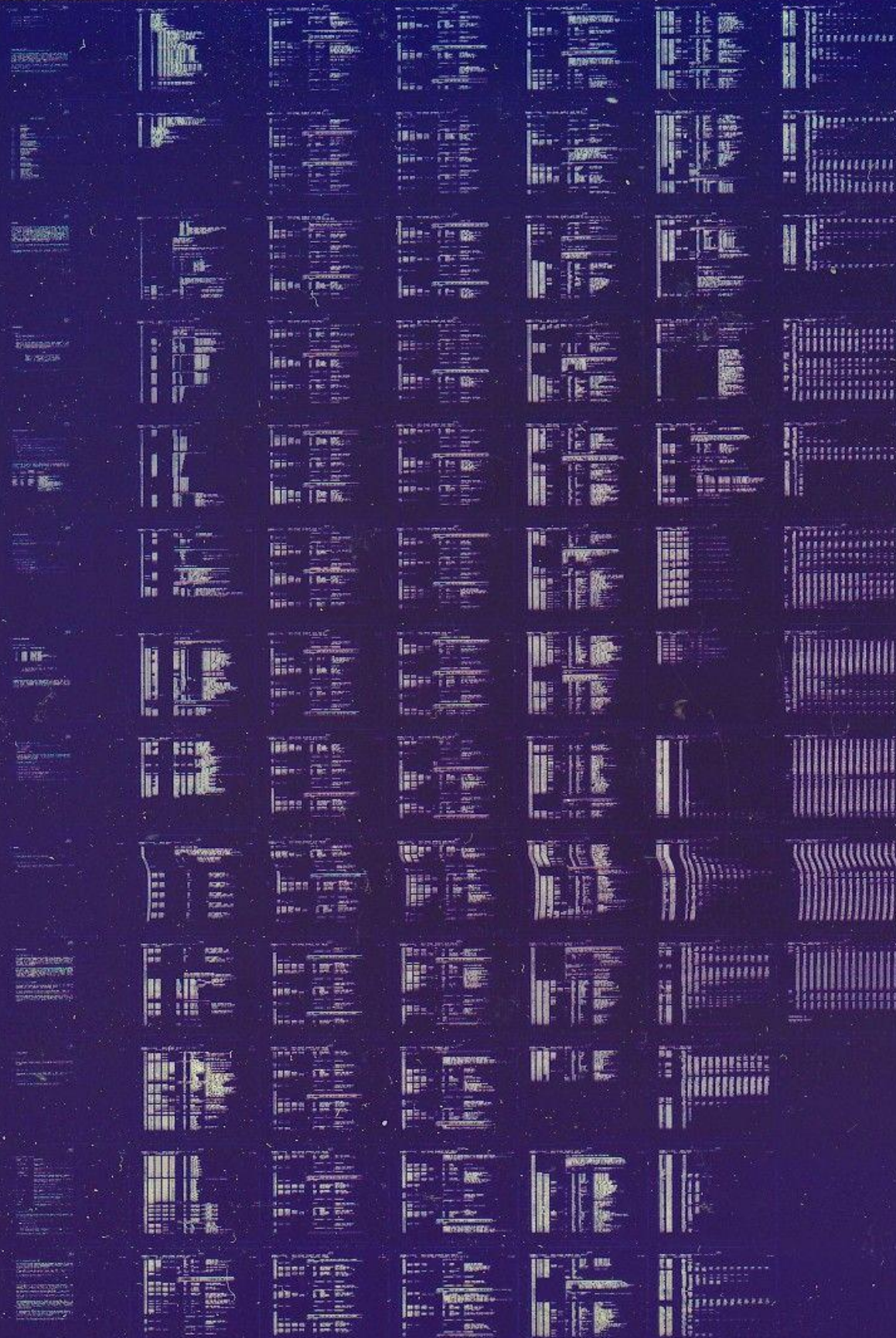


IBV11-A

IBV11-A DIAGNOSTIC
MD-11-DVIBA-A

EP-DVIBA-A-DL-A
COPYRIGHT © 1977
FICHE 1 OF 1

JUN 1977
digital
MADE IN USA



11-11-77

HDR1DVIBARSEQ

00010000

770526

801
PDP10 411

IDENTIFICATION

SEQ 0001

Product Code: MAINDEC-11-DVIBA-A-D

Product Name: IBV11-A Diagnostic

Date Created: JAN 1977

Maintainer: Diagnostic Engineering

Copyright (C) 1977
Digital Equipment Corporation, Maynard, Mass.

This software is furnished under a license for use only on a single computer system and may be copied only with the inclusion of the above copyright notice. This software, or any other copies thereof, may not be provided or otherwise made available to any other person except for use on such system and to one who agrees to these license terms. Title to and ownership of the software shall at all times remain in DEC.

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation.

DEC assumes no responsibility for the use or reliability of its software on equipment which is not supplied by DEC.

