-1522						
T19L02	011270	65-1536	#65-1541						
T19L03	011240	#65-1531	65-1534						
T19L04	011252	#65-1535	65-1538						
T19L05	011330	65-1539	65-1546	#65-1551					
T19L07	011310	65-1543	#65-1547						
T19L09	011266	65-1532	#65-1540						
T20L01	011366	#65-1570	65-1572						
T20L02	011474	65-1581	#65-1589						
T20L03	011442	#65-1580	65-1583						
T20L04	011456	#65-1584	65-1587						
T20L05	011536	65-1588	65-1595	#65-1600					
T20L06	011516	65-1592	#65-1596						
T20L08	011476	65-1585	#65-1590						
T21L01	011742	65-1625	#65-1644						
T21L02	011770	65-1627	#65-1650						
T21L03	011724	65-1630	#65-1639						
T21L04	011776	65-1638	65-1640	65-1643	65-1649	#65-1652			
T22L01	012076	65-1671	#65-1674						
T23L01	007320	65-1192	#65-1214						
T23L02	007254	65-1199	#65-1203						
T23L03	007274	65-1204	#65-1208						
T23L04	007446	65-1232	#65-1236						
T23L05	007344	65-1216	#65-1220						
T23L06	007364	65-1221	#65-1225						
T23L07	007426	65-1228	#65-1231						

SYMBOL	VALUE	REFERENCES				
T24L01	012230	66-1695	#66-1706			
T24L02	012220	66-1699	#66-1703			
T24L03	012366	66-1710	66-1719	#66-1729		
T24L04	012334	66-1717	#66-1722			
T24L05	012266	#66-1714	66-1725			
T24L06	012250	66-1705	#66-1711			
T25L01	012512	66-1745	#66-1759			
T25L02	012542	66-1753	#66-1766			
T25L03	012644	66-1755	#66-1785			
T25L04	012664	66-1757	#66-1790			
T25L05	012420	#66-1746	66-1758			
T25L06	012560	66-1766	#66-1769			
T25L07	012622	66-1770	#66-1779			
T25L08	012702	66-1765	66-1772	66-1778	66-1784	66-1789 #66-1794
T26L01	013044	66-1809	#66-1832			
T26L02	012752	#66-1812	66-1814			
T26L03	013002	66-1816	#66-1820			
T26L04	013022	66-1821	#66-1825			
T26L05	013070	66-1831	66-1834	#66-1838		
T27L01	013324	66-1878	#66-1886			
T27L02	013444	66-1885	66-1899	66-1912	66-1914	#66-1916
T27L03	013370	66-1887	#66-1900			
T27L04	013340	#66-1890	66-1892	66-1895		
T27L05	013432	66-1900	#66-1913			
T27L06	013402	#66-1903	66-1905	66-1908		
T27L07	013464	66-1917	#66-1921			
T28L01	013636	66-1946	#66-1949			
T28L02	013752	66-1956	66-1959	66-1962	#66-1967	
T28L03	013770	66-1966	#66-1971			
T28L04	014120	66-1970	66-1986	#66-1992		
T28L05	013650	66-1950	#66-1952			
T28L06	014012	#66-1974	66-1975			
T28L08	014046	66-1978	#66-1981			
T28L09	014060	66-1982	#66-1984			
T28L10	013524	66-1932	#66-1934			
T29L01	014244	#69-2035	69-2037			
T29L02	014336	69-2038	#69-2051			
T29L03	014374	69-2054	#69-2062			
T29L04	014240	#69-2034	69-2060			
T29L05	014344	#69-2053	69-2057			
T29L06	014310	#69-2043	69-2046			
T29L07	014422	69-2050	69-2061	#69-2068		
T30L01	014516	69-2103	#69-2112			
