

PDP11

UNIBUS EXERCISER MODULE MD-11-DZKUB-A

EP-DZKUB-A-DL-A
COPYRIGHT 1976
FICHE 1 OF 1

NOV 1976
digital
MADE IN USA

Diagram 1	Diagram 2	Diagram 3	Diagram 4	Diagram 5	Diagram 6	Diagram 7	Diagram 8	Diagram 9	Diagram 10
Diagram 11	Diagram 12	Diagram 13	Diagram 14	Diagram 15	Diagram 16	Diagram 17	Diagram 18	Diagram 19	Diagram 20
Diagram 21	Diagram 22	Diagram 23	Diagram 24	Diagram 25	Diagram 26	Diagram 27	Diagram 28	Diagram 29	Diagram 30
Diagram 31	Diagram 32	Diagram 33	Diagram 34	Diagram 35	Diagram 36	Diagram 37	Diagram 38	Diagram 39	Diagram 40
Diagram 41	Diagram 42	Diagram 43	Diagram 44	Diagram 45	Diagram 46	Diagram 47	Diagram 48	Diagram 49	Diagram 50
Diagram 51	Diagram 52	Diagram 53	Diagram 54	Diagram 55	Diagram 56	Diagram 57	Diagram 58	Diagram 59	Diagram 60
Diagram 61	Diagram 62	Diagram 63	Diagram 64	Diagram 65	Diagram 66	Diagram 67	Diagram 68	Diagram 69	Diagram 70
Diagram 71	Diagram 72	Diagram 73	Diagram 74	Diagram 75	Diagram 76	Diagram 77	Diagram 78	Diagram 79	Diagram 80
Diagram 81	Diagram 82	Diagram 83	Diagram 84	Diagram 85	Diagram 86	Diagram 87	Diagram 88	Diagram 89	Diagram 90
Diagram 91	Diagram 92	Diagram 93	Diagram 94	Diagram 95	Diagram 96	Diagram 97	Diagram 98	Diagram 99	Diagram 100
Diagram 101	Diagram 102	Diagram 103	Diagram 104	Diagram 105	Diagram 106	Diagram 107	Diagram 108	Diagram 109	Diagram 110
Diagram 111	Diagram 112	Diagram 113	Diagram 114	Diagram 115	Diagram 116	Diagram 117	Diagram 118	Diagram 119	Diagram 120
Diagram 121	Diagram 122	Diagram 123	Diagram 124	Diagram 125	Diagram 126	Diagram 127	Diagram 128	Diagram 129	Diagram 130
Diagram 131	Diagram 132	Diagram 133	Diagram 134	Diagram 135	Diagram 136	Diagram 137	Diagram 138	Diagram 139	Diagram 140



B01

DATE: 13-OCT-76 15:25:17 Monitor IPC-D 5078 [1A3] *START*

```

DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
    
```

```

MTC: ML21-4:DAVIES-- distribution to ML21-4, slot 134
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
    
```

DATE: 13-OCT-76 15:25:17 Monitor IPC-D 5078 [1A3] *START*

```

DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
DDDD      AAAA      VV      III      EEEEE      SSS      T      OOO      MM MM
    
```

```

MTC: ML21-4:DAVIES-- distribution to ML21-4, slot 134
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
MM MM LLLL 2222 111 4444 < 11 333 4 4 >
    
```

DATE: 13-OCT-76 15:25:17 Monitor IPC-D 5078 [1A3] *START*

.REM I

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZKUB-A-D
PRODUCT NAME:	UNIBUS EXERCISOR MODULE DIAGNOSTIC
DATE CREATED:	FEB 3, 1975
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	WARREN SALTZ

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.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

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, BY DIGITAL EQUIPMENT CORPORATION

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11

45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

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

Handwritten marks

Handwritten marks

85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140

1.0 ABSTRACT

THE UNIBUS EXERCISOR (UBE) MODULE DIAGNOSTIC IS COMPRISED OF A SERIES OF TESTS THAT CHECK ALL PROGRAMMATICALLY ACCESSABLE 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. 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

H01

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11

MACY11 27(732) 17-SEP-76 15:49 PAGE 5

141

TAKES APPROXIMATELY 15 SECONDS PER UBE TESTED.

[Handwritten marks]

142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197

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

SW<15>=1 HALT ON ERROR
SW<14>=1 LOOP ON TEST

J01

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11

MACY11 27(732) 17-SEP-76 15:49 PAGE 7

198
199

SW<13>=1 INHIBIT ERROR TYPEDS
SW<12>=1 INHIBIT MOST TYPEOUTS EXCEPT ERROR

Handwritten initials

200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255

SW<11>=1 INHIBIT TEST ITERATIONS
SW<10>=1 BELL ON ERROR
SW<09>=1 LOOP ON ERROR
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).

256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311

5.8 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.

5.9 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.

5.10 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.

5.11 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.

5.12 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

MO1

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11

MACY11 27(732) 17-SEP-76 15:49 PAGE 10

312
313

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

314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369

BETWEEN THE EXERCISORS.

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:

070
071

1. TYPE - THIS ROUTINE IS USED TO TYPE ASCIZ MESSAGES.

2. TYPOC, 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.

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11

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.

```
[
.TITLE UNIBUS EXERCISOR MODULE DIAGNOSTIC
:*COPYRIGHT (C) OCT 29, 1974
:*DIGITAL EQUIPMENT CORP.
:*MAYNARD, MASS. 01754
:*
:*PROGRAM BY WARREN SALTZ
:*
:*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
:*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
:*
$TN=1
```

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11

4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39

```

.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
8              TEST POWER DOWN
7              INHIBIT TEST OF UBE 4
6              INHIBIT TEST OF UBE 3
5              INHIBIT TEST OF UBE 2
4              INHIBIT TETS OF UBE 1

```

```

.SBTTL BASIC DEFINITIONS

**INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE     ;;BASIC DEFINITION OF SCOPE CALL

```

001100

```

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

```

000011
000012
000015
000200
177776

177774
177772
177570
177570

```

**GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0         ;;GENERAL REGISTER
R1= %1         ;;GENERAL REGISTER
R2= %2         ;;GENERAL REGISTER
R3= %3         ;;GENERAL REGISTER
R4= %4         ;;GENERAL REGISTER
R5= %5         ;;GENERAL REGISTER
R6= %6         ;;GENERAL REGISTER
R7= %7         ;;GENERAL REGISTER
SP= %6         ;;STACK POINTER
PC= %7         ;;PROGRAM COUNTER

```

000000
000001
000002
000003
000004
000005
000006
000007
000006
000007

```

**PRIORITY LEVEL DEFINITIONS
PR0= 0         ;;PRIORITY LEVEL 0
PR1= 40        ;;PRIORITY LEVEL 1
PR2= 100       ;;PRIORITY LEVEL 2
PR3= 140       ;;PRIORITY LEVEL 3
PR4= 200       ;;PRIORITY LEVEL 4
PR5= 240       ;;PRIORITY LEVEL 5
PR6= 300       ;;PRIORITY LEVEL 6
PR7= 340       ;;PRIORITY LEVEL 7

```

000000
000040
000100
000140
000200
000240
000300
000340

540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
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

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

.*"SWITCH REGISTER" SWITCH DEFINITIONS

SW15= 100000
SW14= 40000
SW13= 20000
SW12= 10000
SW11= 4000
SW10= 2000
SW09= 1000
SW08= 400
SW07= 200
SW06= 100
SW05= 40
SW04= 20
SW03= 10
SW02= 4
SW01= 2
SW00= 1
.EQUIV SW09, SW9
.EQUIV SW08, SW8
.EQUIV SW07, SW7
.EQUIV SW06, SW6
.EQUIV SW05, SW5
.EQUIV SW04, SW4
.EQUIV SW03, SW3
.EQUIV SW02, SW2
.EQUIV SW01, SW1
.EQUIV SW00, SW0

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)

BIT15= 100000
BIT14= 40000
BIT13= 20000
BIT12= 10000
BIT11= 4000
BIT10= 2000
BIT09= 1000
BIT08= 400
BIT07= 200
BIT06= 100
BIT05= 40
BIT04= 20
BIT03= 10
BIT02= 4
BIT01= 2
BIT00= 1
.EQUIV BIT09, BIT9
.EQUIV BIT08, BIT8
.EQUIV BIT07, BIT7
.EQUIV BIT06, BIT6
.EQUIV BIT05, BIT5
.EQUIV BIT04, BIT4
.EQUIV BIT03, BIT3
.EQUIV BIT02, BIT2
.EQUIV BIT01, BIT1
.EQUIV BIT00, BIT0

```

596
597
598      000004
599      000010
600      000014
601      000014
602      000014
603      000020
604      000024
605      000030
606      000034
607      000060
608      000064
609      000240
610      170000
611
612
613      000000
614
615
616
617      000174
618      000174 000000
619      000176 000000
620
621      000200 000137 002632
622      001100
623      001100 012737 000137 000200
624      001106 012737 002632 000202
625      001114 020627 001020
626      001120 101002
627      001122 004767 015132
628      001126 000137 002632
629
630
631
632
633      001132
634      000046
635      000046 016100
636      000052
637      000052 000000
638      001132

; *BASIC "CPU" TRAP VECTOR ADDRESSES
ERRVEC= 4      ; TIME OUT AND OTHER ERRORS
RESVEC= 10     ; RESERVED AND ILLEGAL INSTRUCTIONS
TBITVEC=14    ; "T" BIT
TRTVEC= 14    ; TRACE TRAP
BPTVEC= 14    ; BREAKPOINT TRAP (BPT)
IOTVEC= 20    ; INPUT/OUTPUT TRAP (IOT) **SCOPE**
PWRVEC= 24    ; POWER FAIL
EMTVEC= 30    ; EMULATOR TRAP (EMT) **ERROR**
TRAPVEC=34    ; "TRAP" TRAP
TKVEC= 60     ; TTY KEYBOARD VECTOR
TPVEC= 64     ; TTY PRINTER VECTOR
PIRQVEC=240   ; PROGRAM INTERRUPT REQUEST VECTOR
DB=170000     ; DATA BUFFER OF LOWEST ADDRESS UBE
.SBTTL TRAP CATCHER

.=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
.=174
DISPREG: .WORD 0      ; SOFTWARE DISPLAY REGISTER
SWREG:   .WORD 0      ; SOFTWARE SWITCH REGISTER
.SBTTL STARTING ADDRESS(ES)
JMP      @#START ; JUMP TO STARTING ADDRESS OF PROGRAM
.=1100
RSTART:  MOV      #000137,@#200 ; RESTART HERE IF HALTED IN INTERRUPT TEST
        MOV      #START,@#202
        CMP R6,#1020 ; WAS VECTOR AREA DESTROYED IN INT. TEST?
        BHI B      ; BRANCH IF NO
        JSR PC,RVEC ; RESTORE VECTOR AREA
B:       JMP      @#START ; GO TO BEGINNING OF PROGRAM
.SBTTL ACT11 HOOKS

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

```


.SBTTL COMMON TAGS

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

639
640
641
642
643
644
645 001132
646 001132
647 001132 000000
648 001134 000
649 001135 000
650 001136 000000
651 001140 000000
652 001142 000000
653 001144 000000
654 001146 000
655 001147 001
656 001150 000000
657 001152 000000
658 001154 000000
659 001156 000000
660 001160 000000
661 001162 000000
662 001164 000000
663 001166 000
664 001167 000
665 001170 000000
666 001172 177570
667 001174 177570
668 001176 177560
669 001200 177562
670 001202 177564
671 001204 177566
672 001206 000
673 001207 002
674 001210 012
675 001211 000
676 001212 000000
677
678 001214 000000
679 001216 000000
680 001220 000000
681 001222 000000
682 001224 000000
683 001226 000000
684 001230 000000
685 001232 000000
686 001234 000000
687 001236 000000
688 001240 177607 000377
689 001244 077
690 001245 015
691 001246 000012
692

. =1132

SCMTAG: .WORD 0
SPASS: .WORD 0
STSTNM: .BYTE 0
SERFLG: .BYTE 0
SICNT: .WORD 0
SLPADR: .WORD 0
SLPERR: .WORD 0
SERTTL: .WORD 0
SITEMB: .BYTE 0
SERMAX: .BYTE 1
SERRPC: .WORD 0
SGDADR: .WORD 0
SBDADR: .WORD 0
SGDDAT: .WORD 0
SBDDAT: .WORD 0
SAUTOB: .BYTE 0
\$INTAG: .BYTE 0
SWR: .WORD DSWR
DISPLAY: .WORD DDISP
\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566
\$NULL: .BYTE 0
\$FILLS: .BYTE 2
\$FILLC: .BYTE 12
\$TPFLG: .BYTE 0
\$REGAD: .WORD 0
\$REG0: .WORD 0
\$REG1: .WORD 0
\$REG2: .WORD 0
\$REG3: .WORD 0
\$TMP0: .WORD 0
\$TMP1: .WORD 0
\$TMP2: .WORD 0
\$TMP3: .WORD 0
\$TIMES: 0
\$ESCAPE: 0
\$BELL: .ASCIZ <207><377><377>
\$QUES: .ASCII /?/
\$CRLF: .ASCII <15>
\$LF: .ASCIZ <12>

START OF COMMON TAGS
CONTAINS PASS COUNT
CONTAINS THE TEST NUMBER
CONTAINS ERROR FLAG
CONTAINS SUBTEST ITERATION COUNT
CONTAINS SCOPE LOOP ADDRESS
CONTAINS SCOPE RETURN FOR ERRORS
CONTAINS TOTAL ERRORS DETECTED
CONTAINS ITEM CONTROL BYTE
CONTAINS MAX. ERRORS PER TEST
CONTAINS PC OF LAST ERROR INSTRUCTION
CONTAINS ADDRESS OF 'GOOD' DATA
CONTAINS ADDRESS OF 'BAD' DATA
CONTAINS 'GOOD' DATA
CONTAINS 'BAD' DATA
RESERVED--NOT TO BE USED
AUTOMATIC MODE INDICATOR
INTERRUPT MODE INDICATOR
ADDRESS OF SWITCH REGISTER
ADDRESS OF DISPLAY REGISTER
TTY KBD STATUS
TTY KBD BUFFER
TTY PRINTER STATUS REG. ADDRESS
TTY PRINTER BUFFER REG. ADDRESS
CONTAINS NULL CHARACTER FOR FILLS
CONTAINS # OF FILLER CHARACTERS REQUIRED
INSERT FILL CHARS. AFTER A "LINE FEED"
"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
CONTAINS THE ADDRESS FROM WHICH (\$REG0) WAS OBTAINED
CONTAINS ((\$REGAD)+0)
CONTAINS ((\$REGAD)+2)
CONTAINS ((\$REGAD)+4)
CONTAINS ((\$REGAD)+6)
USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED
MAX. NUMBER OF ITERATIONS
ESCAPE ON ERROR ADDRESS
CODE FOR BELL
QUESTION MARK
CARRIAGE RETURN
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

693
694
695
696
697
698
699
700
701
702
703
704
705
706
707 001250
708
709 001250 020576
710 001252 000000
711 001254 000000
712 001256 000000
713
714 001260 020660
715 001262 020720
716 001264 020746
717 001266 000000
718
719 001270 020754
720 001272 021021
721 001274 021030
722 001276 000000
723
724 001300 021034
725 001302 021102
726 001304 021142
727 001306 000000
728
729 001310 021152
730 001312 021102
731 001314 021142
732 001316 000000
733
734 001320 021220
735 001322 021102
736 001324 021142
737 001326 000000
738
739 001330 021270
740 001332 021332
741 001334 021410
742 001336 000000
743
744 001340 021416
745 001342 000000
746 001344 000000
747 001346 000000
748

\$ERRTB:
;ITEM1 EM1
0
0
0
;ITEM2 EM2
DH2
DT2
0
;ITEM3 EM3
DH3
DT3
0
;ITEM 4 EM4
DH4
DT4
0
;ITEM 5 EM5
DH4
DT4
0
;ITEM 6 EM6
DH4
DT4
0
;ITEM 7 EM7
DH7
DT7
0
;ITEM 8 EM8
0
0
0
;ITEM 9

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11 ERROR POINTER TABLE

749	001350	021503	EM9
750	001352	000000	0
751	001354	000000	0
752	001356	000000	0
753			; ITEM 10
754	001360	021544	EM10
755	001362	000000	0
756	001364	000000	0
757	001366	000000	0
758			; ITEM 11
759	001370	021605	EM11
760	001372	000000	0
761	001374	000000	0
762	001376	000000	0
763			; ITEM 12
764	001400	021651	EM12
765	001402	000000	0
766	001404	000000	0
767	001406	000000	0
768			; ITEM 13
769	001410	000000	0
770	001412	000000	0
771	001414	000000	0
772	001416	000000	0
773			; ITEM 14
774	001420	021716	EM14
775	001422	000000	0
776	001424	000000	0
777	001426	000000	0
778			; ITEM 15
779	001430	021747	EM15
780	001432	022033	DH15
781	001434	021030	DT3
782	001436	000000	0
783			; ITEM 16
784	001440	022055	EM16
785	001442	000000	0
786	001444	000000	0
787	001446	000000	0
788			; ITEM 17
789	001450	022150	EM17
790	001452	022207	DH17
791	001454	020746	DT2
792	001456	000000	0
793			; ITEM 18
794	001460	022234	EM18
795	001462	022323	DH18
796	001464	021030	DT3
797	001466	000000	0
798			; ITEM 19
799	001470	022336	EM19
800	001472	022417	DH19
801	001474	021030	DT3
802	001476	000000	0
803			; ITEM 20
804	001500	022440	EM20

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11 ERROR POINTER TABLE

805	001502	000000	0
806	001504	000000	0
807	001506	000000	0
808			
809	001510	022507	; ITEM 21
810	001512	000000	EM21
811	001514	000000	0
812	001516	000000	0
813			
814	001520	022545	; ITEM 22
815	001522	000000	EM22
816	001524	000000	0
817	001526	000000	0
818			
819	001530	022610	; ITEM 23
820	001532	000000	EM23
821	001534	000000	0
822	001536	000000	0
823			
824	001540	022654	; ITEM 24
825	001542	022723	EM24
826	001544	023000	DH24
827	001546	000000	DT24
828			0
829	001550	023012	; ITEM 25
830	001552	022723	EM25
831	001554	023000	DH24
832	001556	000000	DT24
833			0
834	001560	023052	; ITEM 26
835	001562	022723	EM26
836	001564	023000	DH24
837	001566	000000	DT24
838			0
839	001570	023113	; ITEM 27
840	001572	022723	EM27
841	001574	023000	DH24
842	001576	000000	DT24
843			0
844	001600	023153	; ITEM 28
845	001602	000000	EM28
846	001604	000000	0
847	001606	000000	0
848			
849	001610	023202	; ITEM 29
850	001612	000000	EM29
851	001614	000000	0
852	001616	000000	0
853			
854	001620	023231	; ITEM 30
855	001622	000000	EM30
856	001624	000000	0
857	001626	000000	0
858			
859	001630	023261	; ITEM 31
860	001632	000000	EM31
			0

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11 ERROR-POINTER TABLE

861	001634	000000	0
862	001636	000000	0
863			; ITEM 32
864	001640	023311	EM32
865	001642	022723	DH24
866	001644	023000	DT24
867	001646	000000	0
868			; ITEM 33
869	001650	023360	EM33
870	001652	022723	DH24
871	001654	023000	DT24
872	001656	000000	0
873			; ITEM 34
874	001660	023415	EM34
875	001662	023465	DH34
876	001664	021030	DT3
877	001666	000000	0
878			; ITEM 35
879	001670	023504	EM35
880	001672	023561	DH35
881	001674	020746	DT2
882	001676	000000	0
883			; ITEM 36
884	001700	023607	EM36
885	001702	023656	DH36
886	001704	021030	DT3
887	001706	000000	0
888			; ITEM 37
889	001710	023666	EM37
890	001712	023561	DH35
891	001714	020746	DT2
892	001716	000000	0
893			; ITEM 38
894	001720	023725	EM38
895	001722	023561	DH35
896	001724	020746	DT2
897	001726	000000	0
898			; ITEM 39
899	001730	024015	EM39
900	001732	000000	0
901	001734	000000	0
902	001736	000000	0
903			; ITEM 40
904	001740	024073	EM40
905	001742	000000	0
906	001744	000000	0
907	001746	000000	0
908			; ITEM 41
909	001750	024151	EM41
910	001752	000000	0
911	001754	000000	0
912	001756	000000	0
913			; ITEM 42
914	001760	024213	EM42
915	001762	000000	0
916	001764	000000	0

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11 ERROR POINTER TABLE

917	001766	000000	
918			; ITEM 43
919	001770	024252	EM43
920	001772	024336	DH43
921	001774	020746	DT2
922	001776	000000	0
923			; ITEM 44
924	002000	024367	EM44
925	002002	000000	0
926	002004	000000	0
927	002006	000000	0
928			; ITEM 45
929	002010	024433	EM45
930	002012	000000	0
931	002014	000000	0
932	002016	000000	0
933			; ITEM 46
934	002020	024460	EM46
935	002022	024530	DH46
936	002024	021142	DT4
937	002026	000000	0
938			; ITEM 47
939	002030	024566	EM47
940	002032	024336	DH43
941	002034	020746	DT2
942	002036	000000	0
943			; ITEM 48
944	002040	024566	EM47
945	002042	000000	0
946	002044	000000	0
947	002046	000000	0
948			; ITEM 49
949	002050	024644	EM49
950	002052	000000	0
951	002054	000000	0
952	002056	000000	0
953			; ITEM 50
954	002060	024644	EM49
955	002062	024336	DH43
956	002064	020746	DT2
957	002066	000000	0
958			; ITEM 51
959	002070	024723	EM51
960	002072	000000	0
961	002074	000000	0
962	002076	000000	0
963			; ITEM 52
964	002100	024754	EM52
965	002102	000000	0
966	002104	000000	0
967	002106	000000	0
968			; ITEM 53
969	002110	025007	EM53
970	002112	000000	0
971	002114	000000	0
972	002116	000000	0

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11 ERROR POINTER TABLE

973
 974 002120 025061
 975 002122 000000
 976 002124 000000
 977 002126 000000
 978
 979 002130 024252
 980 002132 000000
 981 002134 000000
 982 002136 000000
 983
 984 002140 025330
 985 002142 000000
 986 002144 000000
 987 002146 000000
 988
 989 002150 025407
 990 002152 000000
 991 002154 000000
 992 002156 000000
 993
 994 002160 025437
 995 002162 022033
 996 002164 021030
 997 002166 000000
 998
 999 002170 025460
 1000 002172 000000
 1001 002174 000000
 1002 002176 000000
 1003
 1004 002200 025526
 1005 002202 000000
 1006 002204 000000
 1007 002206 000000
 1008
 1009 002210 025554
 1010 002212 000000
 1011 002214 000000
 1012 002216 000000
 1013
 1014 002220 025603
 1015 002222 000000
 1016 002224 000000
 1017 002226 000000
 1018
 1019 002230 025633
 1020 002232 022033
 1021 002234 021030
 1022 002236 000000
 1023
 1024 002240 025663
 1025 002242 000000
 1026 002244 000000
 1027 002246 000000
 1028

; ITEM 54
 EM54
 0
 0
 0
 ; ITEM 55
 EM43
 0
 0
 0
 ; ITEM 56
 EM56
 0
 0
 0
 ; ITEM 57
 EM57
 0
 0
 0
 ; ITEM 58
 EM58
 DH15
 DT3
 0
 ; ITEM 59
 EM59
 0
 0
 0
 ; ITEM 60
 EM60
 0
 0
 0
 ; ITEM 61
 EM61
 0
 0
 0
 ; ITEM 62
 EM62
 0
 0
 0
 ; ITEM 63
 EM63
 DH15
 DT3
 0
 ; ITEM 64
 EM64
 0
 0
 0
 ; ITEM 65

1069	002250	025761	EM65
1070	002252	026041	DH65
1071	002254	021030	DT3
1072	002256	000000	0
		: ITEM 66	EM66
1073	002260	026060	000000
1074	002262	000000	000000
1075	002264	000000	000000
1076	002266	000000	000000
		: ITEM 67	EM67
1077	002270	026106	DH65
1078	002272	026041	DT3
1079	002274	021030	0
1080	002276	000000	0
		: ITEM 68	EM68
1081	002300	000000	DH65
1082	002302	026041	DT3
1083	002304	021030	0
1084	002306	000000	0
		: ITEM 69	EM69
1085	002310	026150	000000
1086	002312	000000	000000
1087	002314	000000	000000
1088	002316	000000	000000
		: ITEM 70	EM70
1089	002320	026201	000000
1090	002322	000000	000000
1091	002324	000000	000000
1092	002326	000000	000000
		: ITEM 71	EM71
1093	002330	026262	000000
1094	002332	000000	000000
1095	002334	000000	000000
1096	002336	000000	000000
		: ITEM 72	EM72
1097	002340	026310	000000
1098	002342	000000	000000
1099	002344	000000	000000
1070	002346	000000	000000
		: ITEM 73	EM73
1071	002350	026340	000000
1072	002352	000000	000000
1073	002354	000000	000000
1074	002356	000000	000000
		: ITEM 74	EM74
1075	002360	026405	000000
1076	002362	000000	000000
1077	002364	000000	000000
1078	002366	000000	000000
		: ITEM 75	EM75
1079	002370	026427	000000
1080	002372	000000	000000
1081	002374	000000	000000
1082	002376	000000	000000
		: ITEM 76	EM76
1083			
1084	002400	026447	

UNIBUS EXERCISOR MODULE DIAGNOSTIC
DZKUBA.P11 ERROR POINTER TABLE

MACY11 27(732) 17-SEP-76 15:49 PAGE 26

1085	002402	000000
1086	002404	000000
1087	002406	000000
1088		
1089	002410	026474
1090	002412	000000
1091	002414	000000
1092	002416	000000
1093		
1094	002420	026522
1095	002422	021102
1096	002424	021142
1097	002426	000000
1098		
1099	002430	000000
1100	002432	000000
1101	002434	000000
1102	002436	000000
1103		
1104	002440	026562
1105	002442	000000
1106	002444	000000
1107	002446	000000
1108		
1109	002450	026624
1110	002452	024530
1111	002454	021142
1112	002456	000000
1113		
1114	002460	026650
1115	002462	000000
1116	002464	000000
1117	002466	000000
1118		
1119	002470	026715
1120	002472	000000
1121	002474	000000
1122	002476	000000
1123		
1124	002500	026776
1125	002502	000000
1126	002504	000000
1127	002506	000000
1128	002510	000000
1129	002512	000000
1130	002514	000000
1131	002516	000000
1132	002520	000000
1133	002522	000000
1134	002524	000000
1135	002526	000000
1136	002530	000000
1137	002532	000000
1138	002534	170014
1139	002536	000000
1140	002540	000000

```

0
0
0
:ITEM 77
EM77
0
0
0
:ITEM 78
EM78
DH4
DT4
0
:ITEM 79
0
0
0
0
0
:ITEM 80
EM80
0
0
0
0
0
:ITEM 81
EM81
DH46
DT4
0
:ITEM 82
EM82
0
0
0
0
0
:ITEM 83
EM83
0
0
0
0
0
:ITEM 84
EM84
0
0
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

```

```

: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

```

1141	002542	000000	BE1BA:	.WORD 0	:: BEBA ADDRESS OF FIRST UBE TESTED
1142	002544	000000	BE1CR1:	.WORD 0	:: BECR1 ADDRESS OF FIRST UBE TESTED
1143	002546	000000	BE1CR2:	.WORD 0	:: BECR2 ADDRESS OF FIRST UBE TESTED
1144	002550	000000	BE1RE:	.WORD 0	:: CLEAR ERROR ADDRESS OF FIRST UBE TESTED
1145	002552	000000	BE1VEC:	.WORD 0	:: INTERRUPT VECTOR ADDRESS OF FIRST UBE TESTED
1146	002554	000000	BE2BD:	.WORD 0	:: BEBD ADDRESS OF SECOND UBE TESTED
1147	002556	000000	BE2CC:	.WORD 0	:: BECC ADDRESS OF SECOND UBE TESTED
1148	002560	000000	BE2BA:	.WORD 0	:: BEBA ADDRESS OF SECOND UBE TESTED
1149	002562	000000	BE2CR1:	.WORD 0	:: BECR1 ADDRESS OF SECOND UBE TESTED
1150	002564	000000	BE2CR2:	.WORD 0	:: BECR2 ADDRESS OF SECOND UBE TESTED
1151	002566	000000	BE2RE:	.WORD 0	:: CLEAR ERROR ADDRESS OF SECOND UBE TESTED
1152	002570	000000	BE2VEC:	.WORD 0	:: INTERRUPT VECTOR ADDRESS OF SECOND UBE TESTED
1153	002572	000000	BE3BD:	.WORD 0	:: BEBD ADDRESS OF THIRD UBE TESTED
1154	002574	000000	BE3CC:	.WORD 0	:: BECC ADDRESS OF THIRD UBE TESTED
1155	002576	000000	BE3BA:	.WORD 0	:: BEBA ADDRESS OF THIRD UBE TESTED
1156	002600	000000	BE3CR1:	.WORD 0	:: BECR1 ADDRESS OF THIRD UBE TESTED
1157	002602	000000	BE3CR2:	.WORD 0	:: BECR2 ADDRESS OF THIRD UBE TESTED
1158	002604	000000	BE3RE:	.WORD 0	:: CLEAR ERROR ADDRESS OF THIRD UBE TESTED
1159	002606	000000	BE3VEC:	.WORD 0	:: INTERRUPT VECTOR ADDRESS OF THIRD UBE TESTED
1160	002610	000000	BE4BD:	.WORD 0	:: BEBD ADDRESS OF FOURTH UBE TESTED
1161	002612	000000	BE4CC:	.WORD 0	:: BECC ADDRESS OF FOURTH UBE TESTED
1162	002614	000000	BE4BA:	.WORD 0	:: BEBA ADDRESS OF FOURTH UBE TESTED
1163	002616	000000	BE4CR1:	.WORD 0	:: BECR1 ADDRESS OF FOURTH UBE TESTED
1164	002620	000000	BE4CR2:	.WORD 0	:: BECR2 ADDRESS OF FOURTH UBE TESTED
1165	002622	000000	BE4RE:	.WORD 0	:: CLEAR ERROR ADDRESS OF FOURTH UBE TESTED
1166	002624	000000	BE4VEC:	.WORD 0	:: INTERRUPT VECTOR ADDRESS OF FOURTH UBE TESTED
1167	002626	000000	UCNT:	.WORD 0	:: COUNT OF UBE TESTED
1168	002630	000000	NO:	.WORD 0	:: INDEX NUMBER FOR ADDRESS OF 1,2,3,4 UBE
1169			*****		
1170			*****		
1171	002632		START:		
1172			.SBTTL INITIALIZE THE COMMON TAGS		
1173			;;CLEAR THE COMMON TAGS (\$CMTAG) AREA		
1174	002632	012706	001132	MOV	;\$CMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
1175	002636	005026		CLR	(R6)+ ;:CLEAR MEMORY LOCATION
1176	002640	022706	001172	CMP	;\$WR,R6 ;:DONE?
1177	002644	001374		BNE	.-6 ;:LOOP BACK IF NO
1178	002646	012706	001100	MOV	;\$STACK,SP ;:SETUP THE STACK POINTER
1179				;;INITIALIZE A FEW VECTORS	
1180	002652	012737	016370	000020	MOV ;\$SCOPE,;\$IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
1181	002660	012737	000340	000022	MOV ;\$340,;\$IOTVEC+2 ;:LEVEL 7
1182	002666	012737	016620	000030	MOV ;\$ERROR,;\$EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
1183	002674	012737	000340	000032	MOV ;\$340,;\$EMTVEC+2 ;:LEVEL 7
1184	002702	012737	020014	000034	MOV ;\$STRAP,;\$TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
1185	002710	012737	000340	000036	MOV ;\$340,;\$TRAPVEC+2 ;:LEVEL 7
1186	002716	012737	020064	000024	MOV ;\$SPWRDN,;\$PWRVEC ;:POWER FAILURE VECTOR
1187	002724	012737	000340	000026	MOV ;\$340,;\$PWRVEC+2 ;:LEVEL 7
1188	002732	016767	013110	013100	MOV SENDCT,SEOPCT ;:SETUP END-OF-PROGRAM COUNTER
1189	002740	005067	176270		CLR \$TIMES ;:INITIALIZE NUMBER OF ITERATIONS
1190	002744	005067	176266		CLR \$ESCAPE ;:CLEAR THE ESCAPE ON ERROR ADDRESS
1191	002750	112767	000001	176171	MOVB #1,\$ERMAX ;:ALLOW ONE ERROR PER TEST
1192	002756	012767	002756	176154	MOV #,\$SLPADR ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
1193	002764	012767	002764	176150	MOV #,\$SLPERR ;:SETUP THE ERROR LOOP ADDRESS
1194					;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1195					;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
1196	002772	013746	000004	MOV	;\$ERRVEC,-(SP) ;:SAVE ERROR VECTOR


```

1197 002776 012737 003032 000004      MOV      #64$,D#ERRVEC      ;; SET UP ERROR VECTOR
1198 003004 012767 177570 176160      MOV      #DSWR,SWR         ;; SETUP FOR A HARDWARE SWICH REGISTER
1199 003012 012767 177570 176154      MOV      #DDISP,DISPLAY    ;; AND A HARDWARE DISPLAY REGISTER
1200 003020 022777 177777 176144      CMP      #-1,D#SWR         ;; TRY TO REFERENCE HARDWARE SWR
1201 003026 001012                BNE      66$               ;; BRANCH IF NO TIMEOUT TRAP OCCURRED
1202                                ;; AND THE HARDWARE SWR IS NOT = -1
1203 003030 000403                BR       65$               ;; BRANCH IF NO TIMEOUT
1204 003032 012716 003040      64$:    MOV      #65$, (SP)      ;; SET UP FOR TRAP RETURN
1205 003036 000002                RTI
1206 003040 012767 000176 176124      65$:    MOV      #SWREG,SWR     ;; POINT TO SOFTWARE SWR
1207 003046 012767 000174 176120      MOV      #DISPREG,DISPLAY
1208 003054 012637 000004      66$:    MOV      (SP)+,D#ERRVEC  ;; RESTORE ERROR VECTOR
1209
1210 003060 032737 010000 001172      BIT      #SW12,D#SWR       ;; INHIBIT TYPEOUTS?
1211 003066 001004                BNE      START1           ;; BRANCH IF YES
1212 003070 104401 027573      TYPE    ,MSG16            ;; UBE MODULE TEST
1213 003074 104401 027274      TYPE    ,MSG12            ;; JUMPER W1 SHOULD BE IN TO PREVENT MULTIPLE S5YNS
1214 003100 005067 177404      START1: CLR    EMAP         ;; INIT. EMAP
1215 003104 012767 000001 177402      MOV      #1,SPTR         ;; INITIALIZE SWITCH POINTER TO LOOK AT FIRST SWITCH
1216 003112 012767 002632 176022      MOV      #START,$LPERR   ;; SET UP RETURN FOR ERROR1
1217 003120 012737 003230 000004      MOV      #MTRAP,D#4      ;; SET UP MAP TRAP
1218 003126 012737 000340 000006      MOV      #340,D#6        ;; SET PSW PRIORITY=7
1219 003134 012701 170000      MOV      #DB,R1          ;; DATA REG ADDR. OF FIRST REG
1220 003140 012700 000001      MOV      #1,R0           ;; LD PTER
1221 003144 005711      LOOP1: TST    (R1)         ;; LOOK IF EXER. PRESENT, NO TRAPS
1222 003146 050067 177336      BIS      R0,EMAP         ;; YES, INDIC. EXER. PRESENT
1223 003152 062701 000020      LOOP2:  ADD    #20,R1      ;; LOOK AT NEXT EXER. ADDR.
1224 003156 006100      ROL      R0              ;; UPDATE PTER
1225 003160 020027 000020      CMP      R0,#20          ;; AT LAST UBE?
1226 003164 001367      BNE      LOOP1          ;; BRANCH IF NOT AT LAST POSSIBLE EXER.
1227 003166 012737 000006 000004      A:      MOV      #6,D#4        ;; RESTORE TRAP CATCHER
1228 003174 005037 000006      CLR      D#6
1229 003200 032737 010000 001172      BIT      #SW12,D#SWR     ;; INHIBIT TYPEOUTS?
1230 003206 001007      BNE      1$             ;; BRANCH IF YES
1231 003210 104401 020242      TYPE    ,MSG1            ;; TYPE MAP
1232 003214 016746 177270      MOV      EMAP,-(SP)      ;; SAVE EMAP FOR TYPEOUT
1233 003220 104402      TYPOC                ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
1234 003222 104401 001245      TYPE    ,$CRLF
1235 003226 000414      1$:     BR      IADD        ;; GO CALC. ADDRESSES OF UBE
1236
1237 003230 020027 000010      MTRAP:  CMP      R0,#10      ;; AT END OF UBE ADDRESS SPACE?
1238 003234 001346      BNE      LOOP2          ;; NO LOOK AT NEXT EXER.
1239 003236 026727 177246 000000      CMP      EMAP,#0        ;; YES, IS MAP = 0?
1240 003244 001350      BNE      A              ;; NO, BRANCH TO A
1241 003246 104001      ERROR  #D1             ;; NO RESPONSE TO REG ADDRESSES OR NO DEVICE PRESENT
1242 003250 004767 013060      JSR      PC,TERRPC      ;; TYPE PC OF ERROR MSG
1243 003254 000167 012532      JMP      $EOP           ;; GO TO END OF TEST
1244
1245      ; ////////////////////////////////////////////////////////////////////
1246      ; ROUTINE TO CALCULATE ADDRESSES OF UBE TESTED
1247      ; ////////////////////////////////////////////////////////////////////
1247 003260 012767 167760 177230      IADD:   MOV      #167760, BEBD  ;; INITIALIZE BEBD
1248 003266 012767 167762 177224      MOV      #167762, BECC  ;; INITIALIZE BECC
1249 003274 012767 167764 177220      MOV      #167764, BEBA  ;; INITIALIZE BEBA
1250 003302 012767 167766 177214      MOV      #167766, BECR1 ;; INITIALIZE BECR1
1251 003310 012767 167776 177210      MOV      #167776, BECR2 ;; INITIALIZE BECR2
1252 003316 012767 167770 177204      MOV      #167770, BERE  ;; INITIALIZE BERE

```

```

1253 003324 012767 170014 177202      MOV #170014, BEGO      ;INITIALIZE BEGO
1254 003332 012767 000504 177172      MOV #504, INTVEC      ;INITIALIZE INTERRUPT VECTOR
1255 003340 012700 002536                MOV #BE1BD, RO        ;GET POINTER TO PERMANENT VECTOR AREA
1256 003344 005020                CLR (RO)+             ;CLEAR PERMANENT VECTOR AREA
1257 003346 020027 002630      IS:  CMP RO, #NO         ;ENTIRE AREA CLEARED?
1258 003352 001374                BNE IS               ;BRANCH IF NO
1259 003354 012767 002536 177246      MOV #BE1BD, NO        ;INITIALIZE POINTER TO BE1BD
1260 003362 016767 177122 177122      MOV EMAP, TMAP        ;MOVE MAP TO WORK AREA
1261 003370 062767 000020 177120      ACALC: ADD #20, BEBD        ;CALC. ADDR. OF BEBD TESTING
1262 003376 062767 000020 177114      ADD #20, BECC         ;CALC. ADDR. OF BECC TESTING
1263 003404 062767 000020 177110      ADD #20, BEBA         ;CALC. ADDR. OF BEBA TESTING
1264 003412 062767 000020 177104      ADD #20, BECR1        ;CALC. ADDR. OF BECR1 TESTING
1265 003420 062767 000020 177100      ADD #20, BECR2        ;CALC. ADDR. OF BECR2 TESTING
1266 003426 062767 000020 177074      ADD #20, BERE         ;CALC. ADDR. OF BERE TESTING
1267 003434 062767 000004 177070      ADD #4, INTVEC        ;CALC. ADDR. OF INTERRUPT VECTOR
1268 003442 000241                CLC                  ;INIT. CARRY
1269 003444 006267 177042                ASR TMAP             ;LOOK FOR BIT INDICATING EXERCISOR
1270 003450 042767 100000 177034      BIC #100000, TMAP     ;CLEAR MSB IF SET
1271 003456 103405                BCS C                ;IF EXERCISOR PRESENT GO SEE IF TO BE TESTED
1272 003460 005767 177026                TST TMAP            ;ANY EXERCISORS LEFT?
1273 003464 001341                BNE ACALC           ;BRANCH IF MORE
1274 003466 000167 010710                JMP LAST            ;GO TO LAST TEST
1275 003472 032767 000020 177014      C:  BIT #20, SPTR      ;TESTED 4 UBE?
1276 003500 001402                BEQ D                ;BRANCH IF NO
1277 003502 000167 010674                JMP LAST            ;GO TO LAST TEST
1278 003506 036737 177002 001172      D:  BIT SPTR, 2#SWR    ;SHOULD THIS UBE BE TESTED?
1279 003514 001403                BEQ E                ;BRANCH IF YES
1280 003516 006367 176772                ASL SPTR            ;ROTATE POINTER TO NEXT SWITCH
1281 003522 000722                BR ACALC            ;LOOK FOR NEXT UBE
1282 003524 006367 176764      E:  ASL SPTR            ;ROTATE POINTER TO NEXT SWITCH
1283 003530 005267 177072                INC UCNT            ;UPDATE COUNT OF UBE TESTED
1284 003534 104401 027422                TYPE ,MSG13         ;TESTING UBE WITH BEBD ADDRESS:
1285 003540 016746 176752                MOV BEBD, -(SP)     ;SAVE BEBD FOR TYPEOUT
1286 003544 104402                TYPOC               ;GO TYPE--OCTAL ASCII(ALL DIGITS)
1287 003546 104401 001245                TYPE ,SCLF
1288
1289
1290      ;////////////////////
1291      ;ROUTINE TO STORE TEMPORARY ADDRESS OF UBE TESTING IN PERMANENT LOC
1292      ;////////////////////
1293      MOV NO, R1
1294      MOV #BEBD, RO
1295      MOV (RO)+, (R1)+
1296      CMP RO, #BEGO
1297      BNE F
1298      ADD #16, NO
1299
1300      MOV #FIRST, $LPADR
1301      MOV #FIRST, $LPERR
1302      CLRB $STNM
1303      RESET
1304
1305      ;*****
1306      ;*TEST 1 TEST ALL UBE REG CAN BE CLEARED
1307      ;*
1308      ;*RD CONTAINS ADDRESS OF REG UNDER TEST
1309      ;*

```



```

1309
1310
1311 003622 000004
1312 003624 012706 001100
1313 003630 012737 000340 177776
1314 003636 012737 004022 000004
1315 003644 012737 000340 000006
1316 003652 012777 000000 176646
1317 003660 005077 176644
1318 003664 016700 176626
1319 003670 005010
1320 003672 020067 176626
1321 003676 001425
1322 003700 005710
1323 003702 001421
1324 003704 010067 175304
1325 003710 011067 175302
1326 003714 104002
1327 003716 020067 176604
1328 003722 001006
1329 003724 032777 020000 176574
1330 003732 001402
1331 003734 104401 027464
1332 003740 004767 012370
1333 003744 000433
1334 003746 005720
1335 003750 000747
1336 003752 022777 000200 176544
1337 003760 001351
1338 003762 016700 176540
1339 003766 005077 176536
1340 003772 005077 176530
1341 003776 032777 157777 176522
1342 004004 001337
1343 004006 012737 000006 000004
1344 004014 005037 000006
1345 004020 000414
1346
1347 004022 011667 175166
1348 004026 104003
1349 004030 004767 012300
1350
1351 004034 012737 000006 000004
1352 004042 005037 000006
1353 004046 000167 010324
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364

```

```

;*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
MOV BEBD, RO ;SETUP TO LOOK AT FIRST REG.
TO1L01: CLR (RO) ;CLR UBE REG
CMP RO, BECR1 ;TESTING BECR1?
BEQ TO1L04 ;BRANCH IF YES
TST (RO) ;IS REG CLEARED?
BEQ TO1L02 ;BRANCH IF YES
TO1L03: MOV RO, $REGO ;SAVE FAILING ADDRESS
MOV (RO), $REG1 ;SAVE BAD DATA
ERROR ID2 ;FATAL ERROR:REG FAILED TO CLEAR
CMP RO, BECR2 ;DID BECR2 FAIL?
BNE TO1L06 ;BRANCH IF NO
BIT #20000, @BECR2 ;WAS CCOVF =1?
BEQ TO1L06 ;BRANCH IF NO
TYPE MSG14 ;DISREGARD BIT 13=1 OF BECR2
TO1L06: JSR PC, TERRPC ;TYPE PC OF ERROR MSG
BR TO1L05 ;RESTORE TRAP
TO1L02: TST (RO)+ ;INC ADDRESS
BR TO1L01 ;CONTINUE LOOP
TO1L04: CMP #200, @BECR1 ;ALL BITS IN BECR1 0 EXCEPT RDY?
BNE TO1L03 ;BRANCH TO ERROR IF NO
MOV BECR2, RO ;INDICATE LOOKING AT BECR2
CLR @BERE ;RESET ERROR CONDITIONS
CLR @BECR2 ;CLEAR BECR2
BIT #157777, @BECR2 ;IS BECR2 =0 EXECPT CCOVF?
BNE TO1L03 ;NO, TYPE ADDRESS AND DATA ERROR
MOV #6, @#4 ;RESTORE TRAP CATCHER
CLR @#6
BR TST2 ;;GO TO NEXT TEST
STRAP: MOV (SP), $REGO ;SAVE PC FROM STACK
ERROR ID3 ;FATAL ERROR:CPU DID NOT RECEIVE Ssyn
JSR PC, TERRPC ;TYPE PC OF ERROR MSG
TO1L05: MOV #6, @#4 ;RESTORE TRAP CATCHER
CLR @#6
JMP NUBE1 ;TEST NEXT UBE
;*****
;TEST 2 TEST BITS 1-6,8-14 OF BECR1 AND BITS 0-3,14 OF BECR2 CHANGE
;
;R2, R3 CONTAIN THE TRUE AND COMPLEMENT TEST DATA
;R4 CONTAINS A POINTER TO THE REG ADDRESS BEING TESTED
;R5 CONTAINS THE MASKED CONTENTS OF THE REG BEING TESTED
;$TMP1 CONTAINS THE MASK FOR THE REG
;
;*IF THIS TEST FAILS, ALL FOLLOWING TESTS FOR THIS MODULE ARE ABORTED
;*****

```

H03

UNIBUS EXERCISOR MODULE DIAGNOSTIC MACY11 27(732) 17-SEP-76 15:49 PAGE 31
 DZKUBA.P11 T2 TEST BITS 1-6,8-14 OF BECR1 AND BITS 0-3,14 OF BECR2 CHANGE

1365	004052	000004							
1366	004054	012706	001100		TST2:	SCOPE			
1367	004060	012737	000340	177776		MOV #STACK, SP			;RESTORE STACK
1369	004066	012702	052652			MOV #340, #PSW			;LOCK OUT INTERRUPTS
1369	004072	012703	025324			MOV #52652, R2			;SETUP TEST DATA BECR1
1370	004076	012704	002524			MOV #25324, R3			;SETUP COMP. TEST DATA BECR1
1371	004102	005077	176422			MOV #BECR1, R4			;LOAD ADDRESS PTER. FOR BECR1
1372	004106	012767	177777	175112		CLR #BERE			;CLEAR ERROR CONDITIONS
1373	004114	016705	175106		T02L03:	MOV #177777, \$TMP1			;LOAD MASK TO LOOK AT ALL BECR1
1374	004120	011400				MOV \$TMP1, R5			;LOAD R5 WITH MASK
1375	004122	010210				MOV (R4), R0			;GET ADDRESS OF BECR TESTING
1376	004124	011001				MOV R2, (R0)			;LOAD BECR WITH DATA
1377	004126	005101				MOV (R0), R1			;GET CONTENTS OF BECR
1378	004130	040105				COM R1			;ONLY LOOK AT BITS
1379	004132	020502				BIC R1, R5			;SET IN MASK =R5
1380	004134	001424				CMP R5, R2			;DATA OK?
1381	004136	011467	175052		T02L02:	BEQ T02L01			;BRANCH IF YES
1382	004142	011067	175050			MOV (R4), \$REG0			;SAVE BECR ADDRESS
1383	004146	010267	175046			MOV (R0), \$REG1			;SAVE BECR BAD DATA
1384	004152	104006				MOV R2, \$REG2			;SAVE GOOD DATA
1385	004154	021467	176346			ERROR #D6			;FATAL ERROR: CONTROL REG HELD WRONG DATA
1386	004160	001006				CMP (R4), BECR2			;DID BECR2 FAIL?
1387	004162	032777	020000	176336		BNE T02L04			;BRANCH IF NO
1388	004170	001402				BIT #20000, #BECR2			;WAS CCOVF=1?
1389	004172	104401	027464			BEQ T02L04			;BRANCH IF NO
1390	004176	004767	012132		T02L04:	TYPE MSG14			;DISREGARD BIT 13=1 OF BECR2
1391	004202	000167	010164			JSR PC, TERRPC			;TYPE PC OF ERROR MSG
1392	004206	010302			T02L01:	JMP NUBE			;TEST NEXT UBE
1393	004210	010210				MOV R3, R2			;XFER NEW TEST DATA
1394	004212	011001				MOV R2, (R0)			;LOAD BECR WITH COMP.DATA
1395	004214	016705	175006			MOV (R0), R1			;GET CONTENTS OF BECR
1396	004220	005101				MOV \$TMP1, R5			;LOAD R5 WITH MASK
1397	004222	040105				COM R1			;ONLY LOOK AT BITS
1398	004224	020502				BIC R1, R5			;SET IN MASK =R5
1399	004226	001343				CMP R5, R2			;DATA OK?
1400	004230	012702	040012			BNE T02L02			;BRANCH IF NO
1401	004234	012703	000005			MOV #40012, R2			;SETUP TEST DATA BECR2
1402	004240	012704	002526			MOV #5, R3			;SETUP COMP. TEST DATA BECR2
1403	004244	012767	157777	174754		MOV #BECR2, R4			;LOAD ADDRESS PTER. FOR BECR2
1404	004252	020067	176250			MOV #157777, \$TMP1			;HAVE MASK LOOK AT ALL BECR2 EXECPT CCOVF
1405	004256	001316				CMP R0, BECR2			;TESTED BECR2?
1406						BNE T02L03			;NO, BRANCH TO START TEST OF BECR2
1407									
1408						:*****			
1409						:TEST 3 FLOAT A '1' THROUGH BEBD, BECC, BEBA			
1410						:*			
1411						:R0 CONTAINS A POINTER TO THE REG ADDRESS BEING TESTED			
1412						:R1 CONTAINS TEST DATA			
1413						:*			
1414						:IF THIS TEST FAILS, ALL FOLLOWING TESTS FOR THIS MODULE ARE ABORTED			
1415						:*****			
1415	004260	000004			TST3:	SCOPE			
1416	004262	012706	001100			MOV #STACK, SP			;RESTORE STACK
1417	004266	012737	000340	177776		MOV #340, #PSW			;LOCK OUT INTERRUPTS
1418	004274	012700	002516			MOV #BEBD, R0			;GET BEBD ADDRESS PTER.
1419	004300	012701	000001		T03L04:	MOV #1, R1			;SETUP TEST DATA REG
1420	004304	010130			T03L03:	MOV R1, #R0+			;PUT TEST DATA IN REG

1421	004306	025001			CMP 2-(RO),R1	;TEST REG
1422	004310	001413			BEQ T03L01	;BRANCH IF OK
1423	004312	011067	174676		MOV (RO), \$REGO	;SAVE FAILING REG ADDRESS
1424	004316	010167	174676		MOV R1, \$REG2	;SAVE GOOD DATA
1425	004322	013067	174670		MOV 2(RO)+, \$REG1	;SAVE BAD DATA
1426	004326	104004			ERROR 1D4	;FATAL ERROR:REG FAILED TO FLOAT A '1'
1427	004330	004767	012000		JSR PC, TERRPC	;TYPE PC OF ERROR MSG
1428	004334	000167	010032		JMP NUBE	;TEST NEXT UBE
1429	004340	005701		T03L01:	TST R1	;TESTED ALL 16 BITS?
1430	004342	100402			BMI T03L02	;BRANCH IF YES
1431	004344	006301			ASL R1	;TEST NEXT BIT
1432	004346	000756			BR T03L03	;CONTINUE LOOP
1433	004350	022067	176146	T03L02:	CMP (RO)+, BEBA	;TESTED LAST REG? ALSO UPDATE ADDR. PTER.
1434	004354	001351			BNE T03L04	;BRANCH IF REGS NOT TESTED

```

*****
;TEST 4      FLOAT A '0' THROUGH BEBD, BECC, BEBA
;
;RO 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
*****

```

1444	004356	000004			TST4: SCOPE	
1445	004360	012706	001100		MOV #STACK, SP	;RESTORE STACK
1446	004364	012737	000340	177776	MOV #340, 2#PSW	;LOCK OUT INTERRUPTS
1447	004372	012700	002516		MOV #BEBD, RO	;GET BEBD ADDRESS PTER.
1448	004376	012701	177776		T04L04: MOV #177776, R1	;SETUP TEST DATA REG
1449	004402	010130			T04L03: MOV R1, 2(RO)+	;PUT TEST DATA IN REG
1450	004404	025001			CMP 2-(RO), R1	;TEST REG
1451	004406	001413			BEQ T04L01	;BRANCH IF OK
1452	004410	011067	174600		MOV (RO), \$REGO	;SAVE FAILING REG ADDRESS
1453	004414	010167	174600		MOV R1, \$REG2	;SAVE GOOD DATA
1454	004420	013067	174572		MOV 2(RO)+, \$REG1	;SAVE BAD DATA
1455	004424	104005			ERROR 1D5	;FATAL ERROR: REG FAILED TO FLOAT A '0'
1456	004426	004767	011702		JSR PC, TERRPC	;TYPE PC OF ERROR MSG
1457	004432	000167	007734		JMP NUBE	;TEST NEXT UBE
1458	004436	005701		T04L01:	TST R1	;TESTED ALL 16 BITS?
1459	004440	100002			BPL T04L02	;BRANCH IF YES
1460	004442	006001			ROR R1	;TEST NEXT BIT
1461	004444	000756			BR T04L03	;CONTINUE LOOP
1462	004446	022067	176050	T04L02:	CMP(RO)+, BEBA	;TESTED LAST REG? ALSO UPDATE ADDR. PTER.
1463	004452	001351			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.
;RO 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
*****

```

1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476

```

1477 004454 000004          TSTS:  SCOPE
1478 004456 012706 001100      MOV #STACK,SP          ;RESTORE STACK
1479 004462 012737 000340 177776  MOV #340,#PSW         ;LOCK OUT INTERRUPTS
1480 004470 004767 011440      JSR PC,CLRREG        ;CLEAR ALL REG
1481 004474 016700 176016      MOV BEB0,R0         ;INITIALIZE TEST ADDRESS
1482 004500 016701 176012      TOSL04: MOV BEB0,R1   ;INITIALIZE PTER.
1483 004504 012710 000002      MOV #2,(R0)        ;LOAD TEST REG
1484 004510 012702 177777      MOV #177777,R2     ;INITIALIZE MASK TO LOOK AT ALL BITS
1485 004514 030211          TOSL03: BIT R2,(R1)  ;IS DATA IN REG =0?
1486 004516 001422          BEQ TOSL01         ;BRANCH IF DATA OK(=0)
1487 004520 020100          CMP R1,R0         ;LOOKING AT REG LOADED?
1488 004522 001420          BEQ TOSL01         ;BRANCH IF YES (DATA OK)
1489 004524 020167 175774      CMP R1,BECR1       ;LOOKING AT BECR1?
1490 004530 001411          BEQ TOSL07        ;BRANCH IF YES
1491 004532 010067 174456      TOSL08: MOV R0,$REG0 ;ERROR: SAVE REG ADDRESS LOADED
1492 004536 010167 174454      MOV R1,$REG1       ;SAVE REG ADDRESS EXAMINED
1493 004542 104007          ERROR #D7         ;FATAL ERROR: DUAL ADDRESSING ERROR
1494 004544 004767 011564      JSR PC,TERRPC      ;TYPE PC OF ERROR MSG
1495 004550 000167 007616      JMP NJBE           ;TEST NEXT UBE
1496 004554 022777 000200 175742 TOSL07: CMP #200,#BECR1 ;ALL BITS IN BECR1 0 EXCEPT RDY?
1497 004562 001363          BNE TOSL08        ;BRANCH IF NO
1498 004564 020167 175736      TOSL01: CMP R1,BECR2 ;LOOKED AT BECR2?
1499 004570 001412          BEQ TOSL02        ;BRANCH IF YES
1500 004572 020167 175726      CMP R1,BECR1       ;PTER UP TO BECR1?
1501 004576 001005          BNE TOSL06        ;NO, LOOK AT NEXT REG
1502 004600 016701 175722      MOV BECR2,R1       ;NOW LOOK AT BECR2
1503 004604 012702 157777      MOV #157777,R2     ;LOOK AT ALL BECR2 EXCEPT CCOVF
1504 004610 000741          BR TOSL03         ;CONTINUE LOOKING
1505 004612 005721          TOSL06: TST (R1)+  ;UPDATE PTER.
1506 004614 000737          BR TOSL03         ;LOOK AT NEXT REG.
1507 004616 004767 011312      TOSL02: JSR PC,CLRREG ;CLEAR ALL REG
1508 004622 020067 175700      CMP R0,BECR2       ;LOADED AND TESTED BECR2?
1509 004626 001410          BEQ TST6          ;BRANCH IF YES TO NEXT TEST
1510 004630 020067 175670      CMP R0,BECR1       ;LOADED BECR1 WITH DATA YET?
1511 004634 001003          BNE TOSL05        ;BRANCH IF NO
1512 004636 016700 175664      MOV BECR2,R0       ;YES, NOW LOAD BECR2
1513 004642 000716          BR TOSL04         ;CONTINUE LOOKING
1514 004644 005720          TOSL05: TST(R0)+  ;UPDATE ADDRESS OF REG LOADED
1515 004646 000714          BR TOSL04         ;TEST THIS REG

1516
1517
1518 :*****
1519 :*TEST 6          TEST BUS PARITY BIT PB
1520 :*
1521 :*THIS TEST IS NOT RUN ON THOSE MACHINE
1522 :*WITH NO PARITY TRAP (11/05, 11/20)
1523 :*****
1523 004650 000004          TST6:  SCOPE
1524 004652 012706 001100      MOV #STACK,SP          ;RESTORE STACK

1525
1526 :////////////////////
1527 :ROUTINE TO DETERMINE IF RUNNING UNDER 11/05 OR 11/20
1528 :IF 11/05 OR 11/20 BUSS PARITY TEST IS SKIPPED
1529 :////////////////////
1530 004656 012737 004752 000010      MOV #ITRAP,#10       ;SET UP TO GO TO NEXT TEST IF ILLEGAL INST TRAP
1531 004664 012737 000340 000012      MOV #340,#12
1532 004672 006700          SXT R0              ;IF INST TRAPS HAVE 11/05 OR 11/20

```



```

1533
1534 004674 012737 000340 177776      MOV #340, @PSW      ;SET PSW PRIORITY=7
1535 004702 012737 004736 000114      MOV #PTRAP, @#114  ;SET UP PARITY TRAP
1536 004710 012737 000340 000116      MOV #340, @#116
1537 004716 012777 010000 175602      MOV #10000, @BECR2 ;ENABLE PB PARITY
1538 004724 005777 175576      TST @BECR2         ;START PARITY TRAP
1539 004730 104010      ERROR #D8         ;SETTING PB PARITY FAILED TO CAUSE CPU TO TRAP
1540 004732 004767 011376      JSR PC, TERRPC    ;TYPE PC OF ERROR MSG
1541 004736 012737 000116 000114 PTRAP: MOV #116, @#114    ;RESTORE TRAP CATCHER
1542 004744 005037 000116      CLR @#116         ;RESTORE TRAP CATCHER
1543 004750 000411      BR T06L01        ;SKIP MSG
1544 004752 032737 010000 001172 ITRAP: BIT #SW12, @SWR ;INHIBIT TYPEOUTS?
1545 004760 001005      BNE T06L01       ;BRANCH IF YES
1546 004762 012767 000001 174244      MOV #1, $TIMES    ;DO 1 ITERATION WHEN TEST NOT NOT RUN
1547 004770 104401 020511      TYPE MSG5        ;BUS PARITY NOT TESTED ON 11/05 OR 11/20 MACHINES
1548 004774 012737 000012 000010 T06L01: MOV #12, @#10   ;RESTORE TRAP CATCHER
1549 005002 005037 000012      CLR @#12         ;RESTORE TRAP CATCHER
1550 005006 012777 000000 175512      MOV #0, @BECR2   ;DO DATO TO CLEAR PB BIT

```

```

1551
1552 ;*****
1553 ;*TEST 7 TEST GO WORKS, RDY SETS AND CLEARS, AND THE RELEASE BUS IMMEDIATE WORKS
1554 ;*
1555 ;*THE READY AND GO BIT ARE CHECKED USING A RELEASE
1556 ;*BUSS IMMEDIATE FUNCTION. FALSE INTERRUPT ARE CHECKED FOR
1557 ;*
1558 ;*IF THE GO OR READY BITS FAIL, ALL FOLLOWING TESTS FOR THIS MODULE ARE ABORTED.
1559 ;*****

```

```

1560 005014 000004      TST7: SCOPE
1561 005016 012706 001100      MOV #STACK, SP    ;RESTORE STACK
1562 005022 012737 000340 177776      MOV #340, @PSW   ;LOCK OUT INTERRUPTS
1563 005030 004767 011100      JSR PC, CLRREG    ;CLR ALL REG
1564 005034 012777 005154 175470      MOV #FINT1, @INTVEC ;SET UP FOR FALSE INTERRUPT
1565 005042 016700 175464      MOV INTVEC, RO    ;GET INTERRUPT VECTOR
1566 005046 012760 000340 000002      MOV #340, 2(RO)  ;SET PSW PRIORITY=7
1567 005054 012777 006003 175442      MOV #6003, @BECR1 ;SET GO BIT AND DO RELEASE BUSS IMMEDIATE WITH BR4=1
1568 005062 032777 000200 175434      BIT #200, @BECR1 ;LOOK AT RDY BIT
1569 005070 001035      BNE T07L08       ;BRANCH IF NOT CLEARED
1570 005072 005037 177776      CLR @PSW         ;ALLOW INTERRUPTS
1571 005076 005000      T07L07: CLR RO    ;INITIALIZE A COUNT TO WAIT FOR RDY=1
1572 005100 005200      T07L03: INC RO    ;UPDATE COUNT AND LOOP
1573 005102 022700 000011      CMP #11, RO      ;TILL COUNT=10 OR RDY=1
1574 005106 001416      BEQ T07L04       ;BRANCH IF RDY WAS NOT SET
1575 005110 105777 175410      TSTB @BECR1     ;READY SET?
1576 005114 100371      BPL T07L03      ;CONTINUE TO LOOK FOR RDY
1577 005116 032777 000001 175400      BIT #1, @BECR1  ;SEE IF GO BIT CLEARED
1578 005124 001426      BEQ T07L05      ;PROCEED TO NEXT TEST IF YES
1579 005126 004767 011034      JSR PC, RCATCH  ;RESTORE TRAP CATCHER
1580 005132 104013      ERROR #D11     ;FATAL ERROR: GO BIT FAILED TO CLEAR
1581 005134 004767 011174      JSR PC, TERRPC ;TYPE PC OF ERROR MSG
1582 005140 000167 007226      JMP NUBE       ;TEST NEXT UBE
1583 005144 104014      T07L04: ERROR #D12 ;FATAL ERROR: RDY BIT FAILED TO SET
1584 005146 004767 011162      JSR PC, TERRPC ;TYPE PC OF ERROR MSG
1585 005152 000407      BR T07L06     ;ABORT UBE TEST
1586
1587 005154 104123      FINT1: ERROR #D83 ;ERROR: FALSE INTERRUPT WHEN DO RELEASE BUSS IMMED.
1588 005156 004767 011152      JSR PC, TERRPC ;TYPE PC OF ERROR MSG

```

```

1589 005162 000745
1590
1591 005164 104020
1592 005166 004767 011142
1593 005172 004767 010770
1594 005176 000167 007170
1595 005202 004767 010760
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617 005206 000004
1618 005210 012706 001100
1619 005214 012737 000340 177776
1620 005222 004767 010706
1621 005226 010667 173772
1622 005232 005000
1623 005234 012046
1624 005236 022700 000060
1625 005242 001374
1626 005244 012737 000341 000002
1627 005252 012700 000004
1628 005256 012720 005670
1629 005262 012720 000341
1630 005266 022700 001000
1631 005272 001371
1632 005274 012777 005552 175230
1633 005302 012767 000004 173704
1634 005310 012767 000200 173700
1635 005316 012777 000003 175200
1636 005324 012737 000200 177776
1637 005332 000240
1638 005334 012767 000005 173652
1639 005342 012767 000240 173646
1640 005350 012737 000240 177776
1641 005356 012777 000005 175140
1642 005364 000240
1643 005366 012767 000006 173620
1644 005374 012767 000300 173614

```

```

BR T07L07 ;NOW CHECK IF RDY=1 AND GO BIT=0.
T07L08: ERROR ↑D16 ;FATAL ERROR: RDY BIT FAILED TO CLEAR OR GO DID NOT SET
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
T07L06: JSR PC,RCATCH ;RESTORE TRAP CATCHER
JMP NU8E ;TEST NEXT UBE
T07L05: JSR PC,RCATCH ;RESTORE TRAP CATCHER