T30L02	014730	69-2113	#69-2143			
T30L03	014744	69-2116	#69-2146			
T30L04	014760	69-2121	#69-2149			
T30L05	014774	69-2126	#69-2152			
T30L06	015006	69-2145	69-2148	69-2151	#69-2154	
T30L07	014716	69-2135	69-2138	#69-2140		
T30L08	015506	69-2217	69-2225	#69-2245		
T30L09	015332	#69-2217	69-2220			
T30L10	015422	69-2226	#69-2230			

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
T30L11		015410	#69-2227 69-2234
T30L12		015676	69-2190 69-2236 69-2260 #69-2272
T30L13		015474	69-2238 #69-2242
T30L14		015360	#69-2223 69-2241 69-2244
T30L15		015552	69-2250 #69-2253
T30L16		015572	69-2254 #69-2257
T30L17		015600	69-2252 69-2256 #69-2258
T30L20		016022	#69-2298 69-2302
T30L21		014636	69-2120 69-2125 #69-2129
T30L22		015026	69-2157 #69-2160
T30L25		015202	69-2174 #69-2191
T30L26		015100	#69-2176 69-2194 69-2204
T30L27		015250	69-2196 #69-2201
T30L28		015044	#69-2170 69-2212
T30L29		015120	#69-2180 69-2197
T30L30		015072	#69-2175 69-2199
T31L01		013220	66-1857 #66-1860
UCNT		002626	#61-489 *61-567 69-2156 69-2167
WINT		005716	62-915 #62-985
\$AUTOB		001166	#60-29
\$BDADR		001154	#60-29
\$BDDAT		001160	#60-29
\$BELL		001240	#60-29 69-2399 69-2399 69-2399
\$CHARC		017734	*69-2402 *69-2402 69-2402 *69-2402 #69-2402
\$CKSWR	=	*****	69-2404
\$CMTAG		001132	#60-29 61-492 61-492 61-492 61-492 61-492 61-492
\$CM1	=	000004	#60-29 60-29 60-29 #60-29 60-29 60-29 #60-29 60-29 60-29
\$CM2	=	000010	#60-29 60-29 60-29 #60-29 60-29 60-29 #60-29 60-29 60-29
\$CM3	=	000004	#60-29 60-29 60-29
\$CM4	=	000004	#60-29 60-29 60-29 #60-29 60-29 60-29 #60-29 60-29 60-29
\$CRLF		001245	#60-29 61-517 61-570 69-2399 69-2399 69-2399 69-2400 69-2400 69-2400
\$DBLK		017440	69-2400 69-2402 69-2402 69-2402
\$DOAGN		016146	69-2401 69-2401 #69-2401
\$DTBL		017430	69-2312 69-2312 #69-2312
\$ENDAD		016136	69-2401 #69-2401
\$ENDCT		016104	59-28 #69-2312 69-2399
\$ENDMG		016155	61-492 #69-2312
\$ENULL		016152	69-2312 #69-2312
\$EOP		016050	69-2312 #69-2312
\$EOPCT		016076	61-527 69-2108 69-2304 #69-2312
\$ERFLG		001135	*61-492 #69-2312 69-2312 69-2312
\$ERMAX		001147	#60-29 69-2398 69-2398 69-2398 *69-2398 69-2398 69-2398 *69-2399 69-2399
\$ERROR		016722	69-2399 69-2398 69-2398
\$ERRPC		001150	61-492 #69-2399
\$ERRTB		001250	#60-29 69-2390 *69-2399 *69-2399 69-2399 69-2399 69-2399 69-2400
\$ERRTY		017070	#61-29 69-2400
\$ERTTL		001144	69-2399 #69-2400 69-2399 69-2399

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES								
\$ESCAP		001236	#60-29	*61-492	*69-2398	69-2399	69-2399	69-2399	69-2399		
\$FILLC		001210	#60-29	69-2402	69-2402	69-2402					