*****
*TEST 10 TEST UBE CAN INTERRUPT ON ALL LEVELS & THE NO INT. SSWN BIT DOESN'T 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.
*****
TST10: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,#PSW ;LOCK OUT INTERRUPTS
JSR PC,CLRREG ;CLEAR ALL UBE REG
MOV SP,#TMPD ;SAVE STACK ADDRESS
CLR RO ;INIT. RO
T08L08: MOV (RO)+, -(SP) ;SAVE VECTOR AREA 0-56
CMP #60,RO ;ALL SAVED?
BNE T08L08 ;BRANCH IF NO
MOV #341,#2 ;SET UP VECTOR AREA TO DETECT WRONG INT. VECTORS
MOV #4,RO ;INITIALIZE ADDRESS REG
T08L01: MOV #WINT,(RO)+ ;PUT WRONG INTERRUPT PTER IN ALL VECTOR LOCATIONS
MOV #341,(RO)+ ;PUT AN ODD PSW IN ALL PSW LOCATIONS
CMP #1000,RO ;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

```


M03

UNIBUS EXERCISOR MODULE DIAGNOSTIC MACY11 27(732) 17-SEP-76 15:49 PAGE 36
 DZKUBA.P11 T10 TEST UBE CAN INTERRUPT ON ALL LEVELS & THE NO INT. Ssyn BIT DOESN'T SET

1645	005402	012737	000300	177776	MOV #300, @#PSW	;SET PRIORITY=6
1646	005410	012777	000011	175106	MOV #11, @BECR1	;HAVE UBE DO BR=6
1647	005416	000240			NOP	;UBE SHOULD NOT INTERRUPT HERE
1648						
1649						
1650						;NOW TEST UBE WILL INTERRUPT WITH PRIORITY ONE LEVEL BELOW BR
1651	005420	012777	000002	175076	MOV #2, @BECR1	;INITIALIZE UBE TO DO BR=4
1652	005426	012767	000004	173560	MOV #4, \$REG0	;INITIALIZE INDICATOR FOR BR=4
1653	005434	012767	000003	173554	MOV #3, \$REG1	;INITIALIZE INDICATOR FOR PRIORITY=3
1654	005442	012777	005524	175062	MOV #T08L02, @INTVEC	;SET RETURN ADDRESS WHEN GET PROPER INTERRUPT
1655	005450	012737	000140	177776	MOV #140, @#PSW	;INITIALIZE PSW PRIORITY=3
1656	005456	000240			NOP	;UBE SHOULD INTERRUPT HERE
1657	005460	000413			BR T08L09	;BRANCH TO ERROR IF NO INT.
1658	005462	005267	173526		T08L03: INC \$REG0	;INDICATE BR LEVEL DOING
1659	005466	005267	173524		INC \$REG1	;INDICATE PSW PRIORITY LEVEL DOING
1660	005472	000257			CCC	;CLEAR N,Z,V,C
1661	005474	062737	000040	177776	ADD #40, @#PSW	;SET PRIORITY LEVEL BELOW BR LEVEL
1662	005502	005277	175016		INC @BECR1	;HAVE UBE DO BR 1 LEVEL ABOVE PRIORITY
1663	005506	000240			NOP	;UBE SHOULD INTERRUPT HERE
1664	005510	004767	010544		T08L09: JSR PC, RVEC	;RESTORE TRAP CATCHER AND HANDLER
1665	005514	104021			ERROR !D17	;ERROR: UBE FAILED TO INTERRUPT
1666	005516	004767	010612		JSR PC, TERRPC	;TYPE PC OF ERROR MSG
1667	005522	000472			BR T08L06	;BRANCH TO TEST NO INT. Ssyn ERROR BIT
1668	005524	022626			T08L02: CMP (SP)+, (SP)+	;RESTORE STACK AFTER INTERRUPT
1669	005526	032777	000020	174770	BIT #20, @BECR1	;TESTED LAST BR?
1670	005534	001063			BNE T08L07	;BRANCH IF YES TO TEST NO INT. Ssyn ERROR BIT
1671	005536	006377	174762		ASL @BECR1	;SHIFT BECR1 FOR NEXT BR LEVEL
1672	005542	042777	000400	174754	BIC #400, @BECR1	;CLEAR SHIFTED Rdy BIT
1673	005550	000744			BR T08L03	;GO TEST NEXT BR
1674						
1675	005552	022626			FINT3: CMP (SP)+, (SP)+	;RESTORE STACK AFTER INTERRUPT
1676	005554	004767	010500		JSR PC, RVEC	;RESTORE VECTOR AREA
1677	005560	104022			ERROR !D18	;ERROR: UBE INT. WHEN PSW AT SAME PRIORITY LEVEL
1678	005562	004767	010546		JSR PC, TERRPC	;TYPE PC OF ERROR MSG
1679	005566	032777	007740	174732	BIT #7740, @BECR2	;SEE IF ERROR CONDITION OCCURRED IN BECR2
1680	005574	001407			BEG T08L04	;BRANCH IF NO
1681	005576	017767	174724	173410	MOV @BECR2, \$REG0	;SAVE ERROR CONDITIONS
1682	005604	104017			ERROR !D15	;ERROR: ERROR BITS IN BECR2 SET WHEN SHOULD=0
1683	005606	004767	010522		JSR PC, TERRPC	;TYPE PC OF ERROR MSG
1684	005612	000445			BR TST11	;BRANCH TO NEXT TEST
1685						
1686	005614	012777	005622	174710	T08L04: MOV #T08L05, @INTVEC	;SET UP INTVEC TO FIND BR LEVEL UBE MADE
1687	005622	012706	001100		T08L05: MOV #STACK, SP	;RESTORE STACK
1688	005626	062767	000040	173362	ADD #40, \$REG1	;RAISE PRIORITY LEVEL BY 1
1689	005634	005267	173354		INC \$REG0	;INDICATE NEW LEVEL OF PRIORITY
1690	005640	016737	173352	177776	MOV \$REG1, @#PSW	;SET PSW PRIORITY
1691	005646	005277	174652		INC @BECR1	;HAVE UBE INTERRUPT AGAIN
1692	005652	000240			NOP	;IF UBE INT. HERE, INCREMENT PRIORITY
1693	005654	004767	010306		JSR PC, RCATCH	;RESTORE TRAP CATCHER
1694	005660	104023			ERROR !D19	;ERROR: UBE FALSELY INTERRUPTED AT HIGHER LEVEL
1695	005662	004767	010446		JSR PC, TERRPC	;TYPE PC OF ERROR MSG
1696	005666	000417			BR TST11	;BRANCH TO NEXT TEST
1697						
1698	005670	022626			WINT: CMP (SP)+, (SP)+	;RESTORE STACK AFTER INTERRUPT
1699	005672	004767	010362		JSR PC, RVEC	;RESTORE VECTOR AREA
1700	005676	104024			ERROR !D20	;ERROR: UBE INTERRUPTED TO WRONG VECTOR

NO3

UNIBUS EXERCISOR MODULE DIAGNOSTIC MACY11 27(732) 17-SEP-76 15:49 PAGE 37
 DZKUBA.P11 T10 TEST UBE CAN INTERRUPT ON ALL LEVELS & THE NO INT. SSYN BIT DOESN'T SET

1701	005700	004767	010430			JSR PC, TERRPC	;TYPE PC OF ERROR MSG
1702	005704	004767	010350			TOBL07: JSR PC, RVEC	;RETURN VECTOR AREA WHEN FINISH BR TEST
1703	005710	032777	004000	174610	TOBL06:	BIT #4000, @BECR2	;WAS NO INT. SSYN ERROR BIT SET?
1704	005716	001403				BEQ TST11	;BRANCH TO NEXT TEST IF NO
1705	005720	104027				ERROR ↑D23	;ERROR: NO INT. SSYN BIT FALSELY SET
1706	005722	004767	010406			JSR PC, TERRPC	;TYPE PC OF ERROR MSG
1707							
1708							
1709							
1710							
1711							
1712							
1713							
1714							
1715	005726	000004					
1716	005730	012706	001100			TST11: SCOPE	
1717	005734	012737	000340	177776		MOV #STACK, SP	;RESTORE STACK
1718	005742	004767	010166			MOV #340, @#PSW	;LOCK OUT INTERRUPTS
1719	005746	012777	000010	174552		JSR PC, CLRREG	;CLEAR ALL UBE REGS.
1720	005754	012777	006003	174542		MOV #10, @BECR2	;ENABLE INH SACK IN BECR2
1721	005762	005037	177776			MOV #6003, @BECR1	;DO FUN 3 VIA BR4
1722	005766	000240				CLR @#PSW	;ALLOW INTERRUPTS
1723						NOP	;ALLOW UBE TO GET BUSS. CPU SHOULD TIME OUT
1724	005770	005000					
1725	005772	005200			1\$:	CLR R0	;INIT COUNTER
1726	005774	105700				INC R0	;INC COUNTER
1727	005776	100375				TSTB R0	;WAITED >20 USEC?
1728	006000	032777	000200	174520		BPL 1\$;BRANCH IF NO
1729	006006	001403				BIT #200, @BECR2	;WAS NO, NO SACK BIT SET?
1730	006010	104026				BEQ RTR	;BRANCH IF NO
1731	006012	004767	010316			ERROR ↑D22	;ERROR: NO, NO SACK BIT FALSELY SET
1732	006016	004767	010112		RTR:	JSR PC, TERRPC	;TYPE PC OF ERROR MSG
1733						JSR PC, CLRREG	;CLEAR ALL UBE REG
1734							
1735							
1736							
1737							
1738							
1739							
1740							
1741							
1742							
1743	006022	000004					
1744	006024	012706	001100			TST12: SCOPE	
1745	006030	012737	000340	177776		MOV #STACK, SP	;RESTORE STACK
1746	006036	012767	052525	021634		MOV #340, @#PSW	;LOCK OUT INTERRUPTS
1747	006044	004767	010064			MOV #052525, BUFF1	;PUT TEST DATA IN BUFFER
1748	006050	012777	177777	174442		JSR PC, CLRREG	;CLEAR ALL UBE REG
1749	006056	012777	027700	174436		MOV #177777, @BECC	;HAVE UBE DO 1 XFER
1750	006064	012705	006572			MOV #BUFF1, @BEBA	;LOAD UBE WITH BUFFER ADDRESS
1751	006070	012777	002003	174426		MOV #ERR1, R5	;INITIALIZE R5 FOR ERROR ADDRESS
1752	006076	005037	177776			MOV #2003, @BECR1	;HAVE UBE DO DATI VIA BR=4 AND FUNCTION 1
1753	006102	004767	000434			CLR @#PSW	;ALLOW DATA XFER
1754	006106	022777	052525	174402		JSR PC, RDYS	;GO CHECK FOR RDY TO SET
1755	006114	001421				CMP #052525, @BEBO	;IS DATA OK?
1756	006116	017767	174374	173070		BEQ T10L01	;GO TEST DATO IF YES
						MOV @BEBO, \$REGO	;SAVE (BEBO)

1757	006124	016767	021550	173064	MOV	BUFF1,\$REG1	:SAVE MEM DATA
1758	006132	012767	027700	173060	MOV	#BUFF1,\$REG2	:SAVE MEM ADDRESS
1759	006140	012767	052525	173054	MOV	#52525,\$REG3	:SAVE CORRECT DATA
1760	006146	104030			ERROR	↑D24	:ERROR: DATI FAILED TO LOAD PROPER DATA
1761	006150	004767	010160		JSR	PC,TERRPC	:TYPE PC OF ERROR MSG
1762	006154	000167	000450		JMP	TST1A	:GO TO NEXT TEST
1763	006160	004767	007750		T10L01: JSR	PC,CLRREG	:CLEAR UBE REG
1764	006164	005067	021510		CLR	BUFF1	:CLEAR TEST AREA
1765	006170	012777	177777	174322	MOV	#177777,UBECC	:HAVE UBE DO 1 XFER
1766	006176	012777	027700	174316	MOV	#BUFF1,UBEBA	:LOAD UBE WITH BUFFER ADDRESS
1767	006204	012777	052525	174304	MOV	#052525,UBEED	:LOAD UBE WITH DATA
1768	006212	012705	006602		MOV	#ERR2,R5	:INITIALIZE R5 FOR ERROR ADDRESS
1769	006216	012777	003003	174300	MOV	#3003,UBEER1	:HAVE UBE DO DATO VIA BR=4 AND FUNCTION 1
1770	006224	004767	000312		JSR	PC,RDYS	:GO CHECK FOR RDY TO SET
1771	006230	022767	052525	021442	CMP	#052525,BUFF1	:WAS BUFFER LOADED PROPERLY?
1772	006236	001420			BEQ	T10L02	:GO TEST DATIP IF YES
1773	006240	017767	174252	172746	MOV	UBEED,\$REG0	:SAVE (BEED)
1774	006246	016767	021426	172742	MOV	BUFF1,\$REG1	:SAVE MEM DATA
1775	006254	012767	027700	172736	MOV	#BUFF1,\$REG2	:SAVE MEM ADDRESS
1776	006262	012767	052525	172732	MOV	#052525,\$REG3	:SAVE CORRECT DATA
1777	006270	104031			ERROR	↑D25	:ERROR: DATO FAILED TO LOAD PROPER DATA
1778	006272	004767	010036		JSR	PC,TERRPC	:TYPE PC OF ERROR MSG
1779	006276	000554			BR	TST13	:BRANCH TO NEXT TEST
1780							
1781	006300	004767	007630		T10L02: JSR	PC,CLRREG	:CLEAR UBE REG
1782	006304	012767	052525	021366	MOV	#052525,BUFF1	:PUT TEST DATA IN BUFFER
1783	006312	012777	177777	174200	MOV	#177777,UBECC	:HAVE UBE DO 1 XFER
1784	006320	012777	027700	174174	MOV	#BUFF1,UBEBA	:LOAD UBE WITH BUFFER ADDRESS
1785	006326	012705	006612		MOV	#ERR3,R5	:INITIALIZE R5 FOR ERROR ADDRESS
1786	006332	012777	002403	174164	MOV	#2403,UBEER1	:HAVE UBE DO DATIP VIA BR=4 AND FUNCTION 1
1787	006340	004767	000176		JSR	PC,RDYS	:GO CHECK FOR RDY SET
1788	006344	022777	125252	174144	CMP	#125252,UBEED	:HAS UBE SHIFTED DATA?
1789	006352	001004			BNE	T10L06	:BRANCH IF NO
1790	006354	022767	125252	021316	CMP	#125252,BUFF1	:HAS MEM LOC BEEN SHIFTED?
1791	006362	001420			BEQ	T10L03	:GO TEST DATOB IF YES
1792	006364	017767	174126	172622	T10L06: MOV	UBEED,\$REG0	:SAVE (BEED)
1793	006372	016767	021302	172616	MOV	BUFF1,\$REG1	:SAVE MEM DATA
1794	006400	012767	027700	172612	MOV	#BUFF1,\$REG2	:SAVE MEM ADDRESS
1795	006406	012767	125252	172606	MOV	#125252,\$REG3	:SAVE CORRECT DATA
1796	006414	104032			ERROR	↑D26	:ERROR: DATIP FAILED TO LOAD PROPER DATA
1797	006416	004767	007712		JSR	PC,TERRPC	:TYPE PC OF ERROR MSG
1798	006422	000502			BR	TST13	:BRANCH TO NEXT TEST
1799							
1800	006424	012767	000377	021246	T10L03: MOV	#377,BUFF1	:INITIALIZE BUFFER
1801	006432	012705	006622		MOV	#ERR4,R5	:INITIALIZE R5 FOR ERROR ADDRESS
1802	006436	012777	177400	174052	MOV	#177400,UBEED	:LOAD HIGH BYTE OF UBE WITH 1'S
1803	006444	012777	027701	174050	MOV	#BUFF1+1,UBEBA	:LOAD HIGH BYTE BUFF ADDR. INTO UBE
1804	006452	012777	177777	174040	MOV	#177777,UBECC	:HAVE UBE DO 1 XFER
1805	006460	012777	003403	174036	MOV	#3403,UBEER1	:HAVE UBE DO DATOB VIA BR=4 AND FUNCTION 1
1806	006466	004767	000050		JSR	PC,RDYS	:GO CHECK FOR RDY SET
1807	006472	022767	177777	021200	CMP	#177777,BUFF1	:TEST IF DATOB DONE CORRECTLY
1808	006500	001453			BEQ	TST13	:BRANCH IF YES TO NEXT TEST
1809							
1810	006502	017767	174010	172504	MOV	UBEED,\$REG0	:SAVE (BEED)
1811	006510	016767	021164	172500	MOV	BUFF1,\$REG1	:SAVE NEW DATA
1812	006516	012767	027700	172474	MOV	#BUFF1,\$REG2	:SAVE MEM ADDRESS


```

1813 006524 012767 177777 172470      MOV #177777,$REG3      ;SAVE CORRECT DATA
1814 006532 104033      ERROR #D27            ;ERROR: DATOB FAILED TO LOAD DATA PROPERLY
1815 006534 004767 007574      JSR PC,TERRPC        ;TYPE PC OF ENROR MSG
1816 006540 000433      BR TST13             ;:BRANCH TO NEXT TEST
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
006542 005004      RDYS: CLR R4          ;INITIALIZE R4
006544 032777 000200 173752 T10L05: BIT #200,$BECR1 ;IS RDY SET?
006552 001006      BNE T10L04          ;BRANCH IF YES
006554 005204      INC R4              ;UPDATE COUNT
006556 032704 000020      BIT #20,R4         ;COUNT=16?
006562 001770      BEQ T10L05         ;IF NO, GO TEST RDY AGAIN
006564 005726      TST (SP)+          ;RETURN STACK PTER
006566 000115      JMP (R5)           ;GO INDICATE ERROR
006570 000207      T10L04: RTS PC      ;RETURN AND CHECK DATA
006572 104034      ERR1: ERROR #D28   ;ERROR: DATI FAILED TO SET RDY
006574 004767 007534      JSR PC,TERRPC      ;TYPE PC OF ERROR MSG
006600 000413      BR TST13           ;:GO TO NEXT TEST
006602 104035      ERR2: ERROR #D29   ;ERROR: DATO FAILED TO SET RDY
006604 004767 007524      JSR PC,TERRPC      ;TYPE PC OF ERROR MSG
006610 000407      BR TST13           ;:GO TO NEXT TEST
006612 104036      ERR3: ERROR #D30   ;ERROR: DATIP FAILED TO SET RDY
006614 004767 007514      JSR PC,TERRPC      ;TYPE PC OF ERROR MSG
006620 000403      BR TST13           ;:GO TO NEXT TEST
006622 104037      ERR4: ERROR #D31   ;ERROR: DATOB FAILED TO SET RDY
006624 004767 007504      JSR PC,TERRPC      ;TYPE PC OF ERROR MSG
1841 006630      TSTA:
*****
;*TEST 13 TEST INHIBIT DATA SHIFT ON DATIP
*****
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,$BECR1  ;HAVE UBE DO DATIP WITH INH DATA SHIFT
JSR PC,CRDY        ;CHECK FOR RDY BIT
TST R4             ;DID RDY SET?
BEQ T11L01         ;BRANCH IF YES
ERROR #D30         ;ERROR: DATIP FAILED TO SET RDY
JSR PC,TERRPC      ;TYPE PC OF ERROR MSG
BR TST14           ;:BRANCH TO NEXT TEST
006716 022777 052525 173572 T11L01: CMP #052525,$BEBD  ;IS (BEBD) OK?
006724 001004      BNE T11L02         ;BRANCH IF NO
006726 022767 052525 020744      CMP #052525,BUFF1 ;IS MEM OK?
006734 001417      BEQ TST14         ;:BRANCH IF YES TO NEXT TEST
006736 017767 173554 172250 T11L02: MOV $BEBD,$REG0   ;SAVE (BEBD)
006744 016767 020730 172244      MOV BUFF1,$REG1   ;SAVE MEM DATA
006752 012767 027700 172240      MOV #BUFF1,$REG2  ;SAVE MEM ADDRESS
006760 012767 052525 172234      MOV #052525,$REG3 ;SAVE CORRECT DATA
006766 104040      ERROR #D32        ;ERROR: INH. DATA SHIFT ON DATIP FAILED

```


1869 006770 004767 007340

JSR PC,TERRPC

;TYPE PC OF ERROR MSG

1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924

006774 000004
006776 012706 001100
007002 004767 007126
007006 005037 177776
007012 012767 177525 020660
007020 012777 027700 173474
007026 012777 177777 173464
007034 012777 042403 173462
007042 004767 007142
007046 022777 177253 173442
007054 001004
007056 022767 177653 020614
007064 001417
007066 017767 173424 172120
007074 016767 020600 172114
007102 012767 027700 172110
007110 012767 177653 172104
007116 104041
007120 004767 007210

```
*****
*TEST 14 TEST DATOB ON DATIP
*****
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,JBEBB ;LOAD UBE WITH LOW BYTE ADDRESS
MOV #177777,JBEBB ;HAVE UBE DO 1 XFER
MOV #42403,JBECR1 ;HAVE UBE DO DATOB ON DATIP
JSR PC,CRDY ;CHECK FOR RDY SET
CMP #177253,JBEBD ;CHECK (BEBD) OK
BNE T12L01 ;BRANCH IF NO
CMP #177653,BUFF1 ;CHECK BUFFER OK
BEQ TST15 ;BRANCH IF YES TO NEXT TEST
T12L01: MOV JBEBD,$REG0 ;SAVE (BEBD)
MOV BUFF1,$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 ADDRESS REG COUNTS BY 2 AND 1
*
*RO CONTAINS THE TEST DATA
*****
```

```
TST15: SCOPE
MOV #STACK,SP ;RESTORE STACK
JSR PC,CLRREG ;CLEAR UBE REGS
JSR PC,DINT ;DISREGARD UBE INTERRUPTS
CLR #PSW ;ALLOW INTERRUPTS
MOV #2,RO ;INITIALIZE TEST COUNTER
MOV #177777,JBEBB ;HAVE UBE DO 1 XFER
MOV #2003,JBECR1 ;HAVE UBE DO DATI
JSR PC,CRDY ;CHECK RDY SET
CMP RO,JBEBB ;IS ADDRESS CORRECT?
BNE T14L01 ;BRANCH TO ERROR IF NO
INC RO ;UPDATE RO
INC RO ;UPDATE RO
CMP #2,RO ;HAVE ALL ADDRESSES BEEN TESTED?
BNE T14L02 ;LOOK AT NEXT ADDRESS IF NO
MOV #177776,JBEBB ;LOAD MAX ADDRESS IN LOWER 16 BITS UBE
MOV #3,JBECR2 ;LOAD A16,A17 OF UBE WITH 1
MOV #177777,JBEBB ;HAVE UBE DO 1 XFER
INC JBECR1 ;HAVE UBE DO DATI
JSR PC,CRDY ;CHECK RDY SET
BIT #3,JBECR2 ;TEST A16,A17=0
BNE T14L03 ;BRANCH TO ERROR IF NO
```

;NOW TEST ADDRESS COUNTS BY 1

```

1925 007254 012777 027701 173240      MOV #BUFF1+1, @BEBA      ;PUT ODD ADD OF BUFFER IN UBE
1926 007262 012777 177777 173230      MOV #177777, @BECC      ;HAVE UBE DO 1 XFER
1927 007270 012777 003403 173226      MOV #3403, @BECC1      ;HAVE UBE DO DATOB
1929 007276 004767 006706              JSR PC, CRDY           ;CHECK RDY
1929 007302 022777 027702 173212      CMP #BUFF1+2, @BEBA     ;DID ADDRESS UPDATE BY 1?
1930 007310 001434              BEQ T14L04             ;BRANCHIF YES TO RESTORE TRAPS
1931 007312 017767 173204 171674      MOV @BEBA, $REG0       ;SAVE BAD ADDRESS
1932 007320 012767 027702 171670      MOV #BUFF1+2, $REG1    ;SAVE GOOD ADDRESS
1933 007326 104045              ERROR I037           ;ERROR: BEBA DID NOT COUNT BY 1
1934 007330 004767 007000              JSR PC, TERRPC       ;TYPE PC OF ERROR MSG
1935 007334 000422              BR T14L04             ;GO TO RESTORE TRAPS
1936 007336 017767 173160 171650 T14L01: MOV @BEBA, $REG0       ;SAVE BAD ADDRESS
1937 007344 010067 171646              MOV RO, $REG1        ;SAVE CORRECT ADDRESS
1938 007350 104043              ERROR I035           ;ERROR: BEBA DID NOT COUNT BY 2
1939 007352 004767 006756              JSR PC, TERRPC       ;TYPE PC OF ERROR MSG
1940 007356 000411              BR T14L04             ;GO TO RESTORE TRAPS
1941 007360 017700 173142 T14L03: MOV @BECC2, RO      ;GET ADDRESS BITS FROM UBE
1942 007364 042700 177774              BIC #177774, RO      ;JUST LOOK AT A16, A17
1943 007370 010067 171620              MOV RO, $REG0        ;SAVE ADDRESS
1944 007374 104044              ERROR I036           ;ERROR: BEBA BITS A16, A17 DID NOT COUNT = 0
1945 007376 004767 006732              JSR PC, TERRPC       ;TYPE PC OF ERROR MSG
1946 007402 004767 006560 T14L04: JSR PC, RCATCH   ;RESTORE TRAPS AND GO TO NEXT TEST
1947
1948
1949
1949
1950
1951
1952
1953
1954
1954
1955
1955
1956 007406 000004
1957 007410 012706 001100
1958 007414 004767 006514
1959 007420 004767 006614
1960 007424 005037 177776
1961
1962
1963
1964 007430 012737 007460 000004      MOV #T13L01, @#4      ;SET UP TIME OUT TRAP
1965 007436 012700 017776              MOV #17776, RO        ;SET RO=LAST ADDRESS IN 1ST 4K OF MEM
1966 007442 062700 004000 T13L02: ADD #4000, RO     ;UPDATE RO TO NEXT 1K OF MEM
1967 007446 005710              TST (RO)              ;TEST IF 1K PRESENT. TIMES OUT IF NOT.
1968 007450 022700 037776              CMP #37776, RO        ;AT 8K?
1969 007454 001372              BNE T13L02            ;LOOK AT NEXT 1K IF NOT
1970 007456 000402              BR T13L03
1971 007460 162700 004000 T13L01: SUB #4000, RO     ;GET ADDRESS OF LAST 1K OF MEM PRESENT
1972
1973 007464 012737 000006 000004 T13L03: MOV #6, @#4      ;RESTORE TRAP
1974 007472 011001              MOV (RO), R1          ;SAVE CONTENTS OF LAST LOC IN FIRST 8K
1975 007474 010010              MOV RO, (RO)          ;PUT ADDRESS OF LOC IN MEM LOC
1976 007476 012737 000000 000000      MOV #0, @#0           ;PUT 0 IN LOC 0
1977 007504 012777 177777 173006      MOV #177777, @BECC    ;HAVE UBE DO 1 XFER
1978 007512 012777 002003 173004      MOV #2003, @BECC1     ;HAVE UBE DO DATI FROM MEM LOC 0
1979 007520 004767 006464              JSR PC, CRDY         ;CHECK FOR RDY SET
1980 007524 005777 172766              TST @BEBO            ;SEE IF UBE READ 0 FROM LOC 0
    
```

```

*****
;TEST 16 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 GO ADDRESS.
*****
    
```

```

†ST16: SCOPE
MOV #STACK, SP ;RESTORE STACK
JSR PC, CLAREG ;CLEAR UBE REG
JSR PC, DINT ;DISREGARD INTERRUPTS
CLR @PSW ;ALLOW DATA TRANSFERS
    
```

;SIZE MEMORY FROM 4K TO 8K

1981	007530	001034			BNE T13L04	:BRANCH TO ERROR IF DATA NOT = 0
1982	007532	010077	172754		MOV RO, BBEBA	:HAVE UBE ADDRESS HIGHEST MEMORY IN 4K-8K LOCATIONS
1983	007536	012777	177777	172754	MOV #177777, BBECC	:HAVE UBE DO 1 XFER
1984	007544	005277	172754		INC BBECR1	:HAVE UBE DO DATI FROM HIGHEST MEMORY IN 4K-8K LOCATIONS
1985	007550	004767	006434		JSR PC, CRDY	:CHECK FOR RDY SET
1986	007554	020077	172736		CMP RO, BBEBD	:DID UBE READ FROM PROPER LOCATION?
1987	007560	001020			BNE T13L04	:BRANCH IF DATA NOT = RO
1988	007562	016777	172746	172732	MOV BEGO, BBEBA	:HAVE UBE ADDRESS ITS GO ADDRESS
1989	007570	012777	000003	172730	MOV #3, BBECR2	:HAVE UBE ADDRESS ITS GO ADDRESS
1990	007576	012777	177777	172714	MOV #177777, BBECC	:HAVE UBE DO 1 XFER
1991	007604	005277	172714		INC BBECR1	:HAVE UBE DO DATI FROM GO ADDRESS
1992	007610	004767	006374		JSR PC, CRDY	:CHECK FOR RDY SET
1993	007614	005777	172676		TST BBEBD	:DID UBE READ PROPER LOCATION?
1994	007620	001411			BEQ T13L05	:BRANCH IF YES
1995	007622	017767	172674	171364	T13L04: MOV BBEBA, \$REGO	:GET ADDRESS+2 TRIED TO READ FROM
1996	007630	162767	000002	171356	SUB #2, \$REGO	:CALC. ADDRESS TRIED TO READ FROM
1997	007636	104042			ERROR D34	:ERROR: UBE DID DATI FROM WRONG LOCATION
1998	007640	004767	006470		JSR PC, TERRPC	:TYPE PC OF ERROR MSG
1999	007644	004767	006316		T13L05: JSR PC, RCATCH	:RESTORE TRAPS
2000	007650	010110			MOV R1, (RO)	:RESTORE CONTENTS OF LAST LOC OF FIRST 8K

```

*****
:TEST 17 TEST CYCLE COUNT COUNTS BY 1 AND INCREMENTS WITH EACH INTERRUPT
:
:THE BECC REG IS CYCLED FROM 0 TO 177777 BY INTERRUPTING THE
:CPU. AFTER EACH INTERRUPT, THE REG IS COMPARED WITH RO WHICH
:CONTAINS THE PROPER DATA.
*****

```

2009	007652	000004			T17: SCOPE	
2010	007654	012706	001100		MOV #STACK, SP	:RESTORE STACK
2011	007660	012737	000340	177776	MOV #340, BPSW	:LOCK OUT INTERRUPTS
2012	007666	004767	006242		JSR PC, CLREG	:CLEAR UBE REG
2013	007672	005000			CLR RO	:INITIALIZE TEST COUNTER
2014	007674	012777	007716	172630	MOV #T15L01, INTVEC	:SET UP INT VECTOR AREA
2015	007702	012777	000003	172614	T15L03: MOV #3, BBECR1	:HAVE UBE INT VIA BR=4
2016	007710	005037	177776		CLR BPSW	:ALLOW INTERRUPTS
2017	007714	000240			NOP	:UBE WILL INTERRUPT HERE
2018	007716	022626			T15L01: CMP (SP)+, (SP)+	:RESTORE STACK AFTER INTERRUPT
2019	007720	005200			INC RO	:UPDATE TEST COUNTER
2020	007722	005700			TST RO	:IS RO=0?
2021	007724	001423			BEQ T15L02	:RESTORE TRAPS IF YES
2022	007726	020077	172566		CMP RO, BBECC	:DID CYCLE COUNT UPDATE PROPERLY?
2023	007732	001763			BEQ T15L03	:INCREMENT BECC IF YES
2024	007734	017767	172560	171252	MOV BBECC, \$REGO	:SAVE BAD DATA
2025	007742	010067	171250		MOV RO, \$REG1	:SAVE GOOD DATA
2026	007746	104046			ERROR D38	:ERROR: INTERRUPT FAILED TO UPDATE BECC TO CORRECT VALUE
2027	007750	004767	006360		JSR PC, TERRPC	:TYPE PC OF ERROR MSG
2028	007754	012737	000340	177776	MOV #340, BPSW	:LOCK OUT INTERRUPTS
2029	007762	012777	006003	172534	MOV #6003, BBECR1	:HAVE UBE CYCLE SO IT SETS RDY
2030	007770	005037	177776		CLR BPSW	:ALLOW UBE TO CYCLE
2031	007774	004767	006166		T15L02: JSR PC, RCATCH	:RESTORE TRAPS

```

*****
:TEST 20 TEST INHIBIT INCREMENT OF BECC AND BEBA
:
:A DATI IS DONE VIA BR ARBITRATION AND THE BECC AND BEBA REGS

```

```

2037
2038
2039 010000 000004
2040 010002 012706 001100
2041 010006 012737 000340 177776
2042 010014 004767 006114
2043 010020 012777 027700 172474
2044 010026 012777 177777 172464
2045 010034 012767 000001 017636
2046 010042 012777 000004 172456
2047 010050 012777 002003 172446
2048 010056 005037 177776
2049 010062 005777 172430
2050 010066 001775
2051 010070 022777 177777 172422
2052 010076 001010
2053 010100 022777 027700 172414
2054 010106 001407
2055 010110 104047
2056 010112 004767 006216
2057 010116 000403
2058 010120 104050
2059 010122 004767 006206
2060 010126 042777 000004 172372
2061 010134 004767 006050
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071 010140 000004
2072 010142 012706 001100
2073 010146 012737 000340 177776
2074 010154 004767 005754
2075 010160 012700 027700
2076 010164 005020
2077 010166 020027 027720
2078 010172 001374
2079 010174 012777 000377 172314
2080 010202 012777 027700 172312
2081 010210 012777 177774 172302
2082 010216 012777 010260 172306
2083 010224 012777 003121 172272
2084 010232 005037 177776
2085 010236 005000
2086 010240 005200
2087 010242 022700 001000
2088 010246 001374
2089 010250 104051
2090 010252 004767 006056
2091 010256 000470
2092 010260 012700 027700

```

```

;*ARE CHECKED TO NOT INCREMENT.
;*****
†ST20: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,‡PSW ;LOCK OUT INTERRUPTS
JSR PC,CLREG ;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,‡BECCR2 ;HAVE UBE INH. INC. OF BECC AND BEBA
MOV #2003,‡BECCR1 ;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,‡BECCR2 ;ALLOW BEBA AND BECC TO COUNT
JSR PC,CRDY ;WAIT TILL UBE IS DONE

```

```

;*****
;TEST 21 TEST INT ENB AND CCOVF WORK AND THAT UBE WILL DO SEVERAL XFERS
;
;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.
;*****

```

```

†ST21: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,‡PSW ;LOCK OUT INTERRUPTS
JSR PC,CLREG ;CLEAR UBE REG
MOV #BUFF1,RO ;GET BUFFER ADDRESS
T17L01: CLR (RO)+ ;CLEAR BUFFER AREA
CMP RO,#BUFF1+20 ;AT END OF BUFFER?
BNE T17L01 ;BRANCH IF NO
MOV #377,‡BEBD ;SET UP XFER TEST DATA
MOV #BUFF1,‡BEBA ;LOAD UBE WITH BUFF ADDRESS
MOV #177774,‡BECC ;SET UBE TO DO 4 XFERS
MOV #T17L02,‡INTVEC ;SET UP INT VECTOR
MOV #3121,‡BECCR1 ;HAVE UBE DO DATO VIA BR=7 AND INTERRUPT ON DONE
CLR ‡PSW ;ALLOW XFERS
CLR RO ;INITIALIZE COUNT
T17L03: INC RO ;UPDATE COUNT TO WAIT FOR INTERRUPT
CMP #1000,RO ;WAITED LONG ENOUGH?
BNE T17L03 ;BRANCH IF NO
ERROR †D41 ;ERROR: UBE FAILED TO INT. ON DONE
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
BR T17L09 ;GO RESTORE TRAPS
T17L02: MOV #BUFF1,RO ;GET START OF BUFFER

```


H04

UNIBUS EXERCISOR MODULE DIAGNOSTIC MACY11 27(732) 17-SEP-76 15:49 PAGE 44
DZKUBA.P11 T21 TEST INT ENB AND CCOVF WORK AND THAT UBE WILL DO SEVERAL XFRS

2093	010264	005720			T17L05:	TST (R0)+	:TEST FIRST 4 LOC WRITTEN
2094	010266	001433				BEQ T17L04	:BRANCH IF NOT WRITTEN TO ERROR
2095	010270	022700	027710			CMP #BUFF1+10,R0	:LOOKED AT ALL WRITTEN LOCS.
2096	010274	001373				BNE T17L05	:BRANCH IF NO
2097	010276	005720			T17L06:	TST (R0)+	:TEST LAST 4 LOC WERE NOT WRITTEN
2098	010300	001027				BNE T17L10	:BRANCH TO ERROR IF WERE
2099	010302	022700	027720			CMP #BUFF1+20,R0	:AT END OF BUFFER?
2100	010306	001373				BNE T17L06	:NO, LOOK AT NEXT LOCATION
2101	010310	032777	000100	172206		BIT #100, @BECR1	:YES, TEST INT. ON DONE BIT=0
2102	010316	001041				BNE T17L07	:BRANCH TO ERROR IF NOT=0
2103	010320	032777	020000	172200		BIT #20000, @BECR2	:TEST CCOVF=1
2104	010326	001441				BEQ T17L08	:BRANCH TO ERROR IF=0
2105	010330	012777	006003	172166		MOV #5003, @BECR1	:SET GO BIT TO SEE IF CCOVF IS RESET
2106	010336	032777	020000	172162		BIT #20000, @BECR2	:TEST CCOVF=0
2107	010344	001435				BEQ T17L09	:GO RESTORE TRAPS IF YES
2108	010346	104052				ERROR #D42	:ERROR: CCOVF NOT CLEARED BY GO
2109	010350	004767	005760			JSR PC, TERRPC	:TYPE PC OF ERROR MSG
2110	010354	000431				BR T17L09	:GO RESTORE TRAPS
2111	010356	005740			T17L04:	TST -(R0)	:CALC. LAST ADD. WRITTEN
2112	010360	005740			T17L10:	TST -(R0)	:CALC. LAST ADD. WRITTEN
2113	010362	022700	027700			CMP #BUFF1,R0	:WERE ANY ADD. WRITTEN?
2114	010366	003404				BLE T17L11	:BRANCH IF YES
2115	010370	104067				ERROR #D55	:ERROR: UBE DID NOT DO DATO TO PROPER # OF LOC (4)
2116	010372	004767	005736			JSR PC, TERRPC	:TYPE PC OF ERROR MSG
2117	010376	000420				BR T17L09	:GO RESTORE TRAPS
2118	010400	012767	027700	170606	T17L11:	MOV #BUFF1, \$REG0	:SAVE FIRST LOCATION WRITTEN
2119	010406	010067	170604			MOV R0, \$REG1	:SAVE LAST LOCATION WRITTEN
2120	010412	104053				ERROR #D43	:ERROR: UBE DID NOT DO DATO TO PROPER # OF LOCATIONS (4)
2121	010414	004767	005714			JSR PC, TERRPC	:TYPE PC OF ERROR MSG
2122	010420	000407				BR T17L09	:GO RESTORE TRAPS
2123	010422	104054			T17L07:	ERROR #D44	:ERROR: INT. ON DONE BIT NOT CLEARED
2124	010424	004767	005704			JSR PC, TERRPC	:TYPE PC OF ERROR MSG
2125	010430	000403				BR T17L09	:GO RESTORE TRAPS
2126	010432	104055			T17L08:	ERROR #D45	:ERROR: CCOVF NOT SET
2127	010434	004767	005674			JSR PC, TERRPC	:TYPE PC OF ERROR MSG
2128	010440	004767	005522		T17L09:	JSR PC, RCATCH	:RESTORE TRAPS
2129							
2130							
2131							
2132							
2133							
2134							
2135							
2136	010444	000004					
2137	010446	012706	001100				
2138	010452	004767	005456				
2139	010456	005037	177776				
2140	010462	012700	027700				
2141	010466	005020					
2142	010470	020027	027720		T18L01:	CLR (R0)+	:RESTORE STACK
2143	010474	001374				CMP R0, #BUFF1+20	:CLEAR UBE REG
2144	010476	012777	027700	172016		BNE T18L01	:ALLOW INTERRUPTS
2145	010504	012777	177774	172006		MOV #BUFF1, @BEBA	:GET BUFFER ADDRESS
2146	010512	012777	013003	172004		MOV #177774, @BECC	:CLEAR BUFFER AREA
2147	010520	032777	000200	171776	T18L02:	BIT #200, @BECR1	:AT END OF BUFFER?
2148	010526	001774				BEQ T18L02	:BRANCH IF NO

:TEST 22 TEST DATA XFRS FROM BECC

:THE UBE IS SET UP TO DO 4 DATO XFRS VIA BR ARBITRATION FROM
:THE BECC REG TO A BUFFER AREA. THE AREA IS THEN CHECKED.

TST22: 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 XFRS
T18L02: BIT #200, @BECR1 :HAVE UBE DO 4 XFRS FROM BECC
BEQ T18L02 :LOOK FOR RDY SET
:BRANCH TILL SET

2149	010530	012700	027700		MOV #BUFF1,RO	:GET BUFFER ADDRESS
2150	010534	012701	177774		MOV #177774,R1	:INITIALIZE R1=TO FIRST DATA WORD
2151	010540	022001		T18L04:	CMP (RO)+,R1	:IS DATA OK?
2152	010542	001005			BNE T18L03	:NO, GO TO ERROR
2153	010544	005201			INC R1	:UPDATE FOR NEXT DATA
2154	010546	020027	027710		CMP RO,#BUFF1+10	:LOOKED AT ALL DATA?
2155	010552	001372			BNE T18L04	:NO, LOOK AT NEXT WORD
2156	010554	000412			BR TST23	:GO TO NEXT TEST
2157						
2158	010556	005740		T18L03:	TST -(RO)	:CALC. ADDRESS OF FAILURES
2159	010560	010067	170430		MOV RO,\$REGO	:SAVE ADDRESS
2160	010564	011067	170426		MOV (RO),\$REG1	:SAVE BAD DATA
2161	010570	010167	170424		MOV R1,\$REG2	:SAVE GOOD DATA
2162	010574	104056			ERROR 1D46	:ERROR: DATO FROM BECC NOT DONE PROPERLY
2163	010576	004767	005532		JSR PC,TERRPC	:TYPE PC OF ERROR MSG
2164						
2165						
2166						
2167						
2168						
2169						
2170						
2171						
2172						
2173						
2174						
2175						
2176	010602	000004				
2177	010604	012706	001100		TST23: SCOPE	
2178	010610	012737	000340	177776	MOV #STACK,SP	:RESTORE STACK
2179	010616	004767	005312		MOV #340,@#PSW	:LOCK ON INTERRUPTS
2180	010622	012700	027700		JSR PC,CLRREG	:CLEAR UBE REGS
2181	010626	005020			MOV #BUFF1,RO	:GET BUFFER ADDRESS
2182	010630	020027	027720	T19L01:	CLR (RO)+	:CLEAR BUFFER AREA
2183	010634	001374			CMP RO,#BUFF1+20	:AT END OF BUFFER?
2184	010636	012777	027700	171656	BNE T19L01	:CONTINUE TO CLEAR IF NO
2185	010644	012777	177774	171646	MOV #BUFF1,@BEBA	:LOAD BUFFER ADDRESS INTO UBE
2186	010652	012777	000377	171636	MOV #177774,@BECC	:SET UBE TO DO 4 XFERS
2187	010660	012777	005003	171636	MOV #377,@BEBD	:LOAD TEST DATA INTO UBE
2188	010666	005037	177776		MOV #5003,@BEBC1	:HAVE UBE DO 2 DATO/REQUEST VIA BR=4
2189	010672	000240			CLR @#PSW	:ALLOW UBE TO DO XFERS
2190	010674	012737	000340	177776	NOP	:UBE SHOULD DO 2 XFERS HERE
2191	010702	012700	027700		MOV #340,@#PSW	:SET PRIORITY=7 TO STOP LAST 2 XFERS
2192	010706	005720			MOV #BUFF1,RO	:GET BUFF ADDRESS
2193	010710	001411		T19L03:	TST (RO)+	:WAS BUFF WRITTEN?
2194	010712	020027	027704		BEQ T19L09	:BRANCH TO ERROR IF NO
2195	010716	001373			CMP RO,#BUFF1+4	:LOOKED AT FIRST 2 LOCATIONS?
2196	010720	005720		T19L04:	BNE T19L03	:BRANCH IF NO
2197	010722	001005			TST (RO)+	:TEST BUFF LOC NOT WRITTEN
2198	010724	020027	027710		BNE T19L02	:BRANCH TO ERROR IF WRITTEN
2199	010730	001373			CMP RO,#BUFF1+10	:LOOKED AT FOURTH LOC?
2200	010732	000421			BNE T19L04	:BRANCH IF NO
2201	010734	005740			BR T19L05	:GO TO END OF TEST
2202	010736	005740		T19L09:	TST -(RO)	:CALC LAST ADDRESS WRITTEN
2203	010740	022700	027700	T19L02:	TST -(RO)	:CALC LAST ADDRESS WRITTEN
2204	010744	101404			CMP #BUFF1,RO	:WERE ANY ADDRESS WRITTEN?
					BLOS T19L07	:BRANCH IF YES

```

:*****
:TEST 23      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.
:*****

```


2205	010746	104060				ERROR ↑D48	:ERROR: UBE DID NOT DO 2 XFERS/REQUEST
2206	010750	004767	005360			JSR PC,TERRPC	:TYPE PC OF ERROR MSG
2207	010754	000410				BR T19L05	:GO TO END OF TEST
2208	010756	012767	027700	170230	T19L07:	MOV #BUFF1,\$REGO	:SAVE FIRST ADDRESS WRITTEN
2209	010764	010067	170226			MOV RO,\$REG1	:SAVE LAST ADDRESS WRITTEN
2210	010770	104057				ERROR ↑D47	:ERROR: UBE DID NOT DO 2 XFERS FOR EACH REQUEST
2211	010772	004767	005336			JSR PC,TERRPC	:TYPE PC OF ERROR MSG
2212	010776	005037	177776		T19L05:	CLR @PSW	:ALLOW LAST 2 XFERS
2213	011002	000240				NOP	:ALLOW UBE TO GET BUS
2214	011004	004767	005200			JSR PC,CRDY	:WAIT TILL UBE FINISHES XFERS

```

*****
:TEST 24 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.
*****

```

2227	011010	000004				TST24: SCOPE	
2228	011012	012706	001100			MOV #STACK,SP	:RESTORE STACK
2229	011016	012737	000340	177776		MOV #340,@PSW	:LOCK OUT INTERRUPTS
2230	011024	004767	005104			JSR PC,CLRREG	:CLEAR UBE REG
2231	011030	012700	027700			MOV #BUFF1,RO	:GET BUFFER ADDRESS
2232	011034	012720	125252		T20L01:	MOV #125252,(RO)+	:LOAD TEST DATA
2233	011040	020027	027710			CMP RO,#BUFF1+10	:LOADED FIRST 4 LOCS?
2234	011044	001373				BNE T20L01	:BRANCH IF NO
2235	011046	012777	027700	171446		MOV #BUFF1,@BEBA	:LOAD BUFFER ADDRESS INTO UBE
2236	011054	012777	177774	171436		MOV #177774,@BECC	:SET UBE TO DO 4 CYCLES
2237	011062	012777	004403	171434		MOV #4403,@BECC1	:HAVE UBE DO 2 DATIP/REQUEST VIA BR=4
2238	011070	005037	177776			CLR @PSW	:ALLOW UBE TO DO CYCLES
2239	011074	000240				NOP	:UBE SHOULD DO XFERS HERE
2240	011076	012737	000340	177776		MOV #340,@PSW	:SET PRIORITY = 7 TO STOP LAST 2 CYCLES
2241	011104	012700	027700			MOV #BUFF1,RO	:GET BUFF ADDRESS
2242	011110	022720	052525		T20L03:	CMP #052525,(RO)+	:TEST BUFF LOCS WRITTEN
2243	011114	001012				BNE T20L02	:BRANCH TO ERROR IF NOT DONE PROPERLY
2244	011116	022700	027704			CMP #BUFF1+4,RO	:LOOKED AT 2 WRITTEN LOCS?
2245	011122	001372				BNE T20L03	:BRANCH IF NO
2246	011124	022720	125252		T20L04:	CMP #125252,(RO)+	:TEST BUFF LOCS NOT WRITTEN
2247	011130	001005				BNE T20L08	:BRANCH TO ERROR IF WRITTEN
2248	011132	020027	027710			CMP RO,#BUFF1+10	:LOOKED AT FOURTH LOC?
2249	011136	001372				BNE T20L04	:BRANCH IF NO
2250	011140	000421				BR T20L05	:GO TO END OF TEST
2251	011142	005740			T20L02:	TST -(RO)	:CALC LAST ADDRESS WRITTEN
2252	011144	005740			T20L08:	TST -(RO)	:CALC LAST ADDRESS WRITTEN
2253	011146	022700	027700			CMP #BUFF1,RO	:WERE ANY LOC WRITTEN?
2254	011152	101404				BLOS T20L06	:BRANCH IF YES
2255	011154	104061				ERROR ↑D49	:ERROR: DID NOT DO 2 DATIP/REQUEST
2256	011156	004767	005152			JSR PC,TERRPC	:TYPE PC OF ERROR MSG
2257	011162	000410				BR T20L05	:GO TO END OF TEST
2258	011164	012767	027700	170022	T20L06:	MOV #BUFF1,\$REGO	:SAVE FIRST ADDRESS WRITTEN
2259	011172	010067	170020			MOV RO,\$REG1	:SAVE LAST ADDRESS WRITTEN
2260	011176	104062				ERROR ↑D50	:ERROR: UBE DID NOT DO 2 DATIP/REQUEST

```

2261 011200 004767 005130
2262 011204 005037 177776
2263 011210 000240
2264 011212 004767 004772
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276 011216 000004
2277 011220 012706 001100
2278 011224 012737 000340 177776
2279 011232 004767 004676
2280 011236 005067 016436
2281 011242 012777 177777 171246
2282 011250 012777 027700 171244
2283 011256 012777 177777 171234
2284 011264 012777 003041 171232
2285 011272 000240
2286 011274 004767 004710
2287 011300 005704
2288 011302 001042
2289 011304 005767 016370
2290 011310 001452
2291 011312 005067 016362
2292 011316 005067 016360
2293 011322 012777 011372 171202
2294 011330 012777 027700 171164
2295 011336 012777 177777 171154
2296 011344 012777 003143 171152
2297 011352 005037 177776
2298 011356 004767 004626
2299 011362 104065
2300 011364 004767 004744
2301 011370 000425
2302 011372 005767 016304
2303 011376 001422
2304 011400 104066
2305 011402 004767 004726
2306 011406 000416
2307 011410 104064
2308 011412 004767 004716
2309 011416 012777 006003 171100
2310 011424 005037 177776
2311 011430 004767 004554
2312 011434 000403
2313 011436 104063
2314 011440 004767 004670
2315 011444 004767 004516
2316

```

```

T20L05: JSR PC,TERRPC ;TYPE PC OF ERROR MSG
CLR @#PSW ;ALLOW LAST 2 CYCLES
NOP ;ALLOW UBE TO GET BUS
JSR PC,CRDY ;WAIT FOR UBE TO FINISH XFRS

*****
*TEST 25 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.
*****
TST25: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,@#PSW ;LOCK OUT INTERRUPTS
JSR PC,CLRREG ;CLEAR UBE REG
CLR BUFF1 ;CLEAR BUFFER LOC
MOV #177777,@BEED ;LOAD UBE DATA REG WITH TEST DATA
MOV #BUFF1,@BEBA ;LOAD UBE ADDRESS REG WITH BUFF ADD.
MOV #177777,@BECC ;SET UBE TO DO 1 CYCLE
MOV #3041,@BECR1 ;HAVE UBE DO DATO VIA NPR
NOP ;ALLOW UBE TO SET BUS
JSR PC,CRDY ;CHECK RDY SET
TST R4 ;DID RDY SET?
BNE T21L01 ;BRANCH TO ERROR IF RDY DID NOT SET
TST BUFF1 ;WAS DATO DONE?
BEQ T21L02 ;BRANCH TO ERROR IF NPR NOT DONE
CLR BUFF1 ;CLEAR BUFF LOC
CLR BUFF1+2 ;CLEAR BUFF LOC +2
MOV #T21L03,@INTVEC ;SET UP FOR INTERRUPT
MOV #BUFF1,@BEBA ;LOAD TEST ADDRESS
MOV #177777,@BECC ;SET UBE TO DO 1 CYCLE
MOV #3143,@BECR1 ;HAVE UBE DO DATO NPR AND INT WHEN DONE VIA BR=4
CLR @#PSW ;ALLOW UBE TO INTERRUPT
JSR PC,CRDY ;WAIT FOR INT. OR RDY TO SET
ERROR #D53 ;ERROR: UBE DID NOT INT WHEN NPR DONE
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
BR T21L04 ;RESTORE TRAPS
T21L03: TST BUFF1+2 ;DID NPR WRITE MORE THAN 1 LOC?
BEQ T21L04 ;GO TO END OF TEST
ERROR #D54 ;ERROR: UBE WROTE 2 LOC WHEN 1 NPR AND INT DONE
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
BR T21L04 ;RESTORE TRAPS
T21L01: ERROR #D52 ;ERROR: NPR DID NOT SET RDY
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
MOV #6003,@BECR1 ;HAVE UBE SET ITS RDY
CLR @#PSW
JSR PC,CRDY ;WAIT TILL SET
BR T21L04 ;RESTORE TRAPS
T21L02: ERROR #D51 ;ERROR: NPR DATO NOT DONE
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
T21L04: JSR PC,RCATCH ;RESTORE TRAPS

```



```

2317
2318
2319
2320
2321
2322
2323 011450 000004
2324 011452 012767 000001 167554
2325 011460 012706 001100
2326 011464 004767 004444
2327 011470 032737 010000 001172
2328 011476 001002
2329 011500 104401 025157
2330
2331 011504 012777 177777 171006 1S:
2332 011512 012777 000043 171004
2333 011520 005037 177776
2334 011524 000240
2335 011526 032777 000001 170770
2336 011534 001003
2337 011536 104011
2338 011540 004767 004570
2339 011544 005077 170754
2340 011550 032737 010000 001172 T22L01:
2341 011556 001002
2342 011560 104401 025307
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355 011564 000004
2356 011566 012706 001100
2357 011572 012737 000340 177776
2358 011600 004767 004330
2359 011604 012777 011732 170720
2360 011612 012777 160000 170702
2361 011620 012777 000003 170700
2362 011626 012777 177777 170664
2363 011634 012777 002041 170662
2364 011642 004767 004342
2365 011646 032777 000400 170652
2366 011654 001004
2367 011656 104073
2368 011660 104074
2369 011662 004767 004446
2370 011666 032777 100000 170630 T23L02:
2371 011674 001004
2372 011676 104073

```

```

*****
*TEST 26 TEST UBE WILL NOT INTERRUPT DURING AN NPR AND GO BIT SETS
*
*IF THIS TEST FAILS AND THE UBE DOES INTERRUPT AFTER
*TRYING TO DO AN NPR, THE CPU WILL GO DOWN
*****
TST26: SCOPE
MOV #1,STIMES ;DO 1 ITERATION
MOV #STACK,SP ;RESTORE STACK
JSR PC,CLRREG ;CLEAR UBE REG
BIT #SW12,#SWR ;INHIBIT TYPEOUTS?
BNE 1S ;BRANCH IF YES
TYPE ,MSG3 ;TESTING UBE WILL NOT INTERRUPT
;DURING NPR. IF DOES, CPU WILL GO DOWN
1S: MOV #177777,#BECC ;SET UBE TO DO 1 CYCLE
MOV #0043,#BECR1 ;HAVE UBE DO DATI NPR AND INT. (FUN.=0)
CLR #PSW
NOP ;UBE SHOULD NOT GET BUSS HERE
BIT #1,#BECR1 ;IS GO BIT SET?
BNE T22L01 ;BRANCH IF YES
ERROR #D9 ;ERROR: GO BIT FAILED TO LOAD '1'
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
CLR #BECR1 ;RESET GO BIT, NPR AND INTERRUPT
BIT #SW12,#SWR ;INHIBIT TYPEOUTS?
BNE TST27 ;BRANCH IF YES
TYPE ,MSG4 ;EXITING TEST
*****
*TEST 27 TEST NO SSSYN ERROR BIT WORKS & FUN A,B BITS RESET BY ERROR INTERRUPT
*
*A DATI NPR IS DONE TO A MEM LOC (760000) THAT RETURNS
*NO SSSYN. THE NO SSSYN ERROR BIT AND BIT 15 OF BECR1
*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.
*****
TST27: SCOPE
MOV #STACK,SP ;RESTORE STACK
MOV #340,#PSW ;LOCK OUT INTERRUPTS
JSR PC,CLRREG ;CLEAR UBE REG
MOV #T23L01,#INTVEC ;SET UP FOR INTERRUPTS
MOV #160000,#BEBA ;LOAD UBE WITH TEST ADDRESS WHICH RETURNS NO SSSYN
MOV #3,#BECR2 ;LOAD UBE WITH TEST ADDRESS WHICH RETURNS NO SSSYN
MOV #177777,#BECC ;HAVE UBE DO 1 CYCLE
MOV #2041,#BECR1 ;HAVE DATI NPR DONE
JSR PC,CROY ;WAIT TILL RDY SET
BIT #400,#BECR2 ;WAS NSSYN ERROR BIT SET?
BNE T23L02 ;BRANCH IF YES
ERROR #D59 ;ERROR: TEST OF NSSYN ERROR BIT FAILED
ERROR #D60 ;TO SET BIT 8 OF BECR2
JSR PC,TERRPC ;TYPE PC OF ERROR MSG
BIT #100000,#BECR1 ;WAS ERROR BIT SET?
BNE T23L03 ;BRANCH IF YES
ERROR #D59 ;ERROR: TEST OF NSSYN ERROR BIT FAILED

```


2373	011700	104075			ERROR ↑D61		: TO SET BIT 15 OF BECR1
2374	011702	004767	004426		JSR PC,TERRPC		: TYPE PC OF ERROR MSG
2375	011706	005037	177776		T23L03: CLR @#PSW		: ALLOW UBE TO INTERRUPT
2376	011712	000240			NOP		: UBE SHOULD INTERRUPT HERE
2377	011714	017767	170606	167272	MOV @BECR2,\$REGO		: SAVE BECR2
2378	011722	104073			ERROR ↑D59		: ERROR: TEST OF NSSYN ERROR BIT FAILED
2379	011724	104072			ERROR ↑D58		: TO INTERRUPT CPU
2380	011726	004767	004402		JSR PC,TERRPC		: TYPE PC OF ERROR MSG
2381	011732	005077	170572		T23L01: CLR @BERE		: CLEAR ERROR BITS
2382	011736	032777	000400	170562	BIT #400,@BECR2		: WAS NSSYN ERROR BIT CLEARED?
2383	011744	001404			BEQ T23L05		: BRANCH IF YES TO TEST FUN A, B BITS
2384	011746	104073			ERROR ↑D59		: ERROR: TEST OF NSSYN ERROR BIT FAILED
2385	011750	104076			ERROR ↑D62		: TO CLEAR BIT 8 OF BECR2
2386	011752	004767	004356		JSR PC,TERRPC		: TYPE PC OF ERROR MSG
2387	011756	032777	002000	170540	T23L05: BIT #2000,@BECR1		: WAS FUN A BIT RESET?
2388	011764	001404			BEQ T23L06		: BRANCH IF YES
2389	011766	104073			ERROR ↑D59		: ERROR: TEST OF NSSYN ERROR BIT FAILED
2390	011770	104016			ERROR ↑D14		: TO CLEAR BIT 10 OF BECR1
2391	011772	004767	004336		JSR PC,TERRPC		: TYPE PC OF ERROR MSG
2392	011776	012777	160000	170516	T23L06: MOV #160000,@BEBA		: LOAD UBE WITH TEST ADDRESS WHICH RETURNS NO S5YN
2393	012004	012777	000003	170514	MOV #3,@BECR2		: LOAD UBE WITH TEST ADDRESS WHICH RETURNS NO S5YN
2394	012012	012777	177772	170500	MOV #177772,@BECC		: DO 2 CYCLES
2395	012020	012777	012040	170504	MOV #T23L07,@INTVEC		: SET UP FOR INT
2396	012026	012777	004041	170470	MOV #4041,@BECR1		: HAVE UBE DO FUN2 DATI VIA NPR
2397	012034	004767	004150		JSR PC,CRDY		: WAIT TILL RDY SETS
2398	012040	032777	004000	170456	T23L07: BIT #4000,@BECR1		: WAS FUN B BIT RESET
2399	012046	001404			BEQ T23L04		: RESTORE TRAP
2400	012050	104073			ERROR ↑D59		: ERROR: TEST OF NSSYN ERROR BIT FAILED
2401	012052	104105			ERROR ↑D69		: TO CLEAR BIT 11 OF BECR1
2402	012054	004767	004254		JSR PC,TERRPC		: TYPE PC OF ERROR MSG
2403	012060	004767	004102		T23L04: JSR PC,RCATCH		: RESTORE TRAP
2404	012064	005077	170440		CLR @BERE		: CLEAR ALL ERROR CONDITIONS
2405							
2406							
2407							
2408							
2409							
2410							
2411							
2412							
2413							
2414	012070	000004			TST30: SCOPE		
2415	012072	012706	001100		MOV #STACK,SP		: RESTORE STACK
2416	012076	012737	000340	177776	MOV #340,@#PSW		: LOCK OUT INTERRUPTS
2417	012104	004767	004024		JSR PC,CLRREG		: CLEAR UBE REGS
2418	012110	016777	170420	170404	MOV BEGO,@BEBA		: HAVE UBE ADDRESS ITS GO ADDRESS
2419	012116	012777	000003	170402	MOV #3,@BECR2		: HAVE UBE ADDRESS ITS GO ADDRESS
2420	012124	012777	177777	170366	MOV #177777,@BECC		: SET UP TO DO 1 CYCLE
2421	012132	012777	012202	170372	MOV #T24L01,@INTVEC		: SET UP FOR INT.
2422	012140	012777	002041	170356	MOV #2041,@BECR1		: HAVE DATI NPR DONE FROM GO ADDRESS
2423	012146	004767	004036		JSR PC,CRDY		: CHECK FOR RDY SET
2424	012152	032777	001000	170346	BIT #1000,@BECR2		: WAS ADDRESS ERROR SET?
2425	012160	001404			BEQ T24L02		: BRANCH IF NO
2426	012162	104070			ERROR ↑D56		: ERROR: TEST OF WRONG A LINES ERROR BIT FAILED
2427	012164	104071			ERROR ↑D57		: BECR2 BIT 9 FALSELY SET
2428	012166	004767	004142		JSR PC,TERRPC		: TYPE PC OF ERROR MSG

```

*****
: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.
*****

```


2429	012172	005037	177776		T24L02:	CLR @#PSW	:ALLOW ANY INTERRUPTS
2430	012176	000240				NOP	:UBE SHOULD NOT INTERRUPT HERE
2431	012200	000410				BR T24L06	:GO TEST INDIVIDUAL ADDRESS BITS
2432	012202	017767	170320	167004	T24L01:	MOV @BECR2,\$REGO	:SAVE BECR2
2433	012210	104070				ERROR ^D56	:ERROR:TEST OF WRONG A LINES ERROR BIT FAILED
2434	012212	104077				ERROR ^D63	:FALSELY INTERRUPTED CPU
2435	012214	004767	004114			JSR PC,TERRPC	:TYPE PC OF ERROR MSG
2436	012220	000447				BR T24L03	:GO RESTORE TRAP
2437	012222	004767	004012		T24L06:	JSR PC,DINT	:DISREGARD INTERRUPTS
2438	012226	005077	170270			CLR @BEBA	:CLEAR ADDRESS 0-15
2439	012232	012777	000001	170266		MOV #1,@BECR2	:TEST ADDRESS 16
2440	012240	012777	177777	170252	T24L05:	MOV #177777,@BECC	:DO 1 CYCLE
2441	012246	062777	040000	170246		ADD #40000,@BEBA	:TEST NEXT ADDRESS
2442	012254	032777	140000	170240		BIT #140000,@BEBA	:HAVE ADDRESS BITS 14,15 BEEN EXERCISED?
2443	012262	001011				BNE T24L04	:TEST NEXT ADDRESS IF NO
2444	012264	032777	000003	170234		BIT #3,@BECR2	:HAVE ADDRESS BITS 16,17 BEEN EXERCISED?
2445	012272	001422				BEQ T24L03	:GO RESTORE TRAPS IF YES
2446	012274	005277	170226			INC @BECR2	:INC ADDRESS BITS 16,17
2447	012300	042777	000004	170220		BIC #4,@BECR2	:CLEAR BIT 2 OF BECR2 IF SET
2448	012306	012777	002041	170210	T24L04:	MOV #2041,@BECR1	:DO DATI NPR TO ADDRESS
2449	012314	004767	003670			JSR PC,CRDY	:WAIT TILL RDY SET
2450	012320	032777	001000	170200		BIT #1000,@BECR2	:WAS WRONG ADDRESS LINES ERROR BIT SET?
2451	012326	001744				BEQ T24L05	:TEST NEXT ADDRESS IF NO
2452	012330	104070				ERROR ^D56	:ERROR:TEST OF WRONG A LINES ERROR BIT FAILED
2453	012332	104071				ERROR ^D57	:BECR2 BIT 9 FALSELY SET
2454	012334	004767	003774			JSR PC,TERRPC	:TYPE PC OF ERROR MSG
2455	012340	004767	003622		T24L03:	JSR PC,RCATCH	:RESTORE TRAP CATCHER

 :TEST 31 TEST WRONG GRANT AND NO GRANT OR NOT ONE GRANT ERROR BITS DO NOT 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.
 :*****

2456					TST31:	SCOPE	
2457						MOV #STACK,SP	:RESTORE STACK
2458						JSR PC,CLRREG	:CLEAR UBE REG
2459						MOV #2000,@BECR1	:SET UP UBE TO DO 1 DATI XFER/REQ.
2460						MOV #T25L01,@INTVEC	:SET UP FOR INTERRUPTS
2461						CLR @#PSW	:ALLOW DATA XFERS VIA BR AND NPR LEVELS
2462					T25L05:	MOV #177777,@BECC	:SET UBE TO DO 1 CYCLE
2463						MOV #BUFF1,@BEBA	:SET UBE TO ADDRESS BUFFER AREA
2464						ADD #3,@BECR1	:HAVE UBE DO NEXT LEVEL OF REQUEST
2465						JSR PC,CRDY	:WAIT TILL RDY SET
2466						BIT #76,@BECR1	:HAVE ALL REQUEST LEVELS BEEN EXERCISED
2467	012344	000004				BEQ T25L02	:BRANCH IF YES
2468	012346	012706	001100			BIT #40,@BECR2	:WAS WRONG GRANT ERROR BIT SET?
2469	012352	004767	003556			BNE T25L03	:BRANCH TO ERROR IF SET
2470	012356	012777	002000	170140		BIT #2000,@BECR2	:WAS NO GRANT OR NOT ONE GRANT ERROR BIT SET?
2471	012364	012777	012456	170140		BNE T25L04	:BRANCH TO ERROR IF YES
2472	012372	005037	177776			BR T25L05	:GO TEST NEXT LEVEL
2473	012376	012777	177777	170114		MOV #BUFF1,@BEBA	:ERROR:TEST OF WRONG GRANT OR NOT ONE GRANT FAILED
2474	012404	012777	027700	170110		ADD #3,@BECR1	
2475	012412	062777	000003	170104		JSR PC,CRDY	
2476	012420	004767	003564			BIT #76,@BECR1	
2477	012424	032777	000076	170072		BEQ T25L02	
2478	012432	001425				BIT #40,@BECR2	
2479	012434	032777	000040	170064		BNE T25L03	
2480	012442	001062				BIT #2000,@BECR2	
2481	012444	032777	002000	170054		BNE T25L04	
2482	012452	001066				BR T25L05	
2483	012454	000750				MOV #BUFF1,@BEBA	
2484	012456	104100			T25L01:	ERROR ^D64	


```

2485 012460 017767 170042 166526 MOV @BECR2,$REGO ;SAVE ERROR BITS
2486 012466 104077 ERROR 1D63 ;FALSELY INTERRUPTED CPU
2487 012470 017767 170030 166516 MOV @BECR1,$REGO ;SAVE BECR1
2488 012476 104104 ERROR 1D69 ;WITH BECR1=
2489 012500 004767 003630 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2490 012504 000460 BR T25L08 ;GO RESTORE TRAPS
2491 012506 012777 012524 170016 T25L02: MOV #T25L06,@INTVEC ;SET UP NEW INT. AREA
2492 012514 012777 002143 170002 MOV #2143,@BECR1 ;HAVE UBE DO 1 DATI NPR AND INT ON DONE
2493 012522 000001 WAIT ;WAIT TO BE INTERRUPTED
2494 012524 032777 000040 167774 T25L06: BIT #40,@BECR2 ;WAS WRONG GRANT ERROR BIT SET?
2495 012532 001015 BNE T25L07 ;BRANCH TO ERROR IF WAS
2496 012534 032777 002000 167764 BIT #2000,@BECR2 ;WAS NO GRANT OR NOT ONE GRANT BIT SET?
2497 012542 001441 BEQ T25L08 ;GO RESTORE TRAPS IF WAS NOT
2498 012544 104100 ERROR 1D64 ;ERROR: TEST OF WRONG GRANT OR NOT ONE GRANT FAILED
2499 012546 017767 167752 166440 MOV @BECR1,$REGO ;SAVE BECR1
2500 012554 104101 ERROR 1D65 ;NO GRANT OR NOT ONE GRANT ERROR BIT FALSELY SET
2501 012556 104102 ERROR 1D66 ;WITH INT ON DONE = 1
2502 012560 004767 003550 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2503 012564 000430 BR T25L08 ;GO RESTORE TRAPS
2504 012566 104100 T25L07: ERROR 1D64 ;ERROR: TEST OF WRONG GRANT OR NOT ONE GRANT FAILED
2505 012570 017767 167730 166416 MOV @BECR1,$REGO ;SAVE BECR1
2506 012576 104103 ERROR 1D67 ;WRONG GRANT ERROR BIT FALSELY SET
2507 012600 104102 ERROR 1D66 ;WITH INT ON DONE = 1
2508 012602 004767 003526 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2509 012606 000417 BR T25L08 ;GO RESTORE TRAPS
2510 012610 104100 T25L03: ERROR 1D64 ;ERROR: TEST OF WRONG GRANT OR NOT ONE GRANT FAILED
2511 012612 017767 167706 166374 MOV @BECR1,$REGO ;SAVE BECR1
2512 012620 104103 ERROR 1D67 ;WRONG GRANT ERROR BIT FALSELY SET
2513 012622 004767 003506 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2514 012626 000407 BR T25L08 ;GO RESTORE TRAPS
2515 012630 104100 T25L04: ERROR 1D64 ;ERROR: TEST OF WRONG GRANT OR NOT ONE GRANT FAILED
2516 012632 017767 167666 166354 MOV @BECR1,$REGO ;SAVE BECR1
2517 012640 104101 ERROR 1D65 ;NO GRANT OR NOT ONE GRANT ERROR BIT FALSELY SET
2518 012642 004767 003466 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2519 012646 004767 003314 T25L08: JSR PC,RCATCH ;RESTORE TRAP CATCHER

```

```

*****
: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.
*****

```

```

2530 012652 000004 TST32: SCOPE
2531 012654 012706 001100 MOV #STACK,SP ;RESTORE STACK
2532 012660 012737 000340 177776 MOV #340,@PSW ;LOCK OUT INTERRUPTS
2533 012666 004767 003242 JSR PC,CLRREG ;CLEAR UBE REG
2534 012672 012777 040000 167626 MOV #40000,@BECR2 ;SET TIME DELAY BIT
2535 012700 012777 013006 167624 MOV #T26L01,@INTVEC ;SET UP FOR INTERRUPTS
2536 012706 012777 006003 167610 MOV #6003,@BECR1 ;DO RELEASE BUS IMMED.
2537 012714 005000 CLR R0 ;INITIALIZE R0
2538 012716 005200 T26L02: INC R0 ;DELAY TO WAIT FOR
2539 012720 022700 000400 CMP #400,R0 ;BUSS LATENCY ERROR BIT
2540 012724 001374 BNE T26L02 ;TO SET

```



```

2541 012726 032777 000100 167572 BIT #100,JBECR2 ;WAS BUSS LATENCY ERROR BIT SET?
2542 012734 001004 BNE T26L03 ;BRANCH IF YES
2543 012736 104106 ERROR #D70 ;ERROR: TEST OF TIME DALAY AND BUSS LATENCY FAILED
2544 012740 104107 ERROR #D71 ;TO SET BIT 6 OF BECR2
2545 012742 004767 003366 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2546 012746 032777 100000 167550 T26L03: BIT #100000,JBECR1 ;WAS ERROR BIT SET?
2547 012754 001004 BNE T26L04 ;BRANCH IF YES
2548 012756 104106 ERROR #D70 ;ERROR: TEST OF TIME DELAY AND BUSS LATENCY FAILED
2549 012760 104075 ERROR #D61 ;TO SET BIT 15 OF BECR1
2550 012762 004767 003346 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2551 012766 005037 177776 T26L04: CLR #PSW ;ALLOW ERROR INTERRUPTS
2552 012772 000240 NOP ;UBE SHOULD INTERRUPT
2553 012774 104106 ERROR #D70 ;ERROR: TEST OF TIME DELAY AND BUSS LATENCY FAILED
2554 012776 104072 ERROR #D58 ;TO INTERRUPT CPU
2555 013000 004767 003330 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2556 013004 000412 BR T26L05 ;GO TO END OF TEST
2557 013006 005077 167516 T26L01: CLR #BERE ;CLEAR ERROR CONDITION
2558 013012 032777 000100 167506 BIT #100,JBECR2 ;WAS ERROR CLEARED?
2559 013020 001404 BEQ T26L05 ;BRANCH IF YES
2560 013022 104106 ERROR #D70 ;ERROR: TEST OF TIME DELAY AND BUSS LATENCY FAILED
2561 013024 104110 ERROR #D72 ;TO CLEAR BIT 6 OF BECR2
2562 013026 004767 003302 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2563 013032 004767 003076 T26L05: JSR PC,CLAREG ;CLEAR ALL UBE REG
2564 013036 004767 003176 JSR PC,DINT ;DISREGARD ERROR INTERRUPTS
2565 013042 012777 177777 167450 MOV #177777,JBECR ;HAVE UBE DO DATI
2566 013050 012777 027700 167444 MOV #BUFF1,JBEBR ;SO BUSS LATENCY REG
2567 013056 012777 002041 167440 MOV #2041,JBECR1 ;HOLD FLOP CLEARED
2568 013064 004767 003120 JSR PC,CRDY ;WAIT FOR RDY SET
2569 013070 005077 167434 CLR #BERE ;CLEAR LATENCY ERROR IF SET
2570 013074 004767 003066 JSR PC,RCATCH ;RESTORE TRAPS

```

```

*****
*TEST 33 TEST MULTIPLE INTERRUPTS SET RDY BIT
*****

```

```

TST33: SCOPE
MOV #STACK,SP ;INITIALIZE STACK
JSR PC,CLAREG ;CLEAR ALL UBE REG
JSR PC,DINT ;DISREGARD INTERRUPTS
CLR #PSW ;ALLOW INTERRUPTS
MOV #177776,JBECR ;HAVE UBE DO 2 CYCLES
MOV #40000,JBECR2 ;DO TIME DLY
MOV #3,JBECR1 ;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 RD COUNTS FOR A TIME >150
*MS TO SEE IF THE CPU GETS POWERED UP. THE PROGRAM

```



```

2597
2598
2599
2600 013166 000004
2601 013170 012767 000001 166036
2602 013176 032737 000020 001172
2603 013204 001516
2604 013206 012737 000340 177776
2605 013214 012706 001100
2606 013220 013746 000024
2607 013224 013746 000026
2608 013230 012737 013266 000024
2609 013236 012737 000340 000026
2610 013244 012777 000020 167254
2611 013252 000240
2612 013254 104111
2613 013256 104112
2614 013260 004767 003050
2615 013264 000450
2616 013266 022626
2617 013270 012737 013332 000024
2618 013276 005000
2619 013300 005001
2620 013302 005200
2621 013304 005700
2622 013306 001375
2623 013310 005201
2624 013312 022701 000004
2625 013316 001371
2626 013320 104111
2627 013322 104113
2628 013324 004767 003004
2629 013330 000426
2630 013332 012737 013374 000024
2631 013340 005000
2632 013342 005001
2633 013344 005200
2634 013346 005700
2635 013350 001375
2636 013352 005201
2637 013354 022701 000004
2638 013360 001371
2639 013362 104111
2640 013364 104114
2641 013366 004767 002742
2642 013372 000405
2643 013374 022626
2644 013376 012737 013406 000024
2645 013404 000001
2646 013406 032777 000020 167112
2647 013414 001004
2648 013416 104111
2649 013420 104115
2650 013422 004767 002706
2651 013426 012637 000026
2652 013432 012637 000024

```