\$FILLS		001207	#60-29	69-2402	69-2402						
\$GDADR		001152	#60-29								
\$GDDAT		001156	#60-29								
\$GET42		016126	#69-2312								
\$GTSWR	=	*****	69-2404								
\$HD	=	000001	59-13	59-13	59-13						
\$ICNT		001136	#60-29	*69-2398	69-2398	*69-2398	69-2398	69-2398			
\$ILLUP		020376	69-2405	69-2405	#69-2405						
\$INTAG		001167	#60-29								
\$ITEMB		001146	#60-29	*69-2399	69-2399	69-2399	69-2400				
\$LF		001246	#60-29	69-2399	69-2399	69-2402	69-2402				
\$LPADR		001140	#60-29	*61-492	*61-581	*69-2077	*69-2398	*69-2398	69-2398	69-2398	69-2398
\$LPERR		001142	#60-29	*61-492	*61-500	*61-582	*69-2078	69-2398	*69-2398	69-2398	69-2398
			69-2399								
\$MAIL	=	*****	61-492	69-2398	69-2399	69-2402					
\$MXCNT		016720	69-2398	69-2398	69-2398	#69-2398					
\$NULL		001206	#60-29	69-2402	69-2402	69-2402					
\$NWTST	=	000001	#61-593	61-593	#61-593	61-593	#61-646	61-646	#61-646	61-646	#62-696
			62-696	#62-696	62-696	#62-724	62-724	#62-724	62-724	#62-756	62-756
			#62-756	62-756	#62-807	62-807	#62-807	62-807	#62-845	62-845	#62-845
			62-845	#62-902	62-902	#62-902	62-902	#62-1001	62-1001	#62-1001	62-1001
			#65-1031	65-1031	#65-1031	65-1031	#65-1131	65-1131	#65-1131	#65-1156	65-1156
			#65-1156	#65-1188	65-1188	#65-1188	65-1188	#65-1244	65-1244	#65-1244	65-1244
			#65-1299	65-1299	#65-1299	65-1299	#65-1351	65-1351	#65-1351	65-1351	#65-1380
			65-1380	#65-1380	65-1380	#65-1412	65-1412	#65-1412	65-1412	#65-1476	65-1476
			#65-1476	65-1476	#65-1515	65-1515	#65-1515	65-1515	#65-1565	65-1565	#65-1565
			65-1565	#65-1613	65-1613	#65-1613	65-1613	#65-1659	65-1659	#65-1659	65-1659
			#66-1688	66-1688	#66-1688	66-1688	#66-1741	66-1741	#66-1741	66-1741	#66-1804
			66-1804	#66-1804	66-1804	#66-1847	66-1847	#66-1847	#66-1871	66-1871	#66-1871
			66-1871	#66-1930	66-1930	#66-1930	66-1930	#66-2004	66-2004	#66-2004	66-2004
			#68-2029	68-2029	#68-2029	68-2029	#69-2101	69-2101	#69-2101	69-2101	
\$OCNT		020162	*69-2403	*69-2403	#69-2403						
\$OMODE		020164	*69-2403	*69-2403	69-2403	*69-2403	*69-2403	#69-2403			
\$OVER		016704	69-2398	69-2398	69-2398	#69-2398					
\$PASS		001132	#60-29	69-2303	*69-2312	*69-2312	69-2312	69-2312	69-2312	69-2398	69-2398
			69-2398								
\$POWER		020404	69-2405	#69-2405							
\$PWRDN		020236	61-492	#69-2405	69-2405						
\$PWRMG		020372	#69-2405								
\$PWRUP		020310	69-2405	#69-2405							
\$QUES		001244	#60-29	69-2399	69-2399	69-2402	69-2402				
\$RDCHR	=	*****	69-2404								
\$RDDEC	=	*****	69-2404								
\$RDLIN	=	*****	69-2404								
\$RDOCT	=	*****	69-2404								
\$REGAD		001212	#60-29								