```

:*THEN WAITS FOR A TIME >150MS TO SEE IF THE CPU
:*GETS POWERED DOWN AGAIN.
:*****
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 $#25,-($SP) ::SAVE POWER FAIL VECTOR ON STACK
MOV $T27L01,$#24 ::SET UP FOR POWER FAIL
MOV #340,$#26 ::SET UP FOR POWER FAIL
MOV #20,$BECCR2 ::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
ERROR $D73 ::ERROR: TEST OF POWER DOWN BIT FAILED
ERROR $D75 ::TO POWER UP CPU
JSR PC,TERRPC ::TYPE PC OF ERROR MSG
BR T27L02 ::RESTORE TRAPS
T27L03: MOV $T27L05,$#24 ::SET UP TO POWER DOWN AGAIN
CLR R0
CLR R1
T27L06: INC R0 ::COUNT FOR A TIME
TST R0 ::GREATER THAN 150 MS
BNE T27L06
INC R1
CMP #4,R1 ::IS TIME > 150 MS?
BNE T27L06 ::BRANCH IF NO
ERROR $D73 ::ERROR: TEST OF POWER DOWN BIT FAILED
ERROR $D76 ::TO REPOWER DOWN CPU
JSR PC,TERRPC ::TYPE PC OF ERROR MSG
BR T27L02 ::GO CHECK POWER DOWN BIT
T27L05: CMP ($SP)+,($SP)+ ::RESTORE STACK
MOV $T27L02,$#24 ::SET UP TO POWER UP AGAIN
WAIT ::WAIT TO POWER UP AGAIN
T27L02: BIT #20,$BECCR2 ::WAS POWER DOWN BIT SET?
BNE T27L07 ::BRANCH IF YES
ERROR $D73 ::ERROR: TEST OF POWER DOWN BIT FAILED
ERROR $D77 ::TO SET BIT 4 OF BECCR2
JSR PC,TERRPC ::TYPE PC OF ERROR MSG
T27L07: MOV ($SP)+,$#26 ::RESTORE POWER FAIL VECTOR
MOV ($SP)+,$#24

```



```

2653 013436 005077 167064 CLR @BECR2 ;CLEAR POWER DOWN BIT
2654
2655 ;*****
2656 ;*TEST 35 TEST DCLO CLEARS BECC, BEBA, BECR2 AND BITS 0-6, 7-15 OF BECR1
2657 ;*
2658 ;*THIS TEST IS ONLY DONE IF SW4=1.
2659 ;*****
2660 013442 000004 TST35: SCOPE
2661 013444 012767 000001 165562 MOV #1,$TIMES ;DO 1 ITERATION
2662 013452 032737 000020 001172 BIT #20,@$SWR ;SEE IF POWER DOWN TO BE TESTED
2663 013460 001002 BNE T28L10 ;BRANCH IF SW4=1
2664 013462 000167 000410 JMP TSTB ;GO TO NEXT TEST
2665 013466 012777 177777 167024 T28L10: MOV #177777,@BECC ;HAVE UBE DO 1 CYCLE
2666 013474 012777 000003 167024 MOV #3,@BECR2 ;SET ADDRESS BITS 16, 17
2667 013502 012777 160000 167012 MOV #160000,@BEBA ;LOAD UBE WITH ADDRESS THAT RETURNS NO SSYN
2668 013510 004767 002524 JSR PC,DINT ;DISREGARD INTERRUPTS
2669 013514 012777 002041 167002 MOV #2041,@BECR1 ;HAVE UBE DO DATI SO CCOVF=1 AND NSSYN ERROR = 1
2670 013522 005037 177776 CLR @PSW ;ALLOW INTERRUPTS
2671 013526 000001 WAIT ;WAIT TILL ERROR INTERRUPT
2672 013530 013746 000024 MOV @#24,-(SP) ;STORE POWER VECTOR ON STACK
2673 013534 013746 000026 MOV @#26,-(SP) ;STORE POWER VECTOR ON STACK
2674 013540 012777 177777 166754 MOV #177777,@BEBA ;LOAD ADDRESS REG WITH ALL "1"
2675 013546 012777 177777 166744 MOV #177777,@BECC ;LOAD CYCLE COUNT REG WITH ALL "1"
2676 013554 012777 077776 166742 MOV #77776,@BECR1 ;LOAD BECR1 WITH ONES
2677 013562 012737 013600 000024 MOV #T28L01,@#24 ;SET UP FOR POWER DOWN
2678 013570 012777 040037 166730 MOV #40037,@BECR2 ;LOAD BECR2 WITH ONES AND DO POWER DOWN
2679 013576 000001 WAIT ;CPU SHOULD POWER DOWN
2680 013600 022626 T28L01: CMP (SP)+,(SP)+ ;RESTORE STACK
2681 013602 012737 013612 000024 MOV #T28L05,@#24 ;SETUP FOR POWER UP
2682 013610 000001 WAIT ;CPU SHOULD POWER UP
2683 013612 042777 000020 166706 T28L05: BIC #20,@BECR2 ;CLEAR POWER DOWN BIT
2684 013620 016767 166674 165366 MOV BECC,$REG0 ;SAVE BECC ADDRESS
2685 013626 005067 165366 CLR $REG2 ;SAVE CORRECT DATA
2686 013632 005777 166662 TST @BECC ;(BECC)=0?
2687 013636 001026 BNE T28L02 ;BRANCH IF NO
2688 013640 016767 166656 165346 MOV BEBA,$REG0 ;SAVE BEBA ADDRESS
2689 013646 005777 166650 TST @BEBA ;(BEBA)=?
2690 013652 001020 BNE T28L02 ;BRANCH IF NO
2691 013654 016767 166646 165332 MOV BECR2,$REG0 ;SAVE BECR2 ADDRESS
2692 013662 005777 166640 TST @BECR2 ;WAS BECR2 CLEARED?
2693 013666 001012 BNE T28L02 ;BRANCH IF NO
2694 013670 016767 166630 165316 MOV BECR1,$REG0 ;SAVE BECR1 ADDRESS
2695 013676 012767 000200 165314 MOV #200,$REG2 ;SAVE CORRECT DATA (BECR1)
2696 013704 022777 000200 166612 CMP #200,@BECR1 ;WAS BECR1 CLEARED?
2697 013712 001407 BEQ T28L03 ;BRANCH IF YES
2698 013714 017767 165274 165274 T28L02: MOV @SREG0,$REG1 ;SAVE BAD DATA
2699 013722 104116 ERROR 1D78 ;ERROR: DCLO FAILED TO CLEAR REG
2700 013724 004767 002404 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2701 013730 000454 BR T28L04 ;GO RESTORE VECTORS
2702 013732 012737 000340 177776 T28L03: MOV #340,@PSW ;LOCK OUT INTERRUPTS
2703 013740 012777 040000 166560 MOV #40000,@BECR2 ;SET TIME DLY BIT
2704 013746 012777 006003 166550 MOV #6003,@BECR1 ;DO RELEASE BUSS IMMED. TO SET LATENCY ERROR BIT
2705 013754 032777 000100 166544 T28L06: BIT #100,@BECR2 ;TEST LATENCY ERROR BIT
2706 013762 001774 BEQ T28L06 ;WAIT TILL IT SETS
2707 013764 005037 177776 CLR @PSW ;ALLOW LATENCY ERROR INTERRUPT
2708 013770 000240 NOP ;ALLOW INTERRUPT TO BE IGNORED

```

F05

```

2709 013772 012737 014010 000024      MOV #T28L08, @#24      ;SET UP FOR POWER DOWN
2710 014000 052777 000020 166520      BIS #20, @BECR2       ;SET POWER DOWN BIT
2711 014006 000001                WAIT                  ;WAIT FOR POWER DOWN
2712 014010 022626                T28L08: CMP (SP)+, (SP)+ ;RESTORE STACK
2713 014012 012737 014022 000024      MOV #T28L09, @#24     ;SETUP FOR POWER UP
2714 014020 000001                WAIT                  ;CPU SHOULD POWER UP
2715 014022 005077 166500                T28L09: CLR @BECR2     ;CLEAR POWER DOWN BIT
2716 014026 005777 166474                TST @BECR2           ;WAS BUSS LATENCY ERROR BIT CLEARED?
2717 014032 001413                BEQ T28L04           ;BRANCH IF YES
2718 014034 016767 166466 165152      MOV BECR2, $REG0      ;SAVE REG ADDRESS
2719 014042 017767 166460 165146      MOV @BECR2, $REG1     ;SAVE REG DATA
2720 014050 005067 165144                CLR $REG2           ;SAVE CORRECT DATA
2721 014054 104116                ERROR #D78          ;ERROR: DCLO FAILED TO CLEAR REG
2722 014056 004767 002252                JSR PC, TERRPC      ;TYPE PC OF ERROR MSG
2723 014062 004767 002100                T28L04: JSR PC, RCATCH ;RESTORE TRAP CATCHER
2724 014066 012637 000026                MOV (SP)+, @#26     ;RESTORE POWER VECTOR
2725 014072 012637 000024                MOV (SP)+, @#24     ;RESTORE POWER VECTOR

```

```

2726
2727 014076      TSTB:

```

```

2728
2729
2730 *****
2731 *TEST 36      TEST SIMULTANEOUS GO ADDRESS
2732 *
2733 *THE UBE IS SETUP TO INTERRUPT ON LEVEL 7 AND
2734 *THEN TOLD TO GO VIA THE SIMULTANEOUS GO. NO
2735 *INTERRUPT INDICATES AN ERROR.
2736 *****

```

```

2736 014076 000004      †ST36: SCOPE
2737 014100 012706 001100      MOV #STACK, SP      ;RESTORE STACK
2738 014104 012737 000340 177776      MOV #340, @#PSW     ;LOCK OUT INTERRUPTS
2739 014112 004767 002016      JSR PC, CLREG       ;CLEAR ALL UBE REGS.
2740 014116 012777 014154 166406      MOV #T09L01, @INTVEC ;SETUP TO RECEIVE INTERRUPT
2741 014124 012777 000020 166372      MOV #20, @BECR1     ;SETUP TO DO BR=7
2742 014132 005277 166376      INC @BEGO           ;START SIMULTANEOUS GO
2743 014136 012737 000300 177776      MOV #300, @#PSW     ;ALLOW INTERRUPTS
2744 014144 000240                NOP                  ;UBE SHOULD INTERRUPT HERE
2745 014146 104025                ERROR #D21          ;ERROR: SIMULTANEOUS GO FAILED
2746 014150 004767 002160                JSR PC, TERRPC      ;TYPE PC OF ERROR MSG
2747 014154 004767 002006                T09L01: JSR PC, RCATCH ;RESTORE TRAP CATCHER

```

```

2748
2749
2750 *****
2751 *TEST 37      DYNAMIC TEST OF UBE
2752 *
2753 *THIS TEST EXERCISES THE MOST HARDWARE IN THE
2754 *UBE AT ONE TIME. THE EXERCISOR IS SET UP TO DO EIGHT
2755 *DATOB ON DATIP XFERS VIA NPR AND INTERRUPT ON DONE.
2756 *AFTER INTERRUPTING, A BUFFER AREA IS EXAMINED TO SEE IF
2757 *THE OPERATIONS WERE DONE PROPERLY. THE ABOVE IS THEN
2758 *REPEATED 100 TIMES.
2759 *****

```

```

2760 014160 000004      †ST37: SCOPE
2761 014162 012767 000001 165044      MOV #1, $TIMES      ;DO 1 ITERATION
2762 014170 004767 001740                JSR PC, CLREG       ;CLEAR UBE REG
2763 014174 005002                CLR R2              ;INITIALIZE COUNT
2764 014176 005037 177776                CLR @#PSW           ;ALLOW INTERRUPTS

```



```

2765 014202 012700 027700 T29L04: MOV #BUFF1,RO ;GET BUFFER ADDRESS
2766 014206 012720 052525 T29L01: MOV #52525,(RO)+ ;LOAD BUFFER
2767 014212 020027 027722 CMP RO,#BUFF1+22 ;ENTIRE BUFFER LOADED?
2768 014216 001373 BNE T29L01 ;BRANCH IF NO
2769 014220 012777 014300 166304 MOV #T29L02,INTVEC ;SET UP FOR INTERRUPTS
2770 014226 012777 027700 166266 MOV #BUFF1,JBEB ;LOAD BUFF ADDRESS IN UBE
2771 014234 012777 177760 166256 MOV #177760,JBEB ;SET UBE TO DO 16 CYCLES
2772 014242 012777 042561 166254 MOV #42561,JBEB ;DO DATOB ON DATIP AND INT. VIA BR7 WHEN DONE
2773 014250 005000 CLR RO ;INITIALIZE COUNTER
2774 014252 016767 013442 013440 T29L06: MOV BUFF1+20,BUFF1+20 ;DO BACKGROUND NOISE PATTERN
2775 014260 005200 INC RO ;WAIT FOR COUNTER RO
2776 014262 005700 TST RO ;TO OVERFLOW. IF DOES
2777 014264 001372 BNE T29L06 ;UBE FAILED TO INTERRUPT
2778 014266 104120 ERROR D80 ;ERROR: DYNAMIC TEST OF UBE FAILED
2779 014270 104072 ERROR D58 ;TO INTERRUPT CPU
2780 014272 004767 002036 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2781 014276 000432 BR T29L07 ;GO RESTORE TRAPS CATCHER
2782 014300 022626 T29L02: CMP (SP)+,(SP)+ ;RESTORE STACK
2783 014302 012700 027700 MOV #BUFF1,RO ;GET BUFFER ADDRESS
2784 014306 022710 125652 T29L05: CMP #125652,(RO) ;WAS DATA SHIFTED PROPERLY?
2785 014312 001011 BNE T29L03 ;BRANCH TO ERROR IF NO
2786 014314 005720 TST (RO)+ ;INC RO BY 2
2787 014316 022700 027720 CMP #BUFF1+20,RO ;AT END OF BUFFER?
2788 014322 001371 BNE T29L05 ;BRANCH IF NO
2789 014324 005202 INC R2 ;UPDATE COUNT
2790 014326 020227 000100 CMP R2,#100 ;WAS UBE EXERCISED 100 TIMES?
2791 014332 001323 BNE T29L04 ;BRANCH IF NO
2792 014334 000413 BR T29L07 ;RESTORE TRAPS
2793 014336 010067 164652 T29L03: MOV RO,$REG0 ;SAVE ADDRESS
2794 014342 011067 164650 MOV (RO),$REG1 ;SAVE BAD DATA
2795 014346 012767 125652 164644 MOV #125652,$REG2 ;SAVE CORRECT DATA
2796 014354 104120 ERROR D80 ;ERROR: DYNAMIC TEST OF UBE FAILED
2797 014356 104121 ERROR D81 ;TO LOAD PROPER DATA
2798 014360 004767 001750 JSR PC,TERRPC ;TYPE PC OF ERROR MSG
2799 014364 004767 001576 T29L07: JSR PC,RCATCH ;RESTORE TRAP CATCHER
2800
2801 ;////////////////////////////////////
2802 ;RETURN ROUTINE TO TEST NEXT UBE BEFORE DO LAST TEST
2803 ;////////////////////////////////////
2804 014370 000004 SCOPE ;SCOPE FOR PREVIOUS TEST
2805 014372 004767 001536 NUBE: JSR PC,CLAREG ;CLEAR UBE SO NO INT.
2806 014376 000167 166766 NUBE1: JMP ACALC ;GO SEE IF MORE UBE
2807
2808 014402 012767 014424 164530 LAST: MOV #LAST1,$LPADR ;SETUP LOOP ADDRESS FOR LAST TEST
2809 014410 012767 014424 164524 MOV #LAST1,$LPERR ;SETUP LOOP ON ERROR ADDRESS FOR LAST TEST
2810 014416 105367 164512 DECB $STNM ;ADJUST TEST NUMBER
2811
2812
2813
2814 ;*****
2815 ;*TEST 40 TEST PASSING OF GRANTS
2816 ;*
2817 ;*THIS TEST IS ONLY RUN IF THERE ARE MORE THAN ONE
2818 ;*UBE.IT IS COMPOSED OF TWO PARTS.THE FIRST PART CHECKS THAT
2819 ;*A HIGHER ELECTRICAL PRIORITY UBE WITH ALL BR LEVELS =1
2820 ;*AND GO BIT =0 WILL PASS A GRANT TO THE NEXT LOWER ONE.

```

```

2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833 014422 000004
2834 014424 005767 166124
2835 014430 001013
2836 014432 032737 010000 001172
2837 014440 001005
2838 014442 104401 027210
2839 014446 012767 000001 164560
2840 014454 000167 001332
2841
2842
2843
2844 014460 012706 001100
2845 014464 012777 014672 166060
2846 014472 016700 166054
2847 014476 012760 000340 000002
2848 014504 012777 014706 166056
2849 014512 016700 166052
2850 014516 012760 000340 000002
2851 014524 005767 166056
2852 014530 001423
2853 014532 012777 014722 166046
2854 014540 016700 166042
2855 014544 012760 000340 000002
2856 014552 005767 166046
2857 014556 001410
2858 014560 012777 014736 166036
2859 014566 016700 166032
2860 014572 012760 000340 000002
2861 014600 012700 027700
2862 014604 005001
2863 014606 012737 000340 177776
2864 014614 012777 000020 165722
2865 014622 012777 000020 165732
2866 014630 005767 165744
2867 014634 001411
2868 014636 012777 000020 165734
2869 014644 005767 165746
2870 014650 001403
2871 014652 012777 000020 165736
2872 014660 005277 165650
2873 014664 005037 177776
2874 014670 000001
2875 014672 012720 002536
2876 014676 012777 006002 165640

```

```

; *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 SEOP ; 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 @BEGO ; 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
MOV #6002, @BE1CR1 ; SETUP FIRST UBE TO DO A FUN 3

```



```

2877 014704 000421          BR T30L06          ;GO SEE IF ALL UBE INTERRUPTED
2878 014706 012720 002554 T30L03: MOV #BE28D,(R0)+ ;LOAD BUFFER WITH POINTER TO UBE ADDRESSES
2879 014712 012777 006002 165642      MOV #6002,2BE2CR1 ;SETUP SECOND UBE TO DO A FUN3
2880 014720 000413          BR T30L06          ;GO SEE IF ALL UBE INTERRUPTED
2881 014722 012720 002572          MOV #BE38D,(R0)+ ;LOAD BUFFER WITH POINTER TO UBE ADDRESS
2882 014726 012777 006002 165644      MOV #6002,2BE3CR1 ;SETUP THIRD UBE TO DO A FUN3
2883 014734 000405          BR T30L06          ;GO SEE IF ALL UBE INTERRUPTED
2884 014736 012720 002610          MOV #BE48D,(R0)+ ;LOAD BUFFER WITH POINTER TO UBE ADDRESS
2885 014742 012777 006002 165646      MOV #6002,2BE4CR1 ;SETUP FOURTH UBE TO DO A FUN3
2886 014750 022626          T30L06: CMP (SP)+,(SP)+ ;RESTORE STACK
2887 014752 005201          INC R1             ;COUNT INTERRUPTS
2888 014754 020167 165646      CMP R1,UCNT       ;HAVE ALL EXERCISORS INTERRUPTED?
2889 014760 001403          BEQ T30L22        ;BRANCH IF YES
2890 014762 005037 177776      CLR 2#PSW         ;ALLOW NEXT UBE TO INTERRUPT
2891 014766 000001          WAIT              ;WAIT FOR INTERRUPT
2892 014770 024040          T30L22: CMP -(R0),-(R0) ;DECREMENT RO BY 4
2893 014772 011067 012712      MOV (R0),BUFF1+10 ;PUT NEXT TO LOWEST PRIORITY POINTER IN BUFF1+10
2894
2895          ;BUFFER NOW CONTAINS VECTORS IN ORDER OF ELECTRICAL PRIORITY
2896
2897          ;PART 1
2898
2899 014776 016700 165624      MOV UCNT,R0       ;GET COUNT OF UBE
2900 015002 005300          DEC RO            ;ADJUST COUNT
2901 015004 005001          CLR R1           ;CLEAR INDEX REG
2902 015006 016102 027700 T30L28: MOV BUFF1(R1),R2 ;GET PTER TO ADDRESS OF HIGHER PRIORITY UBE
2903 015012 012772 000036 000006      MOV #36,26(R2)   ;SET ALL BR =1 IN THIS UBE
2904 015020 005721          TST (R1)+        ;UPDATE INDEX
2905 015022 016103 027700      MOV BUFF1(R1),R3 ;GET PTER TO ADDRESS OF NEXT LOWER PRIORITY UBE
2906 015026 012773 015144 000014      MOV #T30L25,214(R3) ;SET UP FOR INT.
2907 015034 012773 000002 000006 T30L30: MOV #2,26(R3) ;SETUP LOWER PRIORITY UBE FOR BR4
2908 015042 005273 000006 T30L26: INC 26(R3) ;HAVE UBE INT.
2909 015046 005037 177776      CLR 2#PSW        ;ALLOW INT.
2910 015052 000240          NOP              ;SHOULD INT. HERE
2911 015054 012737 000340 177776      MOV #340,2#PSW   ;LOCK OUT INT.
2912 015062 104122          T30L29: ERROR 1D82 ;ERROR:TEST OF PASSING GRANTS FAILED
2913 015064 032737 020000 001172      BIT #SW13,2#SWR  ;INHIBIT ERROR TYPEOUTS?
2914 015072 001022          BNE 15           ;BRANCH IF YES
2915 015074 016367 000014 164112      MOV 14(R3),SREG0 ;SAVE INT. VECTOR
2916 015102 104401 027062          TYPE ,MSG7       ;UBE WITH INT. VECTOR:
2917 015106 016746 164102      MOV SREG0,-(SP)  ;SAVE SREG0 FOR TYPEOUT
2918 015112 104402          TYPOC           ;GO TYPE--OCTAL ASCII(ALL DIGITS)
2919 015114 017367 000006 164074      MOV 26(R3),SREG1 ;SAVE (BECR1)
2920 015122 104401 026041          TYPE ,DH65       ;WITH BECR1=
2921 015126 016746 164064      MOV SREG1,-(SP) ;SAVE SREG1 FOR TYPEOUT
2922 015132 104402          TYPOC           ;GO TYPE--OCTAL ASCII(ALL DIGITS)
2923 015134 104401 027155          TYPE ,MSG10      ;SHOULD HAVE INT.
2924 015140 000167 000474          JMP T30L12       ;GO TO END OF TEST
2925 015144 006373 000006 T30L25: ASL 26(R3) ;DO NEXT BR LEVEL
2926 015150 042773 000400 000006      BIC #400,26(R3) ;CLEAR SHIFTED RDY BIT
2927 015156 032773 000040 000006      BIT #40,26(R3)  ;ALL BR TESTED?
2928 015164 001726          BEQ T30L26       ;BRANCH IF NO
2929
2930 015166 012773 015212 000014      MOV #T30L27,214(R3) ;SETUP FOR INT.
2931 015174 012772 015062 000014      MOV #T30L29,214(R2) ;SETUP FOR ERROR INT.
2932 015202 012772 000001 000006      MOV #1,26(R2)   ;HAVE HIGHER UBE TRY TO INT.

```

2933	015210	000711			BR T30L30		;LET LOWER UBE INT.
2934							
2935	015212	006373	000006		T30L27: ASL 26(R3)		;DO NEXT LEVEL BR
2936	015216	042773	000400	000006	BIC #400,26(R3)		;CLEAR SHIFTED RDY.
2937	015224	032773	000040	000006	BIT #40,26(R3)		;ALL BR TESTED?
2938	015232	001703			BEQ T30L26		;BRANCH IF NO
2939	015234	012772	006003	000006	MOV #6003,26(R2)		;HAVE HIGHER UBE DO FUN3
2940	015242	005037	177776		CLR 2#PSW		;ALLOW REQUESTS
2941	015246	105772	000006		15: TSTB 26(R2)		;IS UBE DONE?
2942	015252	100375			BPL 15		;BRANCH IF NO
2943	015254	012737	000340	177776	MOV #340,2#PSW		;SET LEVEL =7
2944	015262	005300			DEC R0		;ADJUST UBE COUNT
2945	015264	005700			TST R0		;ALL UBE TESTED?
2946	015266	001247			BNE T30L28		;BRANCH IF NO
2947							
2948							
2949							
2950	015270	012700	000510		MOV #510,R0		;GET FIRST POSSIBLE VECTOR AREA
2951	015274	012720	015450		T30L09: MOV #T30L08,(R0)+		;SET UP VECTOR AREA TO HANDLE DOUBLE INTERRUPTS
2952	015300	012720	000340		MOV #340,(R0)+		;SET PRIORITY = 7
2953	015304	022700	001000		CMP #1000,R0		;AT END OF AREA?
2954	015310	001371			BNE T30L09		;BRANCH IF NO
2955	015312	016700	012362		MOV BUFF1,R0		;GET HIGHEST PRIORITY UBE ADDRESS POINTER
2956	015316	016701	012360		MOV BUFF1+2,R1		;GET NEXT PRIORITY UBE ADDRESS POINTER
2957	015322	012770	000002	000006	T30L14: MOV #2,26(R0)		;HAVE HIGHER PRIORITY UBE DO BR4
2958	015330	012771	000004	000006	MOV #4,26(R1)		;HAVE NEXT LOWER ELEC. PRIORITY UBE DO BR5
2959	015336	012770	015450	000014	MOV #T30L08,214(R0)		;SET UP HIGHER PRIORITY UBE VECTOR FOR DOUBLE INT.
2960	015344	012771	015364	000014	MOV #T30L10,214(R1)		;SET UP FOR INTERRUPT FROM NEXT LOWER ELEC. PRIORITY UBE
2961	015352	005277	165156		T30L11: INC 2BEGO		;START INTERRUPT
2962	015356	005037	177776		CLR 2#PSW		;ALLOW INTERRUPTS
2963	015362	000001			WAIT		
2964	015364	022626			T30L10: CMP (SP)+,(SP)+		;RESTORE STACK
2965	015366	006371	000006		ASL 26(R1)		;HAVE NEXT PRIORITY UBE INT. ONE LEVEL HIGHER
2966	015372	042771	000400	000006	BIC #400,26(R1)		;CLEAR SHIFTED RDY
2967	015400	032771	000040	000006	BIT #40,26(R1)		;TESTED ALL BR LEVELS?
2968	015406	001761			BEQ T30L11		;BRANCH IF NO
2969	015410	020067	012274		CMP R0,BUFF1+10		;TESTED ALL UBE POSSIBLE?
2970	015414	001511			BEQ T30L12		;BRANCH IF YES TO CLEAR BECR1 AND RESTORE TRAPS
2971	015416	020067	012256		CMP R0,BUFF1		;JUST TESTED FIRST UBE?
2972	015422	001005			BNE T30L13		;BRANCH IF NO
2973	015424	016700	012252		MOV BUFF1+2,R0		;TEST SECOND HIGHEST PRIORITY UBE
2974	015430	016701	012250		MOV BUFF1+4,R1		;GET THIRD HIGHEST PRIORITY UBE
2975	015434	000732			BR T30L14		;GO TEST SECOND HIGHEST PRIORITY UBE
2976	015436	016700	012242		T30L13: MOV BUFF1+4,R0		;TEST THIRD HIGHEST PRIORITY UBE
2977	015442	016701	012240		MOV BUFF1+6,R1		;GET FOURTH HIGHEST PRIORITY UBE
2978	015446	000725			BR T30L14		;GO TEST THIRD HIGH PRIORITY UBE
2979	015450	022626			T30L08: CMP (SP)+,(SP)+		;RESTORE STACK
2980	015452	016067	000014	163534	MOV 14(R0),SREG0		;SAVE INTERRUPT VECTOR OF BAD UBE
2981	015460	012767	000004	163530	MOV #4,SREG1		;SAVE BAD BR LEVEL
2982	015466	016167	000014	163524	MOV 14(R1),SREG2		;SAVE NEXT HIGHER PRIORITY UBE VECTOR
2983	015474	032771	000004	000006	BIT #4,26(R1)		;WAS BR=5?
2984	015502	001404			BEQ T30L15		;BRANCH IF NO
2985	015504	012767	000005	163510	MOV #5,SREG3		;BR=5
2986	015512	000413			BR T30L17		;GO INDICATE ERROR
2987	015514	032771	000010	000006	T30L15: BIT #10,26(R1)		;WAS BR=6?
2988	015522	001404			BEQ T30L16		;BRANCH IF NO

2989	015524	012767	000006	163470		MOV #6,\$REG3		;INDICATE BR=6
2990	015532	000403				BR T30L17		;GO INDICATE ERROR
2991	015534	012767	000007	163460	T30L16:	MOV #7,\$REG3		;INDICATE BR=7
2992	015542	104122			T30L17:	ERROR 1082		;ERROR: TEST OF PASSING GRANTS FAILED
2993	015544	032737	020000	001172		BIT #SW13,@#SWR		;INHIBIT ERROR TYPEOUTS?
2994	015552	001032				BNE T30L12		;BRANCH IF YES
2995	015554	104401	027062			TYPE ,MSG7		;TYPE FAILING UBE VECTOR
2996	015560	016746	163430			MOV \$REG0,-(SP)		;SAVE \$REG0 FOR TYPEOUT
2997	015564	104402				TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
2998	015566	104401	027104			TYPE ,MSG8		;TYPE FAILING UBE BR LEVEL
2999	015572	016746	163420			MOV \$REG1,-(SP)		;SAVE \$REG1 FOR TYPEOUT
3000	015576	104402				TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
3001	015600	104401	027121			TYPE ,MSG9		
3002	015604	104401	027062			TYPE ,MSG7		;TYPE UBE USED TO TEST FAILING ONE
3003	015610	016746	163404			MOV \$REG2,-(SP)		;SAVE \$REG2 FOR TYPEOUT
3004	015614	104402				TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
3005	015616	104401	027104			TYPE ,MSG8		;TYPE BR LEVEL TESTING
3006	015622	016746	163374			MOV \$REG3,-(SP)		;SAVE \$REG3 FOR TYPEOUT
3007	015626	104402				TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
3008	015630	104401	027155			TYPE ,MSG10		
3009	015634	004767	000474			JSR PC,TERRPC		;TYPE PC OF ERROR MSG
3010	015640	012777	006003	164676	T30L12:	MOV #6003,@BE1CR1		;SETUP UBE TO DO A FUN3
3011	015646	012777	006003	164706		MOV #6003,@BE2CR1		;SETUP UBE TO DO A FUN3
3012	015654	005767	164720			TST BE3CR1		;ARE THERE 3 UBE?
3013	015660	001411				BEQ 1\$;BRANCH IF NO
3014	015662	012777	006003	164710		MOV #6003,@BE3CR1		;SETUP UBE TO DO A FUN3
3015	015670	005767	164722			TST BE4CR1		;ARE THERE 4 UBE?
3016	015674	001403				BEQ 1\$;BRANCH IF NO
3017	015676	012777	006003	164712		MOV #6003,@BE4CR1		;SETUP UBE TO DO A FUN3
3018	015704	005037	177776		1\$:	CLR @PSW		;ALLOW ALL UBE TO DO FUN3
3019	015710	105777	164630		2\$:	TSTB @BE1CR1		;FIRST UBE DONE?
3020	015714	100375				BPL 2\$;BRANCH IF NO
3021	015716	105777	164640		3\$:	TSTB @BE2CR1		;SECOND UBE DONE?
3022	015722	100375				BPL 3\$;BRANCH IF NO
3023	015724	005767	164650			TST BE3CR1		;ARE THERE THREE UBE?
3024	015730	001411				BEQ 6\$;BRANCH IF NO
3025	015732	105777	164642		4\$:	TSTB @BE3CR1		;THIRD UBE DONE?
3026	015736	100375				BPL 4\$;BRANCH IF NO
3027	015740	005767	164652			TST BE4CR1		;ARE THERE 4 UBE?
3028	015744	001403				BEQ 6\$;BRANCH IF NO
3029	015746	105777	164644		5\$:	TSTB @BE4CR1		;FOURTH UBE DONE?
3030	015752	100375				BPL 5\$;BRANCH IF NO
3031								
3032								
3033								
3034	015754	012700	000510		6\$:	MOV #510,R0		;GET FIRST VECTOR ADDRESS
3035	015760	012701	000512			MOV #512,R1		
3036	015764	010120			T30L20:	MOV R1,(R0)+		;PUT ADDRESS OF NEXT LOC IN THIS ONE
3037	015766	005020				CLR (R0)+		;PUT HALT IN NEXT LOCATION
3038	015770	022121				CMP (R1)+,(R1)+		;INC R1 BY 4
3039	015772	020027	001000			CMP R0,#1000		;AT END OF VECTOR AREA?
3040	015776	001372				BNE T30L20		;BRANCH IF NO
3041	016000	005767	163126			TST \$PASS		;FIRST PASS OF PROGRAM?
3042	016004	001002				BNE \$EOP		;BRANCH IF NO
3043	016006	104401	020253			TYPE ,MSG2		;ALL EXERCISORS TESTED
3044								;NOTE: TO TEST PASSING OF GRANTS FOR

; THE LAST UBE, IT SHOULD BE
; SWAPPED WITH A UBE OF HIGHER
; ELECTRICAL PRIORITY

3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100

016012
016012 000004
016014 005067 163114
016020 005067 163210
016024 005267 163102
016030 042767 100000 163074
016036 005327
016040 000001
016042 003022
016044 012737
016046 000001
016050 016040
016052 104401 016117
016056 016746 163050
016062 104405
016064 104401 016114
016070 013700 000042
016074 001405
016076 000005
016100 004710
016102 000240
016104 000240
016106 000240
016110
016110 000137
016112 003100
016114 377 377 000
016117 015 042412 042116
016124 050040 051501 020123
016132 000043
016134 005077 164370
016140 005077 164362
016144 005077 164354
016150 005077 164346
016154 005077 164340
016160 005077 164332
016164 000207

.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

\$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
TYPE \$SENDMG ; 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,RO ; GET MONITOR ADDRESS
BEQ \$DOAGN ; BRANCH IF NO MONITOR
RESET ; CLEAR THE WORLD
\$SENDAD: JSR PC,(RO) ; 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
\$SENDMG: .ASCIZ <15><12>/END PASS #/

////////////////////////////////////
; SUBROUTINE TO CLEAR ALL UBE REG
////////////////////////////////////

CLREG: 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 @BEED ; CLEAR BEED REG
RTS PC ; RETURN

////////////////////////////////////
; SUBROUTINE TO RESTORE TRAP CATCHER TO UBE VECTOR AREA
////////////////////////////////////

3101 016166 010546
3102 016170 016705 164336
3103 016174 005725
3104 016176 010577 164330
3105 016202 005015
3106 016204 012605
3107 016206 000207
3108
3109
3110
3111
3112
3113

```
RCATCH: MOV R5, -(SP)          ;SAVE R5 ON STACK
          MOV INTVEC, R5       ;GET INT. VECTOR
          TST (R5)+            ;CALC. INTVEC+2
          MOV R5, @INTVEC      ;PUT INTVEC+2 IN INTVEC
          CLR (R5)             ;PUT HALT IN INTVEC+2
          MOV (SP)+, R5        ;RESTORE R5
          RTS PC
```

3114 016210 005004
3115 016212 005005
3116 016214 005205
3117 016216 105777 164302
3118 016222 100405
3119 016224 032705 000200
3120 016230 001771
3121 016232 012704 000001
3122 016236 000207
3123
3124
3125
3126

```
//////
;SUBROUTINE TO CHECK IF RDY BIT SET
//////
CRDY: CLR R4
      CLR R5
2$: INC R5          ;UPDATE COUNT
    TSTB @BECR1    ;SEE IF RDY SET
    BMI 1$         ;BRANCH IF SET
    BIT #200, R5   ;WAITED >100 MICROSECS?
    BEQ 2$         ;CONTINUE TO LOOK FOR RDY IF R5 NOT =128
    MOV #1, R4     ;SET R4=1 TO INDICATE ERROR
1$: RTS PC        ;RETURN
```

3127 016240 016705 164266
3128 016244 005725
3129 016246 010577 164260
3130 016252 012715 000002
3131 016256 000207
3132
3133
3134

```
//////
;SUBROUTINE TO DISREGARD UBE INTERRUPTS
//////
DINT: MOV INTVEC, R5          ;GET INTVEC AND
      TST (R5)+              ;CALC. INTVEC+2
      MOV R5, @INTVEC        ;PUT ADDRESS OF NEXT LOC IN THIS ONE
      MOV #2, (R5)           ;PUT AN RTI IN INTVEC+2
      RTS PC
```

3135 016260 016705 162740
3136 016264 005004
3137 016266 014524
3138 016270 022704 000060
3139 016274 001374
3140 016276 012705 000062
3141 016302 019524
3142 016304 005024
3143 016306 022525
3144 016310 022704 001000
3145 016314 001372
3146 016316 012737 000137 000200
3147 016324 012737 002632 000202
3148 016332 000207
3149
3150

```
//////
;SUBROUTINE TO RESTORE VECTOR AREA 0-56 FROM STACK AREA AND PUT TRAP CATCHER IN REST
//////
RVEC: MOV $TMP0, R5          ;GET AREA WHERE VECTOR STORED
      CLR R4                 ;SET R4 =TO FIRST LOC
1$: MOV -(R5), (R4)+         ;RESTORE VECTORS
    CMP #60, R4             ;AT END OF AREA?
    BNE 1$                  ;BRANCH IF NO
    MOV #62, R5             ;SET R5=TO FIRST TRAP CATCHER ADDRESS
2$: MOV R5, (R4)+           ;PUT ADDRESS OF NEXT LOC IN THIS ONE
    CLR (R4)+               ;PUT HALT IN NEXT LOC
    CMP (R5)+, (R5)+        ;INC R5 BY 4
    CMP #1000, R4           ;AT END OF ENTIRE VECTOR AREA?
    BNE 2$                  ;BRANCH IF NO
    MOV #137, @#200         ;RESTORE JMP @#START TO LOC 200
    MOV #START, @#202
    RTS PC                  ;RETURN
```

3151
3152 016334 032737 020000 001172
3153 016342 001011
3154 016344 104401 027535
3155 016350 016746 162574
3156 016354 104402

```
//////
;SUBROUTINE TO TYPE PC OF ERROR MESSAGE
//////
TERRPC: BIT #SW13, @#SWR     ;INHIBITS ERROR TYPSETS?
        BNE 1$              ;BRANCH IF YES
        TYPE ,MSG15         ;PC OF ERROR MSG WAS:
        MOV $ERRPC, -(SP)   ;SAVE $ERRPC FOR TYPSET
        TYP0C              ;GO TYP--OCTAL ASCII(ALL DIGITS)
```

```

3157 016356 104401 001245
3158 016362 104401 001245
3159 016366 000207
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174 016370
3175 016370 032777 040000 162574
3176 016376 001101
3177
3178 016400 000416
3179
3180 016402 013746 000004
3181 016406 012737 016426 000004
3182 016414 005737 177060
3183 016420 012637 000004
3184 016424 000453
3185 016426 022626
3186 016430 012637 000004
3187 016434 000413
3188 016436
3189 016436 105767 162473
3190 016442 001421
3191 016444 126767 162477 162463
3192 016452 101015
3193 016454 032777 001000 162510
3194 016462 001404
3195 016464 016767 162452 162446
3196 016472 000443
3197 016474 105067 162435
3198 016500 005067 162530
3199 016504 000415
3200 016506 032777 004000 162456
3201 016514 001011
3202 016516 005767 162410
3203 016522 001406
3204 016524 005267 162406
3205 016530 026767 162500 162400
3206 016536 002021
3207 016540 012767 000001 162370
3208 016546 016767 000044 162460
3209 016554 105267 162354
3210 016560 011667 162354
3211 016564 011667 162352
3212 016570 005067 162442

```

```

TYPE , $CRLF
TYPE , $CRLF
1$: RTS PC

.SBTTL SCOPE HANDLER ROUTINE

;*****
;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
;AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
;AND LOAD THE ERROR FLAG ($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

$SCOPE:
1$: BIT #BIT14, $SWR ;; LOOP ON PRESENT TEST?
BNE $OVER ;; YES IF SW14=1
;*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR 6$ ;; IF RUNNING ON THE "XOR" TESTER CHANGE
; THIS INSTRUCTION TO A "NOP" (NOP=240)
MOV @#ERRVEC, -(SP) ;; SAVE THE CONTENTS OF THE ERROR VECTOR
MOV #5$, @#ERRVEC ;; SET FOR TIMEOUT
TST @#177060 ;; TIME OUT ON XOR?
MOV (SP)+, @#ERRVEC ;; RESTORE THE ERROR VECTOR
BR $SVLAD ;; GO TO THE NEXT TEST
5$: CMP (SP)+, (SP)+ ;; CLEAR THE STACK AFTER A TIME OUT
MOV (SP)+, @#ERRVEC ;; RESTORE THE ERROR VECTOR
BR 7$ ;; LOOP ON THE PRESENT TEST
6$; ;*****END OF CODE FOR THE XOR TESTER*****
2$: TSTB $ERFLG ;; HAS AN ERROR OCCURRED?
BEQ 3$ ;; BR IF NO
CMPB $ERMAX, $ERFLG ;; MAX. ERRORS FOR THIS TEST OCCURRED?
BHI 3$ ;; BR IF NO
BIT #BIT09, $SWR ;; LOOP ON ERROR?
BEQ 4$ ;; BR IF NO
7$: MOV $LPERR, $LPADR ;; SET LOOP ADDRESS TO LAST SCOPE
BR $OVER
4$: CLRB $ERFLG ;; ZERO THE ERROR FLAG
CLR $TIMES ;; CLEAR THE NUMBER OF ITERATIONS TO MAKE
BR 1$ ;; ESCAPE TO THE NEXT TEST
3$: BIT #BIT11, $SWR ;; INHIBIT ITERATIONS?
BNE 1$ ;; BR IF YES
TST $PASS ;; IF FIRST PASS OF PROGRAM
BEQ 1$ ;; INHIBIT ITERATIONS
INC $ICNT ;; INCREMENT ITERATION COUNT
CMP $TIMES, $ICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
BGE $OVER ;; BR IF MORE ITERATION REQUIRED
1$: MOV #1, $ICNT ;; REINITIALIZE THE ITERATION COUNTER
MOV $MXCNT, $TIMES ;; SET NUMBER OF ITERATIONS TO DO
$SVLAD: INCB $STNM ;; COUNT TEST NUMBERS
MOV (SP), $LPADR ;; SAVE SCOPE LOOP ADDRESS
MOV (SP), $LPERR ;; SAVE ERROR LOOP ADDRESS
CLR $ESCAPE ;; CLEAR THE ESCAPE FROM ERROR ADDRESS

```



```

000001 016574 112767 000001 162345
000002 016602 016777 162326 162364
000003 016610 016716 162324
000004 016614 000002
000005 016616 000012
000006
000007
000008
000009
000010
000011
000012
000013
000014
000015
000016
000017
000018
000019
000020
000021
000022
000023
000024
000025
000026
000027
000028
000029
000030
000031
000032
000033
000034
000035
000036
000037
000038
000039
000040
000041
000042
000043
000044
000045
000046
000047
000048
000049
000050
000051
000052
000053
000054
000055
000056
000057
000058
000059
000060
000061
000062
000063
000064
000065
000066
000067
000068

```

```

MOV #1,$ERMAX          ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
$OVER: MOV $STNM,$DISPLAY ;; DISPLAY TEST NUMBER
      MOV $LPADR,(SP)    ;; FUDGE RETURN ADDRESS
      RTI                ;; FIXES PS
$MXCNT: 10              ;; MAX. NUMBER OF ITERATIONS
.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

```

```

$ERROR:
7$: INCB $ERFLG          ;; SET THE ERROR FLAG
   BEQ 7$                ;; DON'T LET THE FLAG GO TO ZERO
   MOV $STNM,$DISPLAY    ;; DISPLAY TEST NUMBER AND ERROR FLAG
   BIT #BIT10,$SWR       ;; BELL ON ERROR?
   BEQ 1$                ;; NO - SKIP
   TYPE $BELL            ;; RING BELL
1$: INC $ERTTL           ;; COUNT THE NUMBER OF ERRORS
   MOV (SP),$ERRPC       ;; GET ADDRESS OF ERROR INSTRUCTION
   SUB #2,$ERRPC
   MOVB $ERRPC,$ITEMB    ;; STRIP AND SAVE THE ERROR ITEM CODE
   BIT #BIT13,$SWR       ;; SKIP TYPEOUT IF SET
   BNE 20$              ;; SKIP TYPEOUTS
   JSR PC,$ERRTYP        ;; GO TO USER ERROR ROUTINE
   TYPE $SCLF
20$:
2$: TST $SWR             ;; HALT ON ERROR
   BPL 3$                ;; SKIP IF CONTINUE
   HALT                  ;; HALT ON ERROR!
3$: BIT #BIT09,$SWR      ;; LOOP ON ERROR SWITCH SET?
   BEQ 4$                ;; BR IF NO
   MOV $LPERR,(SP)       ;; FUDGE RETURN FOR LOOPING
4$: TST $ESCAPE          ;; CHECK FOR AN ESCAPE ADDRESS
   BEQ 5$                ;; BR IF NONE
   MOV $ESCAPE,(SP)      ;; FUDGE RETURN ADDRESS FOR ESCAPE
5$: CMP #SENDAD,$#42     ;; ACT-11 AUTO-ACCEPT?
   BNE 6$                ;; BRANCH IF NO
   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.

```



```

3269
3270 016766 104401 001245 $ERRTYP: TYPE $SCLF :: "CARRIAGE RETURN" & "LINE FEED"
3271 016766 104401 001245 MOV RO,-(SP) :: SAVE RO
3272 016772 010046 CLR RO :: PICKUP THE ITEM INDEX
3273 016774 005000 BISS 3*$ITEMB,RO
3274 016776 153700 001146 SNE 1$ :: IF ITEM NUMBER IS ZERO, JUST
3275 017002 001004 MOV $ERRPC,-(SP) :: TYPE THE PC OF THE ERROR
3276
3277 017004 016746 162140 SAVE $ERRPC FOR TYPEOUT
3278 ERROR ADDRESS
3279 017010 104402 TYPOC :: GO TYPE--OCTAL ASCII(ALL DIGITS)
3280 017012 000426 BR 6$ :: GET OUT
3281 017014 005300 1$: DEC RO :: ADJUST THE INDEX SO THAT IT WILL
3282 017016 006300 ASL RO WORK FOR THE ERROR TABLE
3283 017020 006300 ASL RO
3284 017022 006300 ASL RO
3285 017024 062700 001250 ADD #$ERRTB,RO :: FORM TABLE POINTER
3286 017030 012067 000004 MOV (RO)+,2$ :: PICKUP "ERROR MESSAGE" POINTER
3287 017034 001404 BEQ 3$ :: SKIP TYPEOUT IF NO POINTER
3288 017036 104401 TYPE :: TYPE THE "ERROR MESSAGE"
3289 017040 000000 2$: .WORD 0 "ERROR MESSAGE" POINTER GOES HERE
3290 017042 104401 001245 TYPE $SCLF :: "CARRIAGE RETURN" & "LINE FEED"
3291 017046 012067 000004 3$: MOV (RO)+,4$ :: PICKUP "DATA HEADER" POINTER
3292 017052 001404 BEQ 5$ :: SKIP TYPEOUT IF 0
3293 017054 104401 TYPE :: TYPE THE "DATA HEADER"
3294 017056 000000 4$: .WORD 0 "DATA HEADER" POINTER GOES HERE
3295 017060 104401 001245 TYPE $SCLF :: "CARRIAGE RETURN" & "LINE FEED"
3296 017064 011000 5$: MOV (RO),RO :: PICKUP "DATA TABLE" POINTER
3297 017066 001004 BNE 7$ :: GO TYPE THE DATA
3298 017070 012600 6$: MOV (SP)+,RO :: RESTORE RO
3299 017072 104401 001245 TYPE $SCLF :: "CARRIAGE RETURN" & "LINE FEED"
3300 017076 000207 RTS PC :: RETURN
3301 017100 7$:
3302 017100 013046 MOV 3(RO)+,-(SP) :: SAVE 3(RO)+ FOR TYPEOUT
3303 017102 104402 TYPOC :: GO TYPE--OCTAL ASCII(ALL DIGITS)
3304 017104 005710 TST (RO) :: IS THERE ANOTHER NUMBER?
3305 017106 001770 BEQ 6$ :: BR IF NO
3306 017110 104401 017116 TYPE 8$ :: TYPE TWO(2) SPACES
3307 017114 000771 BR 7$ :: LOOP
3308 017116 020040 000 8$: .ASCIZ / / :: TWO(2) SPACES
3309 017122 .EVEN
3310 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
3311
3312 *****
3313 *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
3314 *SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
3315 *NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
3316 *BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
3317 *REPLACED WITH SPACES.
3318 *CALL:
3319 * MOV NUM,-(SP) :: PUT THE BINARY NUMBER ON THE STACK
3320 * TYPDS :: GO TO THE ROUTINE
3321
3322 $TYPDS:
3323 017122 MOV RO,-(SP) :: PUSH RO ON STACK
3324 017124 MOV R1,-(SP) :: PUSH R1 ON STACK

```



```

3325 017126 010246      MOV      R2,-(SP)      ;; PUSH R2 ON STACK
3326 017130 010346      MOV      R3,-(SP)      ;; PUSH R3 ON STACK
3327 017132 010546      MOV      R5,-(SP)      ;; PUSH R5 ON STACK
3328 017134 012746 020200  MOV      #20200,-(SP)  ;; SET BLANK SWITCH AND SIGN
3329 017140 016605 000020  MOV      20(SP),R5    ;; GET THE INPUT NUMBER
3330 017144 100004      BPL      1$           ;; BR IF INPUT IS POS.
3331 017146 005405      NEG      R5           ;; MAKE THE BINARY NUMBER POS.
3332 017150 112766 000055 000001  MOVB     #'-,1(SP)    ;; MAKE THE ASCII NUMBER NEG.
3333 017156 005000      CLR      R0           ;; ZERO THE CONSTANTS INDEX
3334 017160 012703 017336  MOV      #SDBLK,R3    ;; SETUP THE OUTPUT POINTER
3335 017164 112723 000040  MOVB     #' ,(R3)+    ;; SET THE FIRST CHARACTER TO A BLANK
3336 017170 005002      CLR      R2           ;; CLEAR THE BCD NUMBER
3337 017172 016001 017326  MOV      $DTBL(R0),R1 ;; GET THE CONSTANT
3338 017176 160105      SUB      R1,R5        ;; FORM THIS BCD DIGIT
3339 017200 002402      BLT     4$           ;; BR IF DONE
3340 017202 005202      INC      R2           ;; INCREASE THE BCD DIGIT BY 1
3341 017204 000774      BR      3$           ;;
3342 017206 060105      ADD      R1,R5        ;; ADD BACK THE CONSTANT
3343 017210 005702      TST     R2           ;; CHECK IF BCD DIGIT=0
3344 017212 001002      BNE     5$           ;; FALL THROUGH IF 0
3345 017214 105716      TSTB    (SP)         ;; STILL DOING LEADING 0'S?
3346 017216 100407      BMI     7$           ;; BR IF YES
3347 017220 106316      ASLB    (SP)         ;; MSD?
3348 017222 103003      BCC     6$           ;; BR IF NO
3349 017224 116663 000001 177777  MOVB     1(SP),-1(R3) ;; YES--SET THE SIGN
3350 017232 052702 000060 6$:      BIS     #'0,R2        ;; MAKE THE BCD DIGIT ASCII
3351 017236 052702 000040 7$:      BIS     #' ,R2        ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
3352 017242 110223      MOVB    R2,(R3)+     ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
3353 017244 005720      TST     (R0)+        ;; JUST INCREMENTING
3354 017246 020027 000010  CMP     R0,#10       ;; CHECK THE TABLE INDEX
3355 017252 002746      BLT     2$           ;; GO DO THE NEXT DIGIT
3356 017254 003002      BGT     8$           ;; GO TO EXIT
3357 017256 010502      MOV     R5,R2        ;; GET THE LSD
3358 017260 000764      BR      6$           ;; GO CHANGE TO ASCII
3359 017262 105726 9$:      TSTB    (SP)+        ;; WAS THE LSD THE FIRST NON-ZERO?
3360 017264 100003      BPL     9$           ;; BR IF NO
3361 017266 116663 177777 177776 9$:      MOVB    -1(SP),-2(R3) ;; YES--SET THE SIGN FOR TYPING
3362 017274 105013      CLRB    (R3)         ;; SET THE TERMINATOR
3363 017276 012605      MOV     (SP)+,R5     ;; POP STACK INTO R5
3364 017300 012603      MOV     (SP)+,R3     ;; POP STACK INTO R3
3365 017302 012602      MOV     (SP)+,R2     ;; POP STACK INTO R2
3366 017304 012601      MOV     (SP)+,R1     ;; POP STACK INTO R1
3367 017306 012600      MOV     (SP)+,R0     ;; POP STACK INTO R0
3368 017310 104401 017336  TYPE    $SDBLK       ;; NOW TYPE THE NUMBER
3369 017314 016666 000002 000004  MOV     2(SP),4(SP)  ;; ADJUST THE STACK
3370 017322 012616      MOV     (SP)+,(SP)
3371 017324 000002      RTI
3372 017326 023420      SDBLK: 10000.
3373 017330 001750      1000.
3374 017332 000144      100.
3375 017334 000012      10.
3376 017336 000004      SDBLK: .BLKW 4
3377      .SBTTL TYPE ROUTINE
3378
3379      ;;*****
3380      ;;*ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.

```

```

3381      ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
3382      ;*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
3383      ;*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
3384      ;*NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
3385      ;*
3386      ;*CALL:
3387      ;*1) USING A TRAP INSTRUCTION
3388      ;*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
3389      ;*OR
3390      ;*      TYPE
3391      ;*      MESADR
3392      ;*
3393
3394 017346 105767 161637      $TYPE:  TSTB      $TFPLG      ;; IS THERE A TERMINAL?
3395 017352 100002           BPL      1$           ;; BR IF YES
3396 017354 000000           HALT           ;; HALT HERE IF NO TERMINAL
3397 017356 000407           BR      3$           ;; LEAVE
3398 017360 010046           1$:  MOV      RO,-(SP)  ;; SAVE RO
3399 017362 017600 000002   2$:  MOV      22(SP),RO  ;; GET ADDRESS OF ASCIZ STRING
3400 017366 112046           MOV      (RO)+,-(SP)  ;; PUSH CHARACTER TO BE TYPED ONTO STACK
3401 017370 001005           BNE      4$           ;; BR IF IT ISN'T THE TERMINATOR
3402 017372 005726           TST      (SP)+        ;; IF TERMINATOR POP IT OFF THE STACK
3403 017374 012600           60$: MOV      (SP)+,RO  ;; RESTORE RO
3404 017376 062716 000002   3$:  ADD      #2,(SP)    ;; ADJUST RETURN PC
3405 017402 000002           RTI           ;; RETURN
3406 017404 122716 000011   4$:  CMPB     #HT,(SP)  ;; BRANCH IF <HT>
3407 017410 001430           BEQ      8$           ;;
3408 017412 122716 000200   CMPB     #CRLF,(SP)  ;; BRANCH IF NOT <CRLF>
3409 017416 001006           BNE      5$           ;;
3410 017420 005726           TST      (SP)+        ;; POP <CR><LF> EQUIV
3411 017422 104401           TYPE           ;; TYPE A CR AND LF
3412 017424 001245           $CRLF
3413 017426 105067 000130   CLRB     $CHARCNT    ;; CLEAR CHARACTER COUNT
3414 017432 000755           BR      2$           ;; GET NEXT CHARACTER
3415 017434 004767 000056   5$:  JSR      PC,$TYPEC  ;; GO TYPE THIS CHARACTER
3416 017440 126726 161544   6$:  CMPB     $FILLC,(SP)+  ;; IS IT TIME FOR FILLER CHARS.?
3417 017444 001350           BNE      2$           ;; IF NO GO GET NEXT CHAR.
3418 017446 016746 161534   MOV      $NULL,-(SP)  ;; GET # OF FILLER CHARS. NEEDED
3419                                     AND THE NULL CHAR.
3420 017452 105366 000001   7$:  DECB     1(SP)      ;; DOES A NULL NEED TO BE TYPED?
3421 017456 002770           BLT      6$           ;; BR IF NO--GO POP THE NULL OFF OF STACK
3422 017460 004767 000032   JSR      PC,$TYPEC  ;; GO TYPE A NULL
3423 017464 105367 000072   DECB     $CHARCNT    ;; DO NOT COUNT AS A COUNT
3424 017470 000770           BR      7$           ;; LOOP
3425
3426      ;HORIZONTAL TAB PROCESSOR
3427
3428 017472 112716 000040   8$:  MOV      #'(SP)    ;; REPLACE TAB WITH SPACE
3429 017476 004767 000014   9$:  JSR      PC,$TYPEC  ;; TYPE A SPACE
3430 017502 132767 000007 000052  BITB     #7,$CHARCNT  ;; BRANCH IF NOT AT
3431 017510 001372           BNE      9$           ;; TAB STOP
3432 017512 005726           TST      (SP)+        ;; POP SPACE OFF STACK
3433 017514 000724           BR      2$           ;; GET NEXT CHARACTER
3434 017516 105777 161460   $TYPEC: TSTB     2$TPS    ;; WAIT UNTIL PRINTER IS READY
3435 017522 100375           BPL      $TYPEC
3436 017524 116677 000002 161452  MOV      2(SP),2$TPB  ;; LOAD CHAR TO BE TYPED INTO DATA REG.

```



```

3437 017532 122766 000015 000002
3438 017540 001003
3439 017542 105067 000014
3440 017546 000406
3441 017550 122766 000012 000002 1S:
3442 017556 001402
3443 017560 105227
3444 017562 000000
3445 017564 000207
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468
3469
3470
3471
3472 017566 017646 000000
3473 017572 116667 000001 000211
3474 017600 112667 000207
3475 017604 062716 000002
3476 017610 000406
3477 017612 112767 000001 000171
3478 017620 112767 000006 000165
3479 017626 112767 000005 000154
3480 017634 010346
3481 017636 010446
3482 017640 010546
3483 017642 116704 000145
3484 017646 005404
3485 017650 062704 000006
3486 017654 110467 000132
3487 017660 116704 000125
3488 017664 016605 000012
3489 017670 005003
3490 017672 006105 1S:
3491 017674 000404
3492 017676 006105 2S:

```

```

CMPB #CR,2(SP) ;; IS CHARACTER A CARRIAGE RETURN?
BNE 1S ;; BRANCH IF NO
CLRB $CHARCNT ;; YES--CLEAR CHARACTER COUNT
BR $TYPEX ;; EXIT
CMPB #LF,2(SP) ;; IS CHARACTER A LINE FEED?
BEQ $TYPEX ;; BRANCH IF YES
INCB (PC)+ ;; COUNT THE CHARACTER
$CHARCNT: .WORD 0 ;; CHARACTER COUNT STORAGE
$TYPEX: RTS PC

.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
$TYPOS: MOV @2(SP),-(SP) ;; PICKUP THE MODE
MOV 1(SP),$OFILL ;; LOAD ZERO FILL SWITCH
MOVB (SP)+,$SOMODE+1 ;; NUMBER OF DIGITS TO TYPE
ADD #2,(SP) ;; ADJUST RETURN ADDRESS
BR $TYPON
$TYPOC: MOVB #1,$OFILL ;; SET THE ZERO FILL SWITCH
MOVB #6,$SOMODE+1 ;; SET FOR SIX(6) DIGITS
$TYPON: MOVB #5,$SOCNT ;; SET THE ITERATION COUNT
MOV R3,-(SP) ;; SAVE R3
MOV R4,-(SP) ;; SAVE R4
MOV R5,-(SP) ;; SAVE R5
MOVB $SOMODE+1,R4 ;; GET THE NUMBER OF DIGITS TO TYPE
NEG R4
ADD #6,R4 ;; SUBTRACT IT FOR MAX. ALLOWED
MOVB R4,$SOMODE ;; SAVE IT FOR USE
MOVB $OFILL,R4 ;; GET THE ZERO FILL SWITCH
MOV 12(SP),R5 ;; PICKUP THE INPUT NUMBER
CLR R3 ;; CLEAR THE OUTPUT WORD
ROL R5 ;; ROTATE MSB INTO "C"
BR 3S ;; GO DO MSB
ROL R5 ;; FORM THIS DIGIT

```

```

3493 017700 006105      ROL      R5
3494 017702 006105      ROL      R5
3495 017704 010503      MOV      R5,R3
3496 017706 006103      3$:    ROL      R3          ;;GET LSB OF THIS DIGIT
3497 017710 105367 000076  DECB    $OMODE        ;;TYPE THIS DIGIT?
3498 017714 100016      BPL      7$           ;;BR IF NO
3499 017716 042703 177770  BIC      #177770,R3    ;;GET RID OF JUNK
3500 017722 001002      BNE      4$           ;;TEST FOR 0
3501 017724 005704      TST      R4           ;;SUPPRESS THIS 0?
3502 017726 001403      BEQ      5$           ;;BR IF YES
3503 017730 005204      4$:    INC      R4           ;;DON'T SUPPRESS ANYMORE 0'S
3504 017732 052703 000060  BIS      #'0,R3        ;;MAKE THIS DIGIT ASCII
3505 017736 052703 000040  5$:    BIS      #' ,R3        ;;MAKE ASCII IF NOT ALREADY
3506 017742 110367 000040  MOVB     R3,8$         ;;SAVE FOR TYPING
3507 017746 104401 020006  TYPE     8$           ;;GO TYPE THIS DIGIT
3508 017752 105367 000032  7$:    DECB    $OCNT        ;;COUNT BY 1
3509 017756 003347      BGT      2$           ;;BR IF MORE TO DO
3510 017760 002402      BLT      6$           ;;BR IF DONE
3511 017762 005204      INC      R4           ;;INSURE LAST DIGIT ISN'T A BLANK
3512 017764 000744      BR       2$           ;;GO DO THE LAST DIGIT
3513 017766 012605      6$:    MOV      (SP)+,R5    ;;RESTORE R5
3514 017770 012604      MOV      (SP)+,R4    ;;RESTORE R4
3515 017772 012603      MOV      (SP)+,R3    ;;RESTORE R3
3516 017774 016666 000002 000004  MOV      2(SP),4(SP)  ;;SET THE STACK FOR RETURNING
3517 020002 012616      MOV      (SP)+,(SP)
3518 020004 000002      RTI
3519 020006      000      8$:    .BYTE    0           ;;RETURN
3520 020007      000      .BYTE    0           ;;STORAGE FOR ASCII DIGIT
3521 020010      000      .BYTE    0           ;;TERMINATOR FOR TYPE ROUTINE
3522 020011      000      .BYTE    0           ;;OCTAL DIGIT COUNTER
3523 020012 000000  $OCNT:  .BYTE    0           ;;ZERO FILL SWITCH
3524      .SBTTL TRAP DECODER  $OMODE:  .WORD    0           ;;NUMBER OF DIGITS TO TYPE
3525
3526      ;;*****
3527      ;;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
3528      ;;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3529      ;;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
3530      ;;*GO TO THAT ROUTINE.
3531
3532 020014 010046 000002  STRAP:  MOV      RO,-(SP)  ;;SAVE RO
3533 020016 016600      MOV      2(SP),RO      ;;GET TRAP ADDRESS
3534 020022 005740      TST      -(RO)         ;;BACKUP BY 2
3535 020024 111000      MOVB     (RO),RO        ;;GET RIGHT BYTE OF TRAP
3536 020026 006300      ASL      RO            ;;POSITION FOR INDEXING
3537 020030 016000 020050  MOV      $TRPAD(RO),RO  ;;INDEX TO TABLE
3538 020034 000200      RTS      RO            ;;GO TO ROUTINE
3539
3540
3541      ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
3542
3543 020036 011646 000004 000002  STRAP2: MOV      (SP)-,(SP)  ;;MOVE THE PC DOWN
3544 020040 016666      MOV      4(SP),2(SP)   ;;MOVE THE PSW DOWN
3545 020046 000002      RTI
3546
3547      .SBTTL TRAP TABLE
3548

```


3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603
3604

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

```

: ROUTINE
:-----
$TRPAD: .WORD $TRAP2
$TYPE   ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
$TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
$TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
$TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
$TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
    
```

.SBTTL POWER DOWN AND UP ROUTINES

```

:POWER DOWN ROUTINE
$PWRDN: MOV      #$SILLUP, @#PWRVEC ;;SET FOR FAST UP
        MOV      #340, @#PWRVEC+2 ;;PRIO:7
        MOV      R0, -(SP)        ;;PUSH R0 ON STACK
        MOV      R1, -(SP)        ;;PUSH R1 ON STACK
        MOV      R2, -(SP)        ;;PUSH R2 ON STACK
        MOV      R3, -(SP)        ;;PUSH R3 ON STACK
        MOV      R4, -(SP)        ;;PUSH R4 ON STACK
        MOV      R5, -(SP)        ;;PUSH R5 ON STACK
        MOV      @SWR, -(SP)      ;;PUSH @SWR ON STACK
        MOV      SP, $SAVR6      ;;SAVE SP
        MOV      #SPWRUP, @#PWRVEC ;;SET UP VECTOR
        HALT
        BR      .-2                ;;HANG UP
    
```

```

:POWER UP ROUTINE
$PWRUP: MOV      #$SILLUP, @#PWRVEC ;;SET FOR FAST DOWN
        MOV      $SAVR6, SP      ;;GET SP
        CLR      $SAVR6         ;;WAIT LOOP FOR THE TTY
        INC      $SAVR6         ;;WAIT FOR THE INC
1$:     BNE      1$              ;;OF WORD
        MOV      (SP)+, @SWR     ;;POP STACK INTO @SWR
        MOV      (SP)+, R5      ;;POP STACK INTO R5
        MOV      (SP)+, R4      ;;POP STACK INTO R4
        MOV      (SP)+, R3      ;;POP STACK INTO R3
        MOV      (SP)+, R2      ;;POP STACK INTO R2
        MOV      (SP)+, R1      ;;POP STACK INTO R1
        MOV      (SP)+, R0      ;;POP STACK INTO R0
        MOV      #SPWRDN, @#PWRVEC ;;SET UP THE POWER DOWN VECTOR
        MOV      #340, @#PWRVEC+2 ;;PRIO:7
        TYPE     104401         ;;REPORT THE POWER FAILURE
        SPWRMG: .WORD $POWER    ;;POWER FAIL MESSAGE POINTER
        RTI
$SILLUP: HALT
        BR      .-2
$SAVR6: 0
$POWER: .ASCIZ <15><12>"POWER"
        .EVEN
    