\$REGO		001214	#60-29	*61-606	*61-629	*62-663	*62-704	*62-732	*62-770	*62-920	*62-925
				*62-930	*62-939	*62-945	*62-968	*62-976	*65-1044	*65-1061	*65-1080
				*65-1149	*65-1169	*65-1210	*65-1275	*65-1280	*65-1287	*65-1338	*65-1339
				*65-1459	*65-1499	*65-1547	*65-1596	*66-1706	*66-1760	*66-1762	*66-1774

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
\$REG1	001216	*66-1786	*66-1791	*66-1953	*66-1957	*66-1960	*66-1963	66-1967	*66-1987	*69-2062
		*69-2183	69-2185	*69-2246	69-2262	69-2417	69-2421	69-2425	69-2431	69-2455
		#60-29	*61-607	*62-664	*62-706	*62-734	*62-771	*62-921	*62-926	*62-931
		*62-940	*62-946	*62-975	62-977	*65-1045	*65-1062	*65-1081	*65-1099	*65-1150
		*65-1170	*65-1276	*65-1281	*65-1367	*65-1460	*65-1500	*65-1548	*65-1597	*66-1967
		*66-1988	*69-2063	*69-2186	69-2188	*69-2247	69-2264	69-2417	69-2425	69-2431
		69-2455								
\$REG2	001220	#60-29	*62-665	*62-705	*62-733	*65-1046	*65-1063	*65-1082	*65-1100	*65-1151
		*65-1171	*65-1501	*66-1954	*66-1964	*66-1989	*69-2064	*69-2248	69-2267	69-2425
		69-2455								
\$REG3	001222	#60-29	*65-1047	*65-1064	*65-1083	*65-1101	*65-1152	*65-1172	*69-2251	*69-2255
		*69-2257	69-2269	69-2455						
\$RTNAD	016150	#69-2312								
\$R2A	= *****	69-2404								
\$SAVRE	= *****	69-2404								
\$SAVR6	020402	*69-2405	69-2405	*69-2405	*69-2405	#69-2405				
\$SCOPE	016472	61-492	#69-2398							
\$SETUP	= 000037	#59-19	59-19	#59-19	59-19	#59-19	59-19	#59-19	59-19	#59-19
		59-19	#59-19	61-492	61-492	61-492	61-492	61-492	61-492	61-492
		61-492	61-492	61-492	61-492	61-492	69-2312	69-2312	69-2398	69-2399
		69-2399	69-2399	69-2399						
\$STUP	= 177777	#59-19	#59-19	59-19	#59-19	#59-19	59-19	#59-19	#59-19	59-19
		#59-19	#59-19	59-19	#59-19	#59-19	59-19			
\$SVLAD	016656	69-2398	#69-2398							
\$SVPC	= 001132	#59-28	59-28							
\$SWR	= 167000	#59-11	59-13	59-14	59-14	59-14	59-14	59-14	59-14	59-14
		59-14	60-29	60-29	60-29	61-492	61-492	61-492	61-492	61-492
		61-593	61-646	62-696	62-724	62-756	62-807	62-845	62-902	62-1001
		65-1031	65-1131	65-1156	65-1188	65-1244	65-1299	65-1351	65-1380	65-1412
		65-1476	65-1515	65-1565	65-1613	65-1659	66-1688	66-1741	66-1804	66-1847
		66-1871	66-1930	66-2004	68-2029	69-2101	69-2312	69-2312	69-2312	69-2312
		69-2312	69-2398	69-2398	69-2398	69-2398	69-2398	69-2398	69-2398	69-2398
		69-2398	69-2398	69-2398	69-2398	69-2398	69-2398	69-2398	69-2398	69-2398
		69-2398	69-2398	69-2399	69-2399	69-2399	69-2399	69-2399	69-2399	69-2399
		69-2399	69-2399	69-2399	69-2399	69-2405				
\$SWRMK	= 000000	69-2398								