```

3605				
3606				
3607	020242	005015	046505	050101
3608	020250	020072	000	
3609	020253	015	040412	046114
3610	020260	042440	042530	041522
3611	020266	051511	051117	020123
3612	020274	042524	052123	042105
3613	020302	005015		
3614	020304	020040	047040	052117
3615	020312	035105	047524	052040
3616	020320	051505	020124	040520
3617	020326	051523	047111	020107
3618	020334	043117	043440	040522
3619	020342	052116	020123	047506
3620	020350	020122	044124	020105
3621	020356	040514	052123	052440
3622	020364	042502	005015	
3623	020370	020040	020040	020040
3624	020376	020040	052111	051440
3625	020404	047510	046125	020104
3626	020412	042502	051440	040527
3627	020420	050120	042105	053440
3628	020426	052111	020110	020101
3629	020434	041125	006505	012
3630	020441	040	020040	020040
3631	020446	020040	047440	020106
3632	020454	044510	044107	051105
3633	020462	042440	042514	052103
3634	020470	044522	040503	020114
3635	020476	051120	047511	044522
3636	020504	054524	005015	000
3637	020511	015	041012	051525
3638	020516	050040	051101	052111
3639	020524	020131	047516	020124
3640	020532	042524	052123	042105
3641	020540	047440	020116	030461
3642	020546	030057	020065	051117
3643	020554	030440	027461	030062
3644	020562	046440	041501	044510
3645	020570	042516	006523	000012
3646	020576	047516	051040	051505
3647	020604	047520	051516	020105
3648	020612	047524	051040	043505
3649	020620	040440	042104	042522
3650	020626	051523	051505	047440
3651	020634	020122	047516	042040
3652	020642	053105	041511	020105
3653	020650	051120	051505	047105
3654	020656	000124		
3655	020660	040506	040524	020114
3656	020666	051105	047522	035122
3657	020674	042522	020107	040506
3658	020702	046111	042105	052040
3659	020710	020117	046103	040505
3660	020716	000122		

::*****
:*****

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

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

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

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

.ASCIZ/ OF HIGHER ELECTRICAL PRIORITY/<15><12>

MSG5: .ASCIZ<15><12>*BUS PARITY NOT TESTED ON 11/05 OR 11/20 MACHINES*<15><12>

EM1: .ASCIZ/NO RESPONSE TO REG ADDRESSES OR NO DEVICE PRESENT/

EM2: .ASCIZ/FATAL ERROR:REG FAILED TO CLEAR/

3661	020720	042522	020107	042101	DH2:	.ASCIZ*REG ADD/REG CONTENTS *
3662	020726	027504	042522	020107		
3663	020734	047503	052116	047105		
3664	020742	051524	000040			
3665						.EVEN
3666	020746	001214	001216	000000	DT2:	.WORD \$REG0,\$REG1,0
3667	020754	040506	040524	020114	EM3:	.ASCIZ/FATAL ERROR:CPU DID NOT RECEIVE SSYN/
3668	020762	051105	047522	035122		
3669	020770	050103	020125	044504		
3670	020776	020104	047516	020124		
3671	021004	042522	042503	053111		
3672	021012	020105	051523	047131		
3673	021020	000				
3674	021021	120	020103	040527	DH3:	.ASCIZ/PC WAS/
3675	021026	000123				
3676						.EVEN
3677	021030	001214	000000		DT3:	.WORD \$REG0,0
3678	021034	040506	040524	020114	EM4:	.ASCIZ/FATAL ERROR:REG FAILED TO FLOAT A '1'/'
3679	021042	051105	047522	035122		
3680	021050	042522	020107	040506		
3681	021056	046111	042105	052040		
3682	021064	020117	046106	040517		
3683	021072	020124	020101	030447		
3684	021100	000047				
3685	021102	042522	020107	042101	DH4:	.ASCIZ*REG ADD/DATA IS/DATA SHOULD BE*
3686	021110	027504	040504	040524		
3687	021116	044440	027523	040504		
3688	021124	040524	051440	047510		
3689	021132	046125	020104	042502		
3690	021140	000				
3691		021142				.EVEN
3692	021142	001214	001216	001220	DT4:	.WORD \$REG0,\$REG1,\$REG2,0
3693	021150	000000				
3694	021152	040506	040524	020114	EM5:	.ASCIZ/FATAL ERROR:REG FAILED TO FLOAT A '0'/'
3695	021160	051105	047522	035122		
3696	021166	042522	020107	040506		
3697	021174	046111	042105	052040		
3698	021202	020117	046106	040517		
3699	021210	020124	020101	030047		
3700	021216	000047				
3701	021220	040506	040524	020114	EM6:	.ASCIZ/FATAL ERROR:CONTROL REG HELD WRONG DATA/
3702	021226	051105	047522	035122		
3703	021234	047503	052116	047522		
3704	021242	020114	042522	020107		
3705	021250	042510	042114	053440		
3706	021256	047522	043516	042040		
3707	021264	052101	000101			
3708	021270	040506	040524	020114	EM7:	.ASCIZ/FATAL ERROR:DUAL ADDRESSING ERROR/
3709	021276	051105	047522	035122		
3710	021304	052504	046101	040440		
3711	021312	042104	042522	051523		
3712	021320	047111	020107	051105		
3713	021326	047522	000122			
3714	021332	042522	020107	042101	DH7:	.ASCIZ*REG ADD/REG ADD WERE SIMULATANEOUSLY WRITTEN*
3715	021340	027504	042522	020107		
3716	021346	042101	020104	042527		

3717	021354	042522	051440	046511		
3718	021362	046125	052101	047101		
3719	021370	047505	051525	054514		
3720	021376	053440	044522	052124		
3721	021404	047105	000			
3722		021410				
3723	021410	001214	001216	000000	DT7:	.EVEN
3724	021416	051105	047522	035122	EM8:	.WORD \$REG0,\$REG1,0
3725	021424	051440	052105	044524		.ASCIZ/ERROR: SETTING PB PARITY FAILED TO CAUSE CPU TO TRAP/
3726	021432	043516	050040	020102		
3727	021440	040520	044522	054524		
3728	021446	043040	044501	042514		
3729	021454	020104	047524	041440		
3730	021462	052501	042523	041440		
3731	021470	052520	052040	020117		
3732	021476	051124	050101	000		
3733	021503	105	051122	051117	EM9:	.ASCIZ/ERROR: GO BIT FAILED TO LOAD '1'/'
3734	021510	020072	047507	041040		
3735	021516	052111	043040	044501		
3736	021524	042514	020104	047524		
3737	021532	046040	040517	020104		
3738	021540	030447	000047			
3739	021544	051105	047522	035122	EM10:	.ASCIZ/ERROR: GO BIT FAILED TO LOAD '0'/'
3740	021552	043440	020117	044502		
3741	021560	020124	040506	046111		
3742	021566	042105	052040	020117		
3743	021574	047514	042101	023440		
3744	021602	023460	000			
3745	021605	106	052101	046101	EM11:	.ASCIZ/FATAL ERROR: GO BIT FAILED TO CLEAR/
3746	021612	042440	051122	051117		
3747	021620	020072	047507	041040		
3748	021626	052111	043040	044501		
3749	021634	042514	020104	047524		
3750	021642	041440	042514	051101		
3751	021650	000				
3752	021651	106	052101	046101	EM12:	.ASCIZ/FATAL ERROR: READY BIT FAILED TO SET/
3753	021656	042440	051122	051117		
3754	021664	020072	042522	042101		
3755	021672	020131	044502	020124		
3756	021700	040506	046111	042105		
3757	021706	052040	020117	042523		
3758	021714	000124				
3759	021716	047524	041440	042514	EM14:	.ASCIZ/TO CLEAR BIT 10 OF BECR1/
3760	021724	051101	041040	052111		
3761	021732	030440	020060	043117		
3762	021740	041040	041505	030522		
3763	021746	000				
3764	021747	105	051122	051117	EM15:	.ASCIZ/ERROR: ERROR BITS IN BECR2 SET WHEN SHOULD BE CLEAR/
3765	021754	020072	051105	047522		
3766	021762	020122	044502	051524		
3767	021770	044440	020116	042502		
3768	021776	051103	020062	042523		
3769	022004	020124	044127	047105		
3770	022012	051440	047510	046125		
3771	022020	020104	042502	041440		
3772	022026	042514	051101	000		

3773	022033	103	047117	042524	DH15:	.ASCIZ/CONTENTS OF BECR2/
3774	022040	052116	020123	043117		
3775	022046	041040	041505	031122		
3776	022054	000				
3777	022055	106	052101	046101	EM16:	.ASCIZ/FATAL ERROR: READY BIT FAILED TO CLEAR OR GO FAILED TO SET/
3778	022062	042440	051122	051117		
3779	022070	020072	042522	042101		
3780	022076	020131	044502	020124		
3781	022104	040506	046111	042105		
3782	022112	052040	020117	046103		
3783	022120	040505	020122	051117		
3784	022126	043440	020117	040506		
3785	022134	046111	042105	052040		
3786	022142	020117	042523	000124		
3787	022150	051105	047522	035122	EM17:	.ASCIZ/ERROR: UBE FAILED TO INTERRUPT/
3788	022156	052440	042502	043040		
3789	022164	044501	042514	020104		
3790	022172	047524	044440	052116		
3791	022200	051105	052522	052120		
3792	022206	000				
3793	022207	102	020122	051511	DH17:	.ASCIZ*BR IS / PRIORITY IS*
3794	022214	020040	020057	051120		
3795	022222	047511	044522	054524		
3796	022230	044440	000123			
3797	022234	051105	047522	035122	EM18:	.ASCIZ/ERROR: UBE INTERRUPTED WHEN PSW AT SAME PRIORITY LEVEL/
3798	022242	052440	042502	044440		
3799	022250	052116	051105	052522		
3800	022256	052120	042105	053440		
3801	022264	042510	020116	051520		
3802	022272	020127	052101	051440		
3803	022300	046501	020105	051120		
3804	022306	047511	044522	054524		
3805	022314	046040	053105	046105		
3806	022322	000				
3807	022323	125	042502	041040	DH18:	.ASCIZ/UBE BR WAS/
3808	022330	020122	040527	000123		
3809	022336	051105	047522	035122	EM19:	.ASCIZ/ERROR: UBE FALSELY INTERRUPTED AT A HIGHER LEVEL/
3810	022344	052440	042502	043040		
3811	022352	046101	042523	054514		
3812	022360	044440	052116	051105		
3813	022366	052522	052120	042105		
3814	022374	040440	020124	020101		
3815	022402	044510	044107	051105		
3816	022410	046040	053105	046105		
3817	022416	000				
3818	022417	110	043511	042510	DH19:	.ASCIZ/HIGHER LEVEL WAS/
3819	022424	020122	042514	042526		
3820	022432	020114	040527	000123		
3821	022440	051105	047522	035122	EM20:	.ASCIZ/ERROR: UBE INTERRUPTED TO WRONG VECTOR/
3822	022446	052440	042502	044440		
3823	022454	052116	051105	052522		
3824	022462	052120	042105	052040		
3825	022470	020117	051127	047117		
3826	022476	020107	042526	052103		
3827	022504	051117	000			
3828						

3829	022507	105	051122	051117	EM21:	.ASCIZ/ERROR: SIMULTANEOUS GO FAILED/
3830	022514	020072	044523	052515		
3831	022522	052114	047101	047505		
3832	022530	051525	043440	020117		
3833	022536	040506	046111	042105		
3834	022544	000				
3835	022545	105	051122	051117	EM22:	.ASCIZ/ERROR: NO, NO SACK BIT FALSELY SET/
3836	022552	020072	047516	020054		
3837	022560	047516	051440	041501		
3838	022566	020113	044502	020124		
3839	022574	040506	051514	046105		
3840	022602	020131	042523	000124		
3841	022610	051105	047522	035122	EM23:	.ASCIZ/ERROR: NO INT. Ssyn BIT FALSELY SET/
3842	022616	047040	020117	047111		
3843	022624	027124	051440	054523		
3844	022632	020116	044502	020124		
3845	022640	040506	051514	046105		
3846	022646	020131	042523	000124		
3847	022654	051105	047522	035122	EM24:	.ASCIZ/ERROR: DATI FAILED TO LOAD PROPER DATA/
3848	022662	042040	052101	020111		
3849	022670	040506	046111	042105		
3850	022676	052040	020117	047514		
3851	022704	042101	050040	047522		
3852	022712	042520	020122	040504		
3853	022720	040524	000			
3854	022723	102	041105	020104	DH24:	.ASCIZ*BEED /MEM DATA/MEM ADD/DATA SHOULD BE IN MEM*
3855	022730	046457	046505	042040		
3856	022736	052101	027501	042515		
3857	022744	020115	042101	027504		
3858	022752	040504	040524	051440		
3859	022760	047510	046125	020104		
3860	022766	042502	044440	020116		
3861	022774	042515	000115			
3862						.EVEN
3863	023000	001214	001216	001220	DT24:	.WORD \$REG0,\$REG1,\$REG2,\$REG3,0
3864	023006	001222	000000			
3865	023012	040504	047524	043040	EM25:	.ASCIZ/DATO FAILED TO LOAD PROPER DATA/
3866	023020	044501	042514	020104		
3867	023026	047524	046040	040517		
3868	023034	020104	051120	050117		
3869	023042	051105	042040	052101		
3870	023050	000101				
3871	023052	040504	044524	020120	EM26:	.ASCIZ/DATIP FAILED TO LOAD PROPER DATA/
3872	023060	040506	046111	042105		
3873	023066	052040	020117	047514		
3874	023074	042101	050040	047522		
3875	023102	042520	020122	040504		
3876	023110	040524	000			
3877	023113	104	052101	041117	EM27:	.ASCIZ/DATOB FILED TO LOAD PROPER DATA/
3878	023120	043040	046111	042105		
3879	023126	052040	020117	047514		
3880	023134	042101	050040	047522		
3881	023142	042520	020122	040504		
3882	023150	040524	000			
3883	023153	104	052101	020111	EM28:	.ASCIZ/DATI FAILED TO SET RDY/
3884	023160	040506	046111	042105		

3885	023166	052040	020117	042523	
3886	023174	020124	042122	000131	
3887	023202	040504	047524	043040	EM29: .ASCIZ/DATO FAILED TO SET RDY/
3888	023210	044501	042514	020104	
3889	023216	047524	051440	052105	
3890	023224	051040	054504	000	
3891	023231	104	052101	050111	EM30: .ASCIZ/DATIP FAILED TO SET RDY/
3892	023236	043040	044501	042514	
3893	023244	020104	047524	051440	
3894	023252	052105	051040	054504	
3895	023260	000			
3896	023261	104	052101	041117	EM31: .ASCIZ/DATOB FAILED TO SET RDY/
3897	023266	043040	044501	042514	
3898	023274	020104	047524	051440	
3899	023302	052105	051040	054504	
3900	023310	000			
3901	023311	105	051122	051117	EM32: .ASCIZ/ERROR: INH. DATA SHIFT ON DATIP FAILED/
3902	023316	020072	047111	027110	
3903	023324	042040	052101	020101	
3904	023332	044123	043111	020124	
3905	023340	047117	042040	052101	
3906	023346	050111	043040	044501	
3907	023354	042514	000104		
3908	023360	051105	047522	035122	EM33: .ASCIZ/ERROR: DATOB ON DATIP FAILED/
3909	023366	042040	052101	041117	
3910	023374	047440	020116	040504	
3911	023402	044524	020120	040506	
3912	023410	046111	042105	000	
3913	023415	105	051122	051117	EM34: .ASCIZ/ERROR: UBE DID DATI FROM WRONG LOCATION/
3914	023422	020072	041125	020105	
3915	023430	044504	020104	040504	
3916	023436	044524	043040	047522	
3917	023444	020115	051127	047117	
3918	023452	020107	047514	040503	
3919	023460	044524	047117	000	
3920	023465	115	046505	046040	DH34: .ASCIZ/MEM LOC WANTED/
3921	023472	041517	053440	047101	
3922	023500	042524	000104		
3923	023504	051105	047522	035122	EM35: .ASCIZ/ERROR: BEBA LOWER 16 BITS DID NOT COUNT BY 2/
3924	023512	041040	041105	020101	
3925	023520	047514	042527	020122	
3926	023526	033061	041040	052111	
3927	023534	020123	044504	020104	
3928	023542	047516	020124	047503	
3929	023550	047125	020124	054502	
3930	023556	031040	000		
3931	023561	050	042522	024507	DH35: .ASCIZ*(REG) /DATA SHOULD BE*
3932	023566	027440	040504	040524	
3933	023574	051440	047510	046125	
3934	023602	020104	042502	000	
3935	023607	105	051122	051117	EM36: .ASCIZ/ERROR: BEBA BIT A16,17 DID NOT COUNT=0/
3936	023614	020072	042502	040502	
3937	023622	041040	052111	040440	
3938	023630	033061	030454	020067	
3939	023636	044504	020104	047516	
3940	023644	020124	047503	047125	

3991	023652	036524	000060				
3992	023653	030501	026066	030501	DH36:	.ASCIZ/A16,A17/	
3993	023654	000067					
3994	023664	051105	047522	035122	EM37:	.ASCIZ/ERROR: BEBA DID NOT COUNT BY 1/	
3995	023674	041040	041105	020101			
3996	023702	044504	020104	047516			
3997	023710	020124	047503	047125			
3998	023716	020124	054502	030440			
3999	023724	000					
4000	023725	105	051122	051117	EM38:	.ASCIZ/ERROR: INTERRUPT FAILED TO UPDATE BECC TO CORRECT VALUE/	
4001	023732	020072	047111	042524			
4002	023740	051122	050125	020124			
4003	023746	040506	046111	042105			
4004	023754	052040	020117	050125			
4005	023762	040504	042524	041040			
4006	023770	041505	020103	047524			
4007	023776	041440	051117	042522			
4008	024004	052103	053040	046101			
4009	024012	042525	000				
4010	024015	105	051122	051117	EM39:	.ASCIZ/ERROR: BEBA INCREMENTED WHEN IT WAS INHIBITED/	
4011	024022	020072	042502	040502			
4012	024030	044440	041516	042522			
4013	024036	042515	052116	042105			
4014	024044	053440	042510	020116			
4015	024052	052111	053440	051501			
4016	024060	044440	044116	041111			
4017	024066	052111	042105	000			
4018	024073	105	051122	051117	EM40:	.ASCIZ/ERROR: BECC INCREMENTED WHEN IT WAS INHIBITED/	
4019	024100	020072	042502	041503			
4020	024106	044440	041516	042522			
4021	024114	042515	052116	042105			
4022	024122	053440	042510	020116			
4023	024130	052111	053440	051501			
4024	024136	044440	044116	041111			
4025	024144	052111	042105	000			
4026	024151	105	051122	051117	EM41:	.ASCIZ/ERROR: UBE FAILED TO INT. ON DONE/	
4027	024156	020072	041125	020105			
4028	024164	040506	046111	042105			
4029	024172	052040	020117	047111			
4030	024200	027124	047440	020116			
4031	024206	047504	042516	000			
4032	024213	105	051122	051117	EM42:	.ASCIZ/ERROR: CCOVF NOT CLEARED BY GO/	
4033	024220	020072	041503	053117			
4034	024226	020106	047516	020124			
4035	024234	046103	040505	042522			
4036	024242	020104	054502	043440			
4037	024250	000117					
4038	024252	051105	047522	035122	EM43:	.ASCIZ/ERROR: UBE DID NOT DO DATO TO PROPER # OF LOCS. (4)/	
4039	024260	052440	042502	042040			
4040	024266	042111	047040	052117			
4041	024274	042040	020117	040504			
4042	024302	047524	052040	020117			
4043	024310	051120	050117	051105			
4044	024316	021440	047440	020106			
4045	024324	047514	051503	020056			
4046	024332	032050	000051				

3997	024336	042101	020104	051106	DH43:	.ASCIZ*ADD FROM/TO WERE WRITTEN*
3998	024344	046517	052057	020117		
3999	024352	042527	042522	053440		
4000	024360	044522	052124	047105		
4001	024366	000				
4002	024367	105	051122	051117	EM44:	.ASCIZ/ERROR: INT. ON DONE BIT NOT CLEARED/
4003	024374	020072	047111	027124		
4004	024402	047440	020116	047504		
4005	024410	042516	041040	052111		
4006	024416	047040	052117	041440		
4007	024424	042514	051101	042105		
4008	024432	000				
4009	024433	105	051122	051117	EM45:	.ASCIZ/ERROR: CCOVF NOT SET/
4010	024440	020072	041503	053117		
4011	024446	020106	047516	020124		
4012	024454	042523	000124			
4013	024460	051105	047522	035122	EM46:	.ASCIZ/ERROR: DATO FROM BECC NOT DONE PROPERLY/
4014	024466	042040	052101	020117		
4015	024474	051106	046517	041040		
4016	024502	041505	020103	047516		
4017	024510	020124	047504	042516		
4018	024516	050040	047522	042520		
4019	024524	046122	000131			
4020	024530	042101	020104	020040	DH46:	.ASCIZ*ADD /DATA /DATA SHOULD BE*
4021	024536	042057	052101	020101		
4022	024544	020040	042057	052101		
4023	024552	020101	044123	052517		
4024	024560	042114	041040	000105		
4025	024566	051105	047522	035122	EM47:	.ASCIZ/ERROR: UBE DID NOT DO 2 DATO FOR EACH REQUEST/
4026	024574	052440	042502	042040		
4027	024602	042111	047040	052117		
4028	024610	042040	020117	020062		
4029	024616	040504	047524	043040		
4030	024624	051117	042440	041501		
4031	024632	020110	042522	052521		
4032	024640	051505	000124			
4033	024644	051105	047522	035122	EM49:	.ASCIZ/ERROR: UBE DID NOT DO 2 DATIP FOR EACH REQUEST/
4034	024652	052440	042502	042040		
4035	024660	042111	047040	052117		
4036	024666	042040	020117	020062		
4037	024674	040504	044524	020120		
4038	024702	047506	020122	040505		
4039	024710	044103	051040	050505		
4040	024716	042525	052123	000		
4041	024723	105	051122	051117	EM51:	.ASCIZ/ERROR: NPR DATO NOT DONE/
4042	024730	020072	050116	020122		
4043	024736	040504	047524	047040		
4044	024744	052117	042040	047117		
4045	024752	000105				
4046	024754	051105	047522	035122	EM52:	.ASCIZ/ERROR: NPR DID NOT SET RDY/
4047	024762	047040	051120	042040		
4048	024770	042111	047040	052117		
4049	024776	051440	052105	051040		
4050	025004	054504	000			
4051	025007	105	051122	051117	EM53:	.ASCIZ/ERROR: UBE DID NOT INT. WHEN NPR FINISHED/
4052	025014	020072	041125	020105		

4053	025022	044504	020104	047516
4054	025030	020124	047111	027124
4055	025036	053440	042510	020116
4056	025044	050116	020122	044506
4057	025052	044516	044123	042105
4059	025060	000		
4059	025061	105	051122	051117
4060	025066	020072	053524	020117
4061	025074	047514	027103	053440
4062	025102	044522	052124	047105
4063	025110	053440	042510	020116
4064	025116	047117	020105	050116
4065	025124	020122	047101	020104
4066	025132	047111	027124	047440
4067	025140	020116	047504	042516
4068	025146	042440	040516	046102
4069	025154	042105	000	
4070	025157	015	052012	051505
4071	025164	044524	043516	052440
4072	025172	042502	053440	046111
4073	025200	020114	047516	020124
4074	025206	047111	042524	051122
4075	025214	050125	020124	052504
4076	025222	044522	043516	047040
4077	025230	051120	005015	
4078	025234	043111	042040	042517
4079	025242	026123	041440	052520
4080	025250	053440	046111	020114
4081	025256	047507	042040	053517
4082	025264	006516	042412	052116
4083	025272	051105	047111	020107
4084	025300	042524	052123	005015
4085	025306	000		
4086	025307	015	042412	044530
4087	025314	044524	043516	052040
4088	025322	051505	006524	000012
4089	025330	051105	047522	035122
4090	025336	052040	051505	020124
4091	025344	043117	053440	047522
4092	025352	043516	040440	020040
4093	025360	044514	042516	020123
4094	025366	051105	047522	020122
4095	025374	044502	020124	040506
4096	025402	046111	042105	000
4097	025407	102	041505	031122
4098	025414	041040	052111	034440
4099	025422	043040	046101	042523
4100	025430	054514	051440	052105
4101	025436	000		
4102	025437	124	020117	047111
4103	025444	042524	051122	050125
4104	025452	020124	050103	000125
4105	025460	051105	047522	035122
4106	025466	052040	051505	020124
4107	025474	043117	047040	051523
4108	025502	047131	042440	051122

MSG4: .ASCIZ/ERROR: TWO LOC. WRITTEN WHEN ONE NPR AND INT. ON DONE ENABLED/

MSG3: .ASCII<15><12>/TESTING UBE WILL NOT INTERRUPT DURING NPR/<15><12>

.ASCIZ*IF DOES, CPU WILL GO DOWN*<15><12>*ENTERING TEST*<15><12>

MSG4: .ASCIZ<15><12>/EXITING TEST/<15><12>

MSG6: .ASCIZ/ERROR: TEST OF WRONG A LINES ERROR BIT FAILED/

MSG7: .ASCIZ/BECR2 BIT 9 FALSELY SET/

MSG8: .ASCIZ/TO INTERRUPT CPU/

MSG9: .ASCIZ/ERROR: TEST OF NSSYN ERROR BIT FAILED/

4109	025510	051117	041040	052111	
4110	025516	043040	044501	042514	
4111	025524	000104			
4112	025526	047524	051440	052105	EM60: .ASCIZ/TO SET BIT 8 OF BECR2/
4113	025534	041040	052111	034040	
4114	025542	047440	020106	042502	
4115	025550	051103	000062		
4116	025554	047524	051440	052105	EM61: .ASCIZ/TO SET BIT 15 OF BECR1/
4117	025562	041040	052111	030440	
4118	025570	020065	043117	041040	
4119	025576	041505	030522	000	
4120	025603	124	020117	046103	EM62: .ASCIZ/TO CLEAR BIT 8 OF BECR2/
4121	025610	040505	020122	044502	
4122	025616	020124	020070	043117	
4123	025624	041040	041505	031122	
4124	025632	000			
4125	025633	106	046101	042523	EM63: .ASCIZ/FALSELY INTERRUPTED CPU/
4126	025640	054514	044440	052116	
4127	025646	051105	052522	052120	
4128	025654	042105	041440	052520	
4129	025662	000			
4130	025663	105	051122	051117	EM64: .ASCIZ/ERROR: TEST OF WRONG GRANT OR NOT ONE GRANT ERROR BITS FAILED/
4131	025670	020072	042524	052123	
4132	025676	047440	020106	051127	
4133	025704	047117	020107	051107	
4134	025712	047101	020124	051117	
4135	025720	047040	052117	047440	
4136	025726	042516	043440	040522	
4137	025734	052116	042440	051122	
4138	025742	051117	041040	052111	
4139	025750	020123	040506	046111	
4140	025756	042105	000		
4141	025761	116	020117	051107	EM65: .ASCIZ/NO GRANT OR NOT ONE GRANT ERROR BIT FALSELY SET/
4142	025766	047101	020124	051117	
4143	025774	047040	052117	047440	
4144	026002	042516	043440	040522	
4145	026010	052116	042440	051122	
4146	026016	051117	041040	052111	
4147	026024	043040	046101	042523	
4148	026032	054514	051440	052105	
4149	026040	000			
4150	026041	040	044527	044124	DM65: .ASCIZ/ WITH BECR1 = /
4151	026046	041040	041505	030522	
4152	026054	036440	000040		
4153	026060	044527	044124	044440	EM66: .ASCIZ/WITH INT. ON DONE = 1/
4154	026066	052116	020056	047117	
4155	026074	042040	047117	020105	
4156	026102	020075	000061		
4157	026106	051127	047117	020107	EM67: .ASCIZ/WRONG GRANT ERROR BIT FALSELY SET/
4158	026114	051107	047101	020124	
4159	026122	051105	047522	020122	
4160	026130	044502	020124	040506	
4161	026136	051514	046105	020131	
4162	026144	042523	000124		
4163	026150	047524	041440	042514	EM69: .ASCIZ/TO CLEAR BIT 11 OF BECR1/
4164	026156	051101	041040	052111	

4165	026164	030440	020061	043117	
4166	026172	041040	041505	030522	
4167	026200	000			
4169	026201	105	051122	051117	EM70: .ASCIZ/ERROR: TEST OF TIME DELAY AND BUS LATENCY FAILED/
4169	026206	020072	042524	052123	
4170	026214	047440	020106	044524	
4171	026222	042515	042040	046105	
4172	026230	054501	040440	042116	
4173	026236	041040	051525	046040	
4174	026244	052101	047105	054503	
4175	026252	043040	044501	042514	
4176	026260	000104			
4177	026262	047524	051440	052105	EM71: .ASCIZ/TO SET BIT 6 OF BECR2/
4178	026270	041040	052111	033040	
4179	026276	047440	020106	042502	
4180	026304	051103	000062		
4181	026310	047524	041440	042514	EM72: .ASCIZ/TO CLEAR BIT 6 OF BECR2/
4182	026316	051101	041040	052111	
4183	026324	033040	047440	020106	
4184	026332	042502	051103	000062	
4185	026340	051105	047522	035122	EM73: .ASCIZ/ERROR: TEST OF POWER DOWN BIT FAILED/
4186	026346	052040	051505	020124	
4187	026354	043117	050040	053517	
4189	026362	051105	042040	053517	
4189	026370	020116	044502	020124	
4190	026376	040506	046111	042105	
4191	026404	000			
4192	026405	124	020117	047520	EM74: .ASCIZ/TO POWER DOWN CPU/
4193	026412	042527	020122	047504	
4194	026420	047127	041440	052520	
4195	026426	000			
4196	026427	124	020117	047520	EM75: .ASCIZ/TO POWER UP CPU/
4197	026434	042527	020122	050125	
4198	026442	041440	052520	000	
4199	026447	124	020117	042522	EM76: .ASCIZ/TO RE. POWER DOWN CPU/
4200	026454	050040	053517	051105	
4201	026462	042040	053517	020116	
4202	026470	050103	000125		
4203	026474	047524	051440	052105	EM77: .ASCIZ/TO SET BIT 4 OF BECR2/
4204	026502	041040	052111	032040	
4205	026510	047440	020106	042502	
4206	026516	051103	000062		
4207	026522	051105	047522	035122	EM78: .ASCIZ/ERROR: DCLO FAILED TO CLEAR REG/
4208	026530	042040	046103	020117	
4209	026536	040506	046111	042105	
4210	026544	052040	020117	046103	
4211	026552	040505	020122	042522	
4212	026560	000107			
4213	026562	051105	047522	035122	EM80: .ASCIZ/ERROR: DYNAMIC TEST OF UBE FAILED/
4214	026570	042040	047131	046501	
4215	026576	041511	052040	051505	
4216	026604	020124	043117	052440	
4217	026612	042502	043040	044501	
4218	026620	042514	000104		
4219	026624	047524	046040	040517	EM81: .ASCIZ/TO LOAD PROPER DATA/
4220	026632	020104	051120	050117	

4221	026640	051105	042040	052101	
4222	026646	000101			
4223	026650	051105	047522	035122	EM82: .ASCIZ/ERROR: TEST OF PASSING GRANTS FAILED/
4224	026656	052040	051505	020124	
4225	026664	043117	050040	051501	
4226	026672	044523	043516	043440	
4227	026700	040522	052116	020123	
4228	026706	040506	046111	042105	
4229	026714	000			
4230	026715	105	051122	051117	EM83: .ASCIZ/ERROR:FALSE INTERRUPT WHEN DO RELEASE BUS IMMED./
4231	026722	043072	046101	042523	
4232	026730	044440	052116	051105	
4233	026736	052522	052120	053440	
4234	026744	042510	020116	047504	
4235	026752	051040	046105	040505	
4236	026760	042523	041040	051525	
4237	026766	044440	046515	042105	
4238	026774	000056			
4239	026776	051105	047522	035122	EM84: .ASCIZ/ERROR:TEST OF MULTIPLE INTERRUPTS FAILED TO SET RDY/
4240	027004	042524	052123	047440	
4241	027012	020106	052515	052114	
4242	027020	050111	042514	044440	
4243	027026	052116	051105	052522	
4244	027034	052120	020123	040506	
4245	027042	046111	042105	052040	
4246	027050	020117	042523	020124	
4247	027056	042122	000131		
4248	027062	041125	020105	044527	MSG7: .ASCIZ/UBE WITH VECTOR: /
4249	027070	044124	053040	041505	
4250	027076	047524	035122	000040	
4251	027104	040440	042116	041040	MSG8: .ASCIZ/ AND BR AT: /
4252	027112	020122	052101	020072	
4253	027120	000			
4254	027121	040	040506	051514	MSG9: .ASCIZ/ FALSELY INTERRUPTED WHEN/<15><12>
4255	027126	046105	020131	047111	
4256	027134	042524	051122	050125	
4257	027142	042524	020104	044127	
4258	027150	047105	005015	000	
4259	027155	040	044123	052517	MSG10: .ASCIZ/ SHOULD HAVE INTERRUPTED/<15><12>
4260	027162	042114	044040	053101	
4261	027170	020105	047111	042524	
4262	027176	051122	050125	042524	
4263	027204	006504	000012		
4264	027210	005015	040520	051523	MSG11: .ASCIZ<15><12>/PASSING OF GRANTS NOT TESTED WITH ONE EXERCISOR/<15><12>
4265	027216	047111	020107	043117	
4266	027224	043440	040522	052116	
4267	027232	020123	047516	020124	
4268	027240	042524	052123	042105	
4269	027246	053440	052111	020110	
4270	027254	047117	020105	054105	
4271	027262	051105	044503	047523	
4272	027270	006522	000012		
4273	027274	005015	043111	046440	MSG12: .ASCII<15><12>/IF MORE THAN ONE UBE PRESENT JUMPER W1/<15><12>
4274	027302	051117	020105	044124	
4275	027310	047101	047440	042516	
4276	027316	052440	042502	050040	

4277	027324	042522	042523	052116
4278	027332	045040	046525	042520
4279	027340	020122	030527	005015
4280	027346	044123	052517	042114
4281	027354	041040	020105	047111
4282	027362	042523	052122	042105
4283	027370	044440	020116	046101
4284	027376	020114	041125	020105
4285	027404	054105	042503	052120
4286	027412	046040	051501	006524
4287	027420	000012		
4288	027422	005015	042524	052123
4289	027430	047111	020107	041125
4290	027436	020105	044527	044124
4291	027444	041040	042105	020102
4292	027452	042101	051104	051505
4293	027460	035123	000040	
4294	027464	005015	020040	047040
4295	027472	052117	035105	044504
4296	027500	051123	043505	051101
4297	027506	020104	044502	020124
4298	027514	031461	036440	020061
4299	027522	043117	041040	041505
4300	027530	031122	005015	000
4301	027535	015	006412	050012
4302	027542	020103	043117	042440
4303	027550	051122	051117	046440
4304	027556	051505	040523	042507
4305	027564	053440	051501	020072
4306	027572	000		
4307	027573	015	006412	020012
4308	027600	020040	020040	047125
4309	027606	041111	051525	042440
4310	027614	042530	041522	051511
4311	027622	051117	046440	042117
4312	027630	046125	020105	044504
4313	027636	043501	047516	052123
4314	027644	041511	026455	040515
4315	027652	047111	042504	026503
4316	027660	030461	042055	045532
4317	027666	041125	040455	005015
4318	027674	005015	000	
4319				
4320		027700		
4321				
4322				
4323				
4324	027700	000011		
4325		000001		

.ASCIZ/SHOULD BE INSERTED IN ALL UBE EXCEPT LAST/<15><12>

MSG13: .ASCIZ<15><12>/TESTING UBE WITH BEDB ADDRESS: /

MSG14: .ASCIZ<15><12>/ NOTE:DISREGARD BIT 13 =1 OF BECR2/<15><12>

MSG15: .ASCIZ<15><12><15><12>/PC OF ERROR MESSAGE WAS: /

MSG16: .ASCIZ<15><12><15><12>/ UNIBUS EXERCISOR MODULE DIAGNOSTIC--MAINDEC-11-DZKUB