\$TIMES	001234	#60-29	*61-492	*62-832	*65-1659	*66-1871	*66-1930	*68-2029	*69-2107	*69-2312
		*69-2398	69-2398	*69-2398	69-2398	69-2398				
\$TKB	001200	#60-29	69-2402	69-2402						
\$TKS	001176	#60-29	69-2402	69-2402						
\$TMP0	001224	#60-29	*62-906	69-2360						
\$TMP1	001226	#60-29	*61-653	61-654	62-677	*62-685				
\$TMP2	001230	#60-29								
\$TMP3	001232	#60-29								
\$TN	= 000041	59-13	#59-13	61-593	61-593	#61-593	61-627	61-646	61-646	#61-646
		62-696	62-696	#62-696	62-724	62-724	#62-724	62-756	62-756	#62-756
		62-788	62-807	62-807	#62-807	62-845	62-845	#62-845	62-902	62-902
		#62-902	62-971	62-983	62-991	62-1001	62-1001	#62-1001	65-1031	65-1031
		#65-1031	65-1067	65-1086	65-1096	65-1104	65-1119	65-1122	65-1125	65-1131
		65-1131	#65-1131	65-1144	65-1148	65-1156	65-1156	#65-1156	65-1168	65-1188
		65-1188	#65-1188	65-1244	65-1244	#65-1244	65-1299	65-1299	#65-1299	65-1351
		65-1351	#65-1351	65-1380	65-1380	#65-1380	65-1412	65-1412	#65-1412	65-1476

SYMBOL CROSS REFERENCE		REFERENCES								
SYMBOL	VALUE									
		65-1476	#65-1476	65-1496	65-1515	65-1515	#65-1515	65-1565	65-1565	#65-1565
		65-1613	65-1613	#65-1613	65-1659	65-1659	#65-1659	66-1677	66-1688	66-1688
		#66-1688	66-1741	66-1741	#66-1741	66-1804	66-1804	#66-1804	66-1847	66-1847
		#66-1847	66-1871	66-1871	#66-1871	66-1873	66-1930	66-1930	#66-1930	66-2004
		66-2004	#66-2004	68-2029	68-2029	#68-2029	69-2101	69-2101	#69-2101	
\$TPB	001204	#60-29	69-2402	69-2402	69-2402					
\$TPFLG	001211	#60-29	69-2402	69-2402	69-2402					
\$TPS	001202	#60-29	69-2402	69-2402	69-2402					
\$TRAP	020166	61-492	#69-2404							
\$TRAP2	020210	#69-2404	69-2404							
\$TRP	= 000006	#69-2404	69-2404	69-2404	69-2404	69-2404	#69-2404	69-2404	69-2404	69-2404
		69-2404	#69-2404	69-2404	69-2404	69-2404	69-2404	#69-2404	69-2404	69-2404
		69-2404	69-2404	#69-2404	69-2404	69-2404	69-2404	69-2404	#69-2404	69-2404
\$TRPAD	020222	69-2404	#69-2404							
\$TSTNM	001134	#60-29	*61-583	*69-2079	*69-2312	69-2392	69-2398	*69-2398	69-2398	69-2398
		69-2398	69-2399	69-2399	69-2399					
\$TYPBN	= *****	69-2404								
\$TYPDS	017224	#69-2401	69-2404	69-2404						
\$TYPE	017450	#69-2402	69-2404	69-2404						
\$TYPEC	017620	69-2402	69-2402	69-2402	#69-2402	69-2402				
\$TYPEX	017736	69-2402	69-2402	#69-2402						
\$TYPOC	017764	#69-2403	69-2404	69-2404						
\$TYPON	020000	69-2403	#69-2403	69-2404						
\$TYPOS	017740	#69-2403	69-2404							
\$XTSTR	016502	#69-2398								
\$\$GET4	= 000000	#69-2312	69-2312							
\$OFILL	020163	*69-2403	*69-2403	69-2403	#69-2403					
\$4OCAT	= *****	69-2398	69-2399							