```

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

```


PWRVEC= 000024	604#	1186*	1187*	3566*	3567*	3576*	3582*	3594*	3595*	2403	2455	2519	2570
RCATCH 016166	1579	1593	1595	1693	1946	1999	2031	2128	2315				
RDYS 006542	2588	2723	2747	2799	3101#								
RESVEC= 000C10	1753	1770	1787	1806	1820#								
RSTART 001100	599#												
RTR 006016	623#												
RVEC 016260	1729	1732#											
RO =%000000	627	1664	1676	1699	1702	3135#							
	520#	1220*	1222	1224*	1225	1237	1255*	1256*	1257	1292*	1293	1294	1318*
	1319*	1320	1322	1324	1325	1327	1334	1338*	1374*	1375*	1376	1382	1393*
	1394	1404	1418*	1420*	1421	1423	1425	1433	1447*	1449*	1450	1452	1454
	1462	1481*	1483*	1487	1491	1508	1510	1512*	1514	1532*	1565*	1566*	1571*
	1572*	1573	1622*	1623	1624	1627*	1628*	1629*	1630	1724*	1725*	1726	1905*
	1909	1911*	1912*	1913	1937	1941*	1942*	1943	1965*	1966*	1967	1968	1971*
	1974	1975*	1982	1986	2000*	2013*	2019*	2020	2022	2025	2075*	2076*	2077
	2085*	2086*	2087	2092*	2093	2095	2097	2099	2111	2112	2113	2119	2140*
	2141*	2142	2149*	2151	2154	2158	2159	2160	2180*	2181*	2182	2191*	2192
	2194	2196	2198	2201	2202	2203	2209	2231*	2232*	2233	2241*	2242	2244
	2246	2248	2251	2252	2253	2259	2537*	2538*	2539	2618*	2620*	2621	2631*
	2633*	2634	2765*	2766*	2767	2773*	2775*	2776	2783*	2784	2786	2797	2793
	2794	2846*	2847*	2849*	2850*	2854*	2855*	2859*	2860*	2861*	2875*	2878*	2881*
	2884*	2892	2893	2899*	2900*	2944*	2945	2950*	2951*	2952*	2953	2955*	2957*
	2959*	2969	2971	2973*	2976*	2980	3034*	3036*	3037*	3039	3074*	3077	3272
	3273*	3274*	3281*	3282*	3283*	3284*	3285*	3286	3291	3296*	3298*	3302	3304
	3323	3333*	3337	3353	3354	3367*	3398	3399*	3400	3403*	3532	3533*	3534
	3535*	3536*	3537*	3538*	3568	3593*							
R1 =%000001	521#	1219*	1221	1223*	1291*	1293*	1376*	1377*	1378	1394*	1396*	1397	1419*
	1420	1421	1424	1429	1431*	1448*	1449	1450	1453	1458	1460*	1482*	1485
	1487	1489	1492	1498	1500	1502*	1505	1974*	2000	2150*	2151	2153*	2161
	2619*	2623*	2624	2632*	2636*	2637	2862*	2887*	2888	2901*	2902	2904	2905
	2956*	2958*	2960*	2965*	2966*	2967	2974*	2977*	2982	2983	2987	3035*	3036
	3038	3324	3337*	3338	3342	3366*	3569	3592*					
R2 =%000002	522#	1368*	1375	1379	1383	1392*	1393	1398	1400*	1484*	1485	1503*	2763*
	2789*	2790	2902*	2903*	2931*	2932*	2939*	2941	3325	3336*	3340*	3343	3350*
	3351*	3352	3357*	3365*	3570	3591*							
R3 =%000003	523#	1369*	1392	1401*	2905*	2906*	2907*	2908*	2915	2919	2925*	2926*	2927
	2930*	2935*	2936*	2937	3326	3334*	3335*	3349*	3352*	3361*	3362*	3364*	3480
	3489*	3495*	3496*	3499*	3504*	3505*	3506	3515*	3571	3590*			
R4 =%000004	524#	1370*	1374	1381	1385	1402*	1820*	1823*	1824	1855	2287	2584	3114*
	3121*	3136*	3137*	3138	3141*	3142*	3144	3481	3483*	3484*	3485*	3486	3487*
	3501	3503*	3511*	3514*	3572	3589*							
R5 =%000005	525#	1373*	1378*	1379	1395*	1397*	1398	1750*	1768*	1785*	1801*	1827	3101
	3102*	3103	3104	3105*	3106*	3115*	3116*	3119	3127*	3128	3129	3130*	3135*
	3137	3140*	3141	3143	3327	3329*	3331*	3338*	3342*	3357	3363*	3482	3488*
	3490*	3492*	3493*	3494*	3495	3513*	3573	3588*					
R6 =%000006	526#	625	1174*	1175*	1176								
R7 =%000007	527#												
SP =%000006	528#	1178*	1196*	1204*	1208	1232*	1285*	1312*	1347	1366*	1416*	1445*	1478*
	1524*	1561*	1618*	1621	1623*	1668	1675	1687*	1698	1716*	1744*	1826	1847*
	1875*	1901*	1957*	2010*	2018	2040*	2072*	2137*	2177*	2228*	2277*	2325*	2356*
	2415*	2468*	2531*	2576*	2605*	2606*	2607*	2616	2643	2651	2652	2672*	2673*
	2680	2712	2724	2725	2737*	2782	2844*	2886	2917*	2921*	2964	2979	2995*
	2999*	3003*	3006*	3071*	3101*	3106	3155*	3180*	3183	3185	3186	3210	3211
	3215*	3240	3253*	3256*	3272*	3277*	3298	3302*	3323*	3324*	3325*	3326*	3327*
	3328*	3329	3332*	3345	3347*	3349	3359	3361	3363	3364	3365	3366	3367
	3369*	3370*	3398*	3399	3400*	3402	3403	3404*	3406	3408	3410	3416	3418*

SPTTR	=	002514	3420*	3428*	3432	3436	3437	3441	3472*	3473	3474	3475*	3480*	3481*	3482*
STACK	=	001100	3511*	3513*	3514*	3515*	3516*	3517*	3532*	3533	3543*	3544*	3568*	3569*	3570*
START	=	002632	1120*	1121*	1122*	1123*	1124*	1125*	1479	1524	1561	1518	1687	1716	1744
TKL	=	003100	1126*	1127*	1128*	1129*	1130*	1131*	1524	2137	2177	2228	2277	2325	2356
TRAP	=	004022	1132*	1133*	1134*	1135*	1136*	1137*	2072	2137	2177	2228	2277	2325	2356
SW	=	001172	1138*	1139*	1140*	1141*	1142*	1143*	2072	2137	2177	2228	2277	2325	2356
SWREG	=	000176	1144*	1145*	1146*	1147*	1148*	1149*	2072	2137	2177	2228	2277	2325	2356
SW0	=	000001	1150*	1151*	1152*	1153*	1154*	1155*	2072	2137	2177	2228	2277	2325	2356
SW01	=	000002	1156*	1157*	1158*	1159*	1160*	1161*	2072	2137	2177	2228	2277	2325	2356
SW02	=	000004	1162*	1163*	1164*	1165*	1166*	1167*	2072	2137	2177	2228	2277	2325	2356
SW03	=	000010	1168*	1169*	1170*	1171*	1172*	1173*	2072	2137	2177	2228	2277	2325	2356
SW04	=	000020	1174*	1175*	1176*	1177*	1178*	1179*	2072	2137	2177	2228	2277	2325	2356
SW05	=	000040	1180*	1181*	1182*	1183*	1184*	1185*	2072	2137	2177	2228	2277	2325	2356
SW06	=	000100	1186*	1187*	1188*	1189*	1190*	1191*	2072	2137	2177	2228	2277	2325	2356
SW07	=	000200	1192*	1193*	1194*	1195*	1196*	1197*	2072	2137	2177	2228	2277	2325	2356
SW08	=	000400	1198*	1199*	1200*	1201*	1202*	1203*	2072	2137	2177	2228	2277	2325	2356
SW09	=	001000	1204*	1205*	1206*	1207*	1208*	1209*	2072	2137	2177	2228	2277	2325	2356
SW1	=	000002	1210*	1211*	1212*	1213*	1214*	1215*	2072	2137	2177	2228	2277	2325	2356
SW10	=	002000	1216*	1217*	1218*	1219*	1220*	1221*	2072	2137	2177	2228	2277	2325	2356
SW11	=	004000	1222*	1223*	1224*	1225*	1226*	1227*	2072	2137	2177	2228	2277	2325	2356
SW12	=	010000	1228*	1229*	1230*	1231*	1232*	1233*	2072	2137	2177	2228	2277	2325	2356
SW13	=	020000	1234*	1235*	1236*	1237*	1238*	1239*	2072	2137	2177	2228	2277	2325	2356
SW14	=	040000	1240*	1241*	1242*	1243*	1244*	1245*	2072	2137	2177	2228	2277	2325	2356
SW15	=	100000	1246*	1247*	1248*	1249*	1250*	1251*	2072	2137	2177	2228	2277	2325	2356
SW16	=	000004	1252*	1253*	1254*	1255*	1256*	1257*	2072	2137	2177	2228	2277	2325	2356
SW17	=	000010	1258*	1259*	1260*	1261*	1262*	1263*	2072	2137	2177	2228	2277	2325	2356
SW18	=	000020	1264*	1265*	1266*	1267*	1268*	1269*	2072	2137	2177	2228	2277	2325	2356
SW19	=	000100	1270*	1271*	1272*	1273*	1274*	1275*	2072	2137	2177	2228	2277	2325	2356
SW20	=	000400	1276*	1277*	1278*	1279*	1280*	1281*	2072	2137	2177	2228	2277	2325	2356
SW21	=	001000	1282*	1283*	1284*	1285*	1286*	1287*	2072	2137	2177	2228	2277	2325	2356
TRITVE	=	000014	1288*	1289*	1290*	1291*	1292*	1293*	2072	2137	2177	2228	2277	2325	2356
TERRPC	=	016334	1294*	1295*	1296*	1297*	1298*	1299*	2072	2137	2177	2228	2277	2325	2356
TKVEC	=	000060	1300*	1301*	1302*	1303*	1304*	1305*	2072	2137	2177	2228	2277	2325	2356
TMAP	=	002512	1306*	1307*	1308*	1309*	1310*	1311*	2072	2137	2177	2228	2277	2325	2356
TPVEC	=	000064	1312*	1313*	1314*	1315*	1316*	1317*	2072	2137	2177	2228	2277	2325	2356
TRAPVE	=	000034	1318*	1319*	1320*	1321*	1322*	1323*	2072	2137	2177	2228	2277	2325	2356
TRTVEC	=	000014	1324*	1325*	1326*	1327*	1328*	1329*	2072	2137	2177	2228	2277	2325	2356
TSTA	=	006630	1330*	1331*	1332*	1333*	1334*	1335*	2072	2137	2177	2228	2277	2325	2356
TSTB	=	014076	1336*	1337*	1338*	1339*	1340*	1341*	2072	2137	2177	2228	2277	2325	2356
TST1	=	003622	1342*	1343*	1344*	1345*	1346*	1347*	2072	2137	2177	2228	2277	2325	2356

T04L04	004376	1448#	1463		
T05L01	004564	1486#	1489	1498#	
T05L02	004616	1499	1507#		
T05L03	004514	1485#	1504	1506	
T05L04	004500	1482#	1513	1515	
T05L05	004644	1511	1514#		
T05L06	004612	1501	1505#		
T05L07	004554	1490	1496#		
T05L08	004532	1491#	1497		
T06L01	004774	1543	1545	1548#	
T07L03	005100	1572#	1576		
T07L04	005144	1574	1583#		
T07L05	005202	1578	1595#		
T07L06	005172	1585	1593#		
T07L07	005076	1571#	1589		
T07L08	005164	1569	1591#		
T08L01	005256	1628#	1631		
T08L02	005524	1654	1668#		
T08L03	005462	1658#	1673		
T08L04	005614	1680	1686#		
T08L05	005622	1686	1687#		
T08L06	005710	1667	1703#		
T08L07	005704	1670	1702#		
T08L08	005234	1623#	1625		
T08L09	005510	1657	1664#		
T09L01	014154	2740	2747#		
T10L01	006160	1755	1763#		
T10L02	006300	1772	1781#		
T10L03	006424	1791	1800#		
T10L04	006570	1822	1828#		
T10L05	006544	1821#	1825		
T10L06	006364	1789	1792#		
T11L01	006716	1856	1860#		
T11L02	006736	1861	1864#		
T12L01	007066	1884	1887#		
T13L01	007460	1964	1971#		
T13L02	007442	1966#	1969		
T13L03	007464	1970	1973#		
T13L04	007622	1981	1987	1995#	
T13L05	007644	1994	1999#		
T14L01	007336	1910	1936#		
T14L02	007152	1906#	1914		
T14L03	007360	1921	1941#		
T14L04	007402	1930	1935	1940	1946#
T15L01	007716	2014	2018#		
T15L02	007774	2021	2031#		
T15L03	007702	2015#	2023		
T16L01	010062	2049#	2050		
T16L02	010120	2052	2058#		
T16L03	010126	2054	2057	2060#	
T17L01	010164	2076#	2078		
T17L02	010260	2082	2092#		
T17L03	010240	2086#	2088		
T17L04	010356	2094	2111#		
T17L05	010264	2093#	2096		
T17L06	010276	2097#	2100		

T17L07	010422	2102	2123#					
T17L08	010432	2104	2126#					
T17L09	010440	2091	2107	2110	2117	2122	2125	2128#
T17L10	010360	2098	2112#					
T17L11	010400	2114	2118#					
T18L01	010466	2141#	2143					
T18L02	010520	2147#	2148					
T18L03	010556	2152	2158#					
T18L04	010540	2151#	2155					
T19L01	010626	2181#	2183					
T19L02	010736	2197	2202#					
T19L03	010706	2192#	2195					
T19L04	010720	2196#	2199					
T19L05	010776	2200	2207	2212#				
T19L07	010756	2204	2208#					
T19L09	010734	2193	2201#					
T20L01	011034	2232#	2234					
T20L02	011142	2243	2251#					
T20L03	011110	2242#	2245					
T20L04	011124	2246#	2249					
T20L05	011204	2250	2257	2262#				
T20L06	011164	2254	2258#					
T20L08	011144	2247	2252#					
T21L01	011410	2288	2307#					
T21L02	011436	2290	2313#					
T21L03	011372	2293	2302#					
T21L04	011444	2301	2303	2306	2312	2315#		
T22L01	011544	2336	2339#					
T23L01	011732	2359	2381#					
T23L02	011666	2366	2370#					
T23L03	011706	2371	2375#					
T23L04	012060	2399	2403#					
T23L05	011756	2383	2387#					
T23L06	011776	2388	2392#					
T23L07	012040	2395	2398#					
T24L01	012202	2421	2432#					
T24L02	012172	2425	2429#					
T24L03	012340	2436	2445	2455#				
T24L04	012306	2443	2448#					
T24L05	012240	2440#	2451					
T24L06	012222	2431	2437#					
T25L01	012456	2471	2484#					
T25L02	012506	2478	2491#					
T25L03	012610	2480	2510#					
T25L04	012630	2482	2515#					
T25L05	012376	2473#	2483					
T25L06	012524	2491	2494#					
T25L07	012566	2495	2504#					
T25L08	012646	2490	2497	2503	2509	2514	2519#	
T26L01	013006	2535	2557#					
T26L02	012716	2538#	2540					
T26L03	012746	2542	2546#					
T26L04	012766	2547	2551#					
T26L05	013032	2556	2559	2563#				
T27L01	013266	2608	2616#					
T27L02	013406	2615	2629	2642	2644	2646#		

T27L03	013332	2617	2630#		
T27L04	013302	2620#	2622	2625	
T27L05	013374	2630	2643#		
T27L06	013344	2633#	2635	2638	
T27L07	013426	2647	2651#		
T28L01	013600	2677	2680#		
T28L02	013714	2687	2690	2693	2698#
T28L03	013732	2697	2702#		
T28L04	014062	2701	2717	2723#	
T28L05	013612	2681	2683#		
T28L06	013754	2705#	2706		
T28L08	014010	2709	2712#		
T28L09	014022	2713	2715#		
T28L10	013466	2663	2665#		
T29L01	014206	2766#	2768		
T29L02	014300	2769	2782#		
T29L03	014336	2785	2793#		
T29L04	014202	2765#	2791		
T29L05	014306	2784#	2788		
T29L06	014252	2774#	2777		
T29L07	014364	2781	2792	2799#	
T30L01	014460	2835	2844#		
T30L02	014672	2845	2875#		
T30L03	014706	2848	2878#		
T30L04	014722	2853	2881#		
T30L05	014736	2858	2884#		
T30L06	014750	2877	2880	2883	2886#
T30L07	014660	2867	2870	2872#	
T30L08	015450	2951	2959	2979#	
T30L09	015274	2951#	2954		
T30L10	015364	2960	2964#		
T30L11	015352	2961#	2968		
T30L12	015640	2924	2970	2994	3010#
T30L13	015436	2972	2976#		
T30L14	015322	2957#	2975	2978	
T30L15	015514	2984	2987#		
T30L16	015534	2988	2991#		
T30L17	015542	2986	2990	2992#	
T30L20	015764	3036#	3040		
T30L21	014600	2852	2857	2861#	
T30L22	014770	2889	2892#		
T30L25	015144	2906	2925#		
T30L26	015042	2908#	2928	2938	
T30L27	015212	2930	2935#		
T30L28	015006	2902#	2946		
T30L29	015062	2912#	2931		
T30L30	015034	2907#	2933		
T31L01	013162	2585	2588#		
UCNT	002626	1167#	1283*	2888	2899
WINT	005670	1628	1698#		
\$AUTOB	001166	663#			
\$BDADR	001154	658#			
\$BDDAT	001160	660#			
\$BELL	001240	688#	3238	3263	
\$CHARC	017562	3413*	3423*	3430	3439* 3444#
\$CKSWR=	***** U	3562			

STYPE	017346	3394#	3547	3555															
STYPEC	017516	3415	3422	3429	3434#	3435													
STYPEX	017564	3440	3442	3445#															
STYPOC	017612	3477#	3556																
STYPON	017626	3476	3479#	3558															
STYPOS	017566	3472#	3557																
SXTSTR	016400	3178#																	
SSGET4=	000000	3076#																	
SOFILL	020011	3473*	3477*	3487	3522#														
S4OCAT=	***** U	3175	3245																
=	027722	613#	617#	622#	633	634#	636#	638#	645#	692	1177	1192	1193	3084					
		3088	3217	3218	3263	3309#	3376#	3447	3578	3600	3591#	3722#	4320#	4324#					

COMMEN	1#	610#													
ENDCOM	1#	610#													
ERROR	504#	1241	1326	1348	1384	1426	1455	1493	1539	1580	1583	1587	1591	1665	1677
	1682	1694	1700	1705	1730	1760	1777	1796	1814	1829	1832	1835	1838	1857	1868
	1891	1933	1938	1944	1997	2026	2055	2058	2089	2108	2115	2120	2123	2126	2162
	2205	2210	2255	2260	2299	2304	2307	2313	2337	2367	2368	2372	2373	2378	2379
	2384	2385	2389	2390	2400	2401	2426	2427	2433	2434	2452	2453	2484	2486	2488
	2498	2500	2501	2504	2506	2507	2510	2512	2515	2517	2543	2544	2548	2549	2553
	2554	2560	2561	2586	2612	2613	2626	2627	2639	2640	2648	2649	2699	2721	2745
	2778	2779	2796	2797	2912	2992									
ESCAPE	1#	610#													
GETPRI	1#	610#													
GETSWR	1#	610#													
MSG	1304#	1306	1355#	1357	1407#	1409	1436#	1438	1465#	1467	1517#	1519	1552#	1554	1597#
	1599	1708#	1710	1734#	1736	1895#	1897	1948#	1950	2002#	2004	2033#	2035	2063#	2065
	2130#	2132	2165#	2167	2216#	2218	2266#	2268	2317#	2319	2344#	2346	2406#	2408	2457#
	2459	2521#	2523	2590#	2592	2655#	2657	2729#	2731	2750#	2752	2814#	2816		
MULT	1#	610#													
NEWTST	1#	610#	1304	1355	1407	1436	1465	1517	1552	1597	1708	1734	1843	1871	1895
	1948	2002	2033	2063	2130	2165	2216	2266	2317	2344	2406	2457	2521	2572	2590
	2655	2729	2750	2814											
POP	1#	610#	3363	3587	3588										
PUSH	1#	610#	3322	3568	3574										
REPORT	1#	610#													
SCOPE	505#	1311	1365	1415	1444	1477	1523	1560	1617	1715	1743	1846	1874	1900	1956
	2009	2039	2071	2136	2176	2227	2276	2323	2355	2414	2467	2530	2575	2600	2660
	2736	2760	2804	2833	3059										
SETPRI	1#	610#													
SETTRA	3547#	3556	3557	3558	3559										
SETUP	1#	610#	1171												
SKIP	1#	610#	1345	1509	1667	1670	1684	1696	1704	1779	1798	1808	1816	1831	1834
	1837	1859	1863	1886	1930	1935	1940	2091	2107	2110	2117	2122	2125	2156	2200
	2207	2250	2257	2303	2341	2603									
SLASH	1#	610#	1244	1246	1288	1290	1526	1529	2801	2803	3088	3090	3098	3100	3111
	3113	3124	3126	3132	3134	3149	3151	4321	4323						
SPACE	610#														
STARS	1#	610#	631	641	692	1169	1304	1310	1355	1364	1407	1414	1436	1443	1465
	1476	1517	1522	1552	1559	1597	1616	1708	1714	1734	1742	1843	1845	1871	1873
	1895	1899	1948	1955	2002	2008	2033	2038	2063	2070	2130	2135	2165	2175	2216
	2226	2266	2275	2317	2322	2344	2354	2406	2413	2457	2466	2521	2529	2572	2574
	2590	2599	2655	2659	2729	2735	2750	2759	2814	2832	3052	3163	3220	3265	3312
	3379	3449	3526	3564	3580	3605									
SWRSU	1#	610#	1194#												
TRMTRP	3547#														
TYPBIN	1#	610#													
TYPDEC	1#	610#	3071												
TYPNAM	1#	610#													
TYPNUM	1#	610#													
TYPOCS	1#	610#													
TYPOCT	1#	610#	1232	1285	2917	2921	2996	2999	3003	3006	3155	3277	3301		
TYPTXT	1#	610#													
\$\$CMRE	639#	678	679	680	681										
\$\$CMTM	639#	682	683	684	685										
\$\$ESCA	1#	610#													
\$\$NEWT	1#	610#	1304	1355	1407	1436	1465	1517	1552	1597	1708	1734	1843	1871	1895
	1948	2002	2033	2063	2130	2165	2216	2266	2317	2344	2406	2457	2521	2572	2590

ADD	1223	1261	1262	1263	1264	1265	1266	1267	1296	1661	1688	1966	2441	2475	3285
ASL	3342	3404	3475	3485											
ASLB	1280	1282	1431	1671	2925	2935	2965	3282	3283	3284	3536				
ASR	3347														
BCC	1269														
BCS	3348														
BEQ	1271														
	1276	1279	1321	1323	1330	1380	1388	1422	1451	1486	1488	1490	1499	1509	1574
	1578	1680	1704	1729	1755	1772	1791	1808	1825	1856	1863	1886	1930	1994	2021
	2023	2050	2054	2094	2104	2107	2148	2193	2290	2303	2383	2388	2399	2425	2445
	2451	2478	2497	2559	2585	2603	2697	2706	2717	2852	2857	2867	2870	2889	2928
	2938	2968	2970	2984	2988	3013	3016	3024	3028	3075	3120	3190	3194	3203	3234
	3237	3252	3255	3287	3292	3305	3407	3442	3502						
BGE	3206														
BGT	3066	3356	3509												
BHI	626	3192													
BIC	1270	1378	1397	1672	1942	2060	2447	2683	2926	2936	2966	3063	3499		
BIS	1222	2710	3350	3351	3504	3505									
BISB	3274														
BIT	1210	1229	1275	1278	1329	1341	1387	1485	1544	1568	1577	1669	1679	1703	1728
	1821	1824	1920	2101	2103	2106	2147	2327	2335	2340	2365	2370	2382	2387	2398
	2424	2442	2444	2450	2477	2479	2481	2494	2496	2541	2546	2558	2602	2646	2662
	2705	2836	2913	2927	2937	2967	2983	2987	2993	3119	3152	3175	3193	3200	3236
	3243	3251													
BITB	3430														
BLE	2114														
BLOS	2204	2254													
BLT	3339	3355	3421	3510											
BMI	1430	3118	3346												
BNE	1177	1201	1211	1226	1230	1238	1240	1258	1273	1295	1328	1337	1342	1386	1399
	1405	1434	1463	1497	1501	1511	1545	1569	1625	1631	1670	1789	1822	1861	1884
	1910	1914	1921	1969	1981	1987	2052	2078	2088	2096	2098	2100	2102	2143	2152
	2155	2183	2195	2197	2199	2234	2243	2245	2247	2249	2288	2328	2336	2341	2366
	2371	2443	2480	2482	2495	2540	2542	2547	2622	2625	2635	2638	2647	2663	2697
	2690	2693	2768	2777	2785	2788	2791	2835	2837	2914	2946	2954	2972	2994	3040
	3042	3139	3145	3153	3176	3201	3244	3259	3275	3297	3344	3401	3409	3417	3431
	3438	3500	3586												
BPL	1459	1576	1727	2942	3020	3022	3026	3030	3249	3330	3360	3395	3435	3498	
BR	1203	1235	1281	1333	1335	1345	1432	1461	1504	1506	1513	1515	1543	1585	1589
	1657	1667	1673	1684	1696	1779	1798	1816	1831	1834	1837	1859	1935	1940	1970
	2057	2091	2110	2117	2122	2125	2156	2200	2207	2250	2257	2301	2306	2312	2431
	2436	2483	2490	2503	2509	2514	2556	2615	2629	2642	2701	2781	2792	2877	2880
	2883	2933	2975	2978	2986	2990	3178	3184	3187	3196	3199	3280	3307	3341	3358
	3397	3414	3424	3433	3440	3476	3491	3512	3578	3600					
CCC	1660														
CLC	1268														
CLR	1175	1189	1190	1214	1228	1256	1317	1319	1339	1340	1344	1352	1371	1542	1549
	1570	1571	1622	1721	1724	1752	1764	1820	1849	1877	1904	1960	2013	2016	2030
	2048	2076	2084	2085	2139	2141	2181	2188	2212	2238	2262	2280	2291	2292	2297
	2310	2333	2339	2375	2381	2404	2429	2438	2472	2537	2551	2557	2569	2579	2618
	2619	2631	2632	2653	2670	2685	2707	2715	2720	2763	2764	2773	2862	2873	2890
	2901	2909	2940	2962	3018	3037	3060	3061	3091	3092	3093	3094	3095	3096	3105
	3114	3115	3136	3142	3198	3212	3273	3333	3336	3489	3584				
CLRB	1300	3197	3362	3413	3439										
CMP	625	1176	1200	1225	1237	1239	1257	1294	1320	1327	1336	1379	1385	1398	1404
	1421	1433	1450	1462	1487	1489	1496	1498	1500	1508	1510	1573	1624	1630	1668

	1675	1698	1754	1771	1788	1790	1807	1860	1862	1883	1885	1909	1913	1929	1968
	1986	2018	2022	2051	2053	2077	2087	2095	2099	2113	2142	2151	2154	2182	2194
	2198	2203	2233	2242	2244	2246	2248	2253	2539	2616	2624	2637	2643	2680	2696
	2712	2767	2782	2784	2787	2790	2886	2888	2892	2953	2964	2969	2971	2979	3038
	3039	3138	3143	3144	3185	3205	3258	3354							
CMPB	3191	3406	3408	3416	3437	3441									
COM	1377	1396													
DEC	2900	2944	3064	3281											
DECB	2810	3420	3423	3497	3508										
EMT	504														
HALT	617	3250	3260	3396	3577	3599									
INC	1283	1572	1658	1659	1662	1689	1691	1725	1823	1911	1912	1918	1984	1991	2019
	2085	2153	2446	2538	2620	2623	2633	2636	2742	2775	2789	2872	2887	2908	2961
	3062	3116	3204	3239	3340	3503	3511	3585							
INCB	3209	3233	3443												
IOT	505														
JMP	621	628	1243	1274	1277	1353	1391	1428	1457	1495	1582	1594	1762	1827	2664
	2806	2840	2924	3082											
JSR	627	1242	1332	1349	1390	1427	1456	1480	1494	1507	1540	1563	1579	1581	1584
	1588	1592	1593	1595	1620	1664	1666	1676	1678	1683	1693	1695	1699	1701	1702
	1706	1718	1731	1732	1747	1753	1761	1763	1770	1778	1781	1787	1797	1806	1815
	1830	1833	1836	1839	1848	1854	1858	1869	1876	1882	1892	1902	1903	1908	1919
	1928	1934	1939	1945	1946	1958	1959	1979	1985	1992	1998	1999	2012	2027	2031
	2042	2056	2059	2061	2074	2090	2109	2116	2121	2124	2127	2128	2138	2163	2179
	2206	2211	2214	2230	2256	2261	2264	2279	2286	2298	2300	2305	2308	2311	2314
	2315	2326	2338	2358	2364	2369	2374	2380	2386	2391	2397	2402	2403	2417	2423
	2428	2435	2437	2449	2454	2455	2469	2476	2489	2502	2508	2513	2518	2519	2533
	2545	2550	2555	2562	2563	2564	2568	2570	2577	2578	2583	2587	2588	2614	2628
	2641	2650	2668	2700	2722	2723	2739	2746	2747	2762	2780	2798	2799	2805	3009
	3077	3245	3415	3422	3429										
MOV	623	624	1174	1178	1180	1181	1182	1183	1184	1185	1186	1187	1188	1192	1193
	1196	1197	1198	1199	1204	1206	1207	1208	1215	1216	1217	1218	1219	1220	1227
	1232	1247	1248	1249	1250	1251	1252	1253	1254	1255	1259	1260	1285	1291	1292
	1293	1298	1299	1312	1313	1314	1315	1316	1318	1324	1325	1338	1343	1347	1351
	1366	1367	1368	1369	1370	1372	1373	1374	1375	1376	1381	1382	1383	1392	1393
	1394	1395	1400	1401	1402	1403	1416	1417	1418	1419	1420	1423	1424	1425	1445
	1446	1447	1448	1449	1452	1453	1454	1478	1479	1481	1482	1483	1484	1491	1492
	1502	1503	1512	1524	1530	1531	1534	1535	1536	1537	1541	1546	1548	1550	1561
	1562	1564	1565	1566	1567	1618	1619	1621	1623	1626	1627	1628	1629	1632	1633
	1634	1635	1636	1638	1639	1640	1641	1643	1644	1645	1646	1651	1652	1653	1654
	1655	1681	1686	1687	1690	1716	1717	1719	1720	1744	1745	1746	1748	1749	1750
	1751	1756	1757	1758	1759	1765	1766	1767	1768	1769	1773	1774	1775	1776	1782
	1783	1784	1785	1786	1792	1793	1794	1795	1800	1801	1802	1803	1804	1805	1810
	1811	1812	1813	1847	1850	1851	1852	1853	1864	1865	1866	1867	1875	1878	1879
	1880	1881	1887	1888	1889	1890	1901	1905	1906	1907	1915	1916	1917	1925	1926
	1927	1931	1932	1936	1937	1941	1943	1957	1964	1965	1973	1974	1975	1976	1977
	1978	1982	1983	1988	1989	1990	1995	2000	2010	2011	2014	2015	2024	2025	2028
	2029	2040	2041	2043	2044	2045	2046	2047	2072	2073	2075	2079	2080	2081	2082
	2083	2092	2105	2118	2119	2137	2140	2144	2145	2146	2149	2150	2159	2160	2161
	2177	2178	2180	2184	2185	2186	2187	2190	2191	2208	2209	2228	2229	2231	2232
	2235	2236	2237	2240	2241	2258	2259	2277	2278	2281	2282	2283	2284	2293	2294
	2295	2296	2309	2324	2325	2331	2332	2356	2357	2359	2360	2361	2362	2363	2377
	2392	2393	2394	2395	2396	2415	2416	2418	2419	2420	2421	2422	2432	2439	2440
	2448	2468	2470	2471	2473	2474	2485	2487	2491	2492	2499	2505	2511	2516	2531
	2532	2534	2535	2536	2565	2566	2567	2576	2580	2581	2582	2601	2604	2605	2606
	2607	2608	2609	2610	2617	2630	2644	2651	2652	2661	2665	2666	2667	2669	2672

	2673	2674	2675	2676	2677	2678	2681	2684	2688	2691	2694	2695	2698	2702	2703
	2704	2709	2713	2718	2719	2724	2725	2737	2738	2740	2741	2743	2761	2765	2766
	2769	2770	2771	2772	2774	2783	2793	2794	2795	2808	2809	2839	2844	2845	2846
	2847	2848	2849	2850	2853	2854	2855	2858	2859	2860	2861	2863	2864	2865	2868
	2871	2875	2876	2878	2879	2881	2882	2884	2885	2893	2899	2902	2903	2905	2906
	2907	2911	2915	2917	2919	2921	2930	2931	2932	2939	2943	2950	2951	2952	2955
	2956	2957	2958	2959	2960	2973	2974	2976	2977	2980	2981	2982	2985	2989	2991
	2996	2999	3003	3006	3010	3011	3014	3017	3034	3035	3036	3067	3071	3074	3101
	3102	3104	3106	3121	3127	3129	3130	3135	3137	3140	3141	3146	3147	3155	3180
	3181	3183	3186	3195	3207	3208	3210	3211	3214	3215	3235	3240	3253	3256	3272
	3277	3286	3291	3296	3298	3302	3323	3324	3325	3326	3327	3328	3329	3334	3337
	3357	3363	3364	3365	3366	3367	3369	3370	3398	3399	3403	3418	3472	3480	3481
	3482	3488	3495	3513	3514	3515	3516	3517	3532	3533	3537	3543	3544	3566	3567
	3568	3569	3570	3571	3572	3573	3574	3575	3576	3582	3583	3587	3588	3589	3590
	3591	3592	3593	3594	3595										
MOV8	1191	3213	3242	3332	3335	3349	3352	3361	3400	3428	3436	3473	3474	3477	3478
	3479	3483	3486	3487	3506	3535									
NEG	3331	3484													
NOP	1637	1642	1647	1656	1663	1692	1722	2017	2189	2213	2239	2263	2285	2334	2376
	2430	2552	2611	2708	2744	2910	3078	3079	3080						
RESET	1301	3076													
ROL	1224	3490	3492	3493	3494	3496									
ROR	1460														
RTI	1205	3216	3262	3371	3405	3518	3545	3598							
RTS	1928	3097	3107	3122	3131	3148	3159	3300	3445	3538					
SUB	1971	1996	3241	3338											
SXT	1532														
TRAP	3547	3556	3557	3558	3559										
TST	1221	1272	1322	1334	1429	1458	1505	1514	1538	1826	1855	1967	1980	1993	2020
	2049	2093	2097	2111	2112	2158	2192	2196	2201	2202	2251	2252	2287	2289	2302
	2584	2621	2634	2686	2689	2692	2716	2776	2786	2834	2851	2856	2866	2869	2904
	2945	3012	3015	3023	3027	3041	3103	3128	3182	3202	3248	3254	3304	3343	3353
	3402	3410	3432	3501	3534										
TSTB	1575	1726	2941	3019	3021	3025	3029	3117	3189	3345	3359	3394	3434		
WAIT	2493	2645	2671	2679	2682	2711	2714	2874	2891	2963					
.ASCII	689	690	3609	3614	3623	4070	4273								
.ASCIZ	688	691	3085	3308	3602	3607	3630	3637	3646	3655	3661	3667	3674	3678	3685
	3694	3701	3708	3714	3724	3733	3739	3745	3752	3759	3764	3773	3777	3787	3793
	3797	3807	3809	3818	3821	3829	3835	3841	3847	3854	3865	3871	3877	3883	3887
	3891	3896	3901	3908	3913	3920	3923	3931	3935	3942	3944	3950	3960	3968	3976
	3982	3988	3997	4002	4009	4013	4020	4025	4033	4041	4046	4051	4059	4078	4086
	4089	4097	4102	4105	4112	4116	4120	4125	4130	4141	4150	4153	4157	4163	4168
	4177	4181	4185	4192	4196	4199	4203	4207	4213	4219	4223	4230	4239	4248	4251
	4254	4259	4264	4280	4288	4294	4301	4307							
.BLKW	3376	4324													
.BYTE	648	649	654	655	663	664	672	673	674	675	3084	3519	3520	3521	3522
.ENABL	1	473													
.END	4325														
.ENDC	478	492	494	495	504	596	610	611	622	632	636	638	642	646	648
	676	682	686	687	688	689	693	1171	1178	1179	1182	1184	1186	1188	1189
	1190	1192	1194	1210	1245	1247	1289	1291	1305	1306	1310	1311	1312	1346	1356
	1357	1364	1365	1366	1408	1409	1414	1415	1416	1437	1438	1443	1444	1445	1466
	1467	1476	1477	1478	1510	1518	1519	1522	1523	1524	1527	1530	1553	1554	1559
	1560	1561	1598	1599	1616	1617	1618	1668	1671	1685	1697	1705	1709	1710	1714
	1715	1716	1735	1736	1742	1743	1744	1780	1799	1809	1817	1832	1835	1838	1844
	1845	1846	1847	1860	1864	1872	1873	1874	1875	1887	1896	1897	1899	1900	1901

	1179	1182	1188	1189	1190	1192	1193	1233	1286	2919	2922	2997	3000	3004	3007
	3054	3060	3061	3072	3084	3088	3156	3167	3168	3169	3170	3171	3175	3198	3199
	3214	3217	3218	3224	3225	3226	3227	3228	3233	3251	3258	3263	3278	3303	3447
.IRP	3555	3556	3557	3558	3559										
	611	1304	1355	1407	1436	1465	1517	1552	1597	1708	1734	1843	1871	1895	1948
	2002	2033	2063	2130	2165	2216	2266	2317	2344	2406	2457	2521	2572	2590	2655
.LIST	2729	2750	2814	3323	3363	3568	3574	3587	3588						
	1	473	610	611	617	676	678	679	680	681	682	683	684	685	686
	1169	1170	1171	1194	1304	1312	1355	1366	1407	1416	1436	1445	1465	1478	1517
	1524	1552	1561	1597	1618	1708	1716	1734	1744	1843	1847	1871	1875	1895	1901
	1948	1957	2002	2010	2033	2040	2063	2072	2130	2137	2165	2177	2216	2228	2266
	2277	2317	2324	2344	2356	2406	2415	2457	2468	2521	2531	2572	2576	2590	2601
	2655	2661	2729	2737	2750	2761	2814	2834	3060	3076	3171	3258	3547	3555	3556
.MACRO	3557	3558	3559	3560	3605	3606	3607								
	1	495	639	1304	1355	1407	1436	1465	1517	1552	1597	1708	1734	1895	1948
	2002	2033	2063	2130	2165	2216	2266	2317	2344	2406	2457	2521	2590	2655	2729
.MCALL	2750	2814	3547												
.NLIST	473	610	1194												
	1	473	610	611	617	676	678	679	680	681	682	693	684	685	686
	1169	1170	1171	1194	1304	1312	1355	1366	1407	1416	1436	1445	1465	1478	1517
	1524	1552	1561	1597	1618	1708	1716	1734	1744	1843	1847	1871	1875	1895	1901
	1948	1957	2002	2010	2033	2040	2063	2072	2130	2137	2165	2177	2216	2228	2266
	2277	2317	2324	2344	2356	2406	2415	2457	2468	2521	2531	2572	2576	2590	2601
	2655	2661	2729	2737	2750	2761	2814	2834	3060	3076	3171	3258	3547	3555	3556
.PAGE	3557	3558	3559	3560	3605	3606	3607								
.REF	639	693													
.SECT	1														
.SETTL	617	678	682	1169	3605										
	484	500	611	620	629	639	693	1172	1304	1355	1407	1436	1465	1517	1552
	1597	1708	1734	1843	1871	1895	1948	2002	2033	2063	2130	2165	2216	2266	2317
	2344	2406	2457	2521	2572	2590	2655	2729	2750	2814	3050	3161	3218	3263	3310
.TITLE	3377	3447	3524	3547	3562										
.WORD	473														
	617	618	619	637	647	650	651	652	653	656	657	658	659	660	661
	662	665	666	667	676	678	679	680	681	682	683	684	685	1128	1129
	1130	1131	1132	1133	1134	1135	1136	1137	1138	1139	1140	1141	1142	1143	1144
	1145	1146	1147	1148	1149	1150	1151	1152	1153	1154	1155	1156	1157	1158	1159
	1160	1161	1162	1163	1164	1165	1166	1167	1168	3065	3068	3083	3289	3294	3444
	3523	3554	3597	3666	3677	3692	3723	3863							

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

*.DZKUBA/SOL/CRF/PAGNUM=SYSMAC.SML(400,1066),DZKUBA(400,4571)
RUN-TIME: 44 61 8 SECONDS
RUN-TIME RATIO: 228/114=2.0
CORE USED: 34K (67 PAGES)