TABLE OF CONTENTS

| | |
|-------|--|
| 1.0 | ABSTRACT |
| 2.0 | REQUIREMENTS |
| 2.1 | Equipment |
| 2.2 | Storage |
| 3.0 | LOADING PROCEDURE |
| 3.1 | Method |
| 3.2 | Non-Standard Address, Vector, or Use of Software Switch Register |
| 4.0 | STARTING PROCEDURE |
| 4.1 | Control Switch Settings |
| 4.2 | Starting Address |
| 4.3 | Program and/or Operator Action |
| 5.0 | OPERATING PROCEDURE |
| 5.1 | Switch Register Function |
| 5.2 | Scope Loops |
| 5.3 | Program and/or Operator Action |
| 5.3.1 | Logic Test |
| 6.0 | ERRORS |
| 6.1 | Error Printout |
| 6.2 | Non-Standard Error Halts |
| 7.0 | RESTRICTIONS |
| 7.1 | Starting Restriction |
| 7.2 | Possible Program "Bombs" |
| 8.0 | MISCELLANEOUS |
| 8.1 | Power Fail |
| 8.2 | XXDP, ACT, APT |
| 8.3 | Execution Time |
| 8.4 | LSI-11 "ODT" Commands |
| 8.5 | Entering LSI-11 "ODT" |
| 8.6 | Use of Program Software SWR |
| 8.7 | Trap Catcher |

1.0 ABSTRACT

This program allows the user to check-out or debug the IBV11-A, LSI/IB interface option. In order to check-out a greater portion of logic on this option, a second IBV11-A is needed. See section 2.1. When a second IBV11-A can be obtained in order to run this diagnostic, the user must inform this diagnostic to exercise the logic on one IBV11-A that requires a KGM (Known Good Module). Please note that the second IBV11-A should be known good. No attempt is made to checkout the KGM and no conclusion that if good passes are made through this diagnostic that the KGM is also good. Signals "SRC", "eri", "BIAKI", "DA11" and "ERIHS" are not tested on the IBV11-A if a KGM is not used.

If the user is unfamiliar with an LSI-11 he should review sections 8.4 and 8.5. A software switch register is included with this program.

Every effort was made to make this program conform to LSI-11 programming restrictions. However, the user should read sections 7.1 and 7.2.

2.0 REQUIREMENTS

2.1 Equipment

1. PDP-11 Family Computer with 4K of memory (or more) and console I/O facilities (i.e., TTY).
2. IBV11-A under test.
3. (Optional) Second IBV11-A "KGM" (known good module). The "KGM" must be electrically second on the CSI-11-BUS. It must have an instrumentation Bus Cable between it and the first IBV-11. Its base address should be 760160 and vector address of 660 (see section 3.2 if different).

NOTE

While it is generally recommended that a "KGM" is used, if one is available, deposit a "000001" into location "SCDW1". No test will be performed that requires the "KGM" if SCDW1 is zero.

2.2 Storage

This program occupies and uses the lower 4K of memory.

3.0 LOADING PROCEDURE

3.1 Method

Standard procedure for normal binary tapes should be followed.

1. Absolute loader must be in memory.
2. Place binary tape in reader.
3. Type address #7500 (* determine by location of loader).
4. Type "G" (program will be loaded into memory).

The program can also be loaded by XXDP, ACT or APT.

3.2 Non-Standard Address, Vector, or Use of Software Switch Register

This program is set to test a IBV11-A with a standard address and vector. If any of these are different on the IBV11-A you are testing, change the corresponding location in memory before starting this test.

| TAG | ADDRESS | CURRENT CONTENTS | COMMENTS |
|----------|---------|------------------|--------------------------------------|
| \$BASE: | 1250 | 160150 | :: BASE ADDRESS OF EQUIPMENT |
| \$VECT1: | 1244 | 000640 | :: UNDER TEST |
| \$WREG: | 176 | 000000 | :: INTERRUPT VECTOR #1 |
| IBS2: | 1366 | 160160 | :: MANUAL SWR. |
| VECTA2: | 1372 | 660 | :: ADDRESS OF SECOND IBV11-A. |
| \$CDW1: | 1254 | 000000 | :: VECTOR ADDRESS OF SECOND IBV11-A. |
| | | | :: DEVICE DESCRIPTOR WORD #1 |
| | | | (if = 000001 to use |
| | | | "KGM" in testing 1st IBV-11) |
| | | | (Default = 00000 to disable |
| | | | use of KGM in tests. |

4.0 STARTING PROCEDURE

4.1 Control Switch Setting

Before starting the diagnostic, set all switch register bits as desired. See section 5.1.

4.2 Starting Addresses

200 Start of Logic Tests

4.3 Program and/or Operator Action

1. Load program into memory.
2. Enter keyboard "OOT".
3. Alter location "SWREG" to reflect desired options of a switch register - See section 5.1.
4. Type starting address, followed by "G" to start program.

5.0 OPERATING PROCEDURE

5.1 Switch Register Function

| SWR BIT | OCTAL | FUNCTION WHEN SET |
|---------|--------|---------------------------------|
| 15 | 100000 | HALT ON ERROR |
| 14 | 040000 | LOOP ON TEST |
| 13 | 020000 | INHIBIT ERROR TYPEOUT |
| 11 | 004000 | INHIBIT ITERATIONS (SHORT PASS) |
| 10 | 002000 | BELL ON ERROR |
| 09 | 001000 | LOOP ON ERROR |
| 08 | 000400 | LOOP ON TEST IN SWR <7:0> |

NOTE

The Switch Register may be changed at any time while the diagnostic is running by typing "tG".

5.2 Scope Loops

If an error occurs and the user wishes to scope the error, "SWREG" should be altered to "100000" at the start of the test to halt on error, then when the program halts on error and the CPU enters "OOT", "SSWREG" should be altered to "060000" to loop on current test and inhibit error typeout, then type "P" to continue program execution.

5.3 Program and/or Operator Action

1. When the program is initially started it will type:

MD-11-DVIBA-A

SWR=000000 NEW=

2. Program now waits for the operator to enter a switch register setting (see section 8.6). If the program is restarted, only "MD-11-DVIBA-A" is typed. To change the switch register setting, see section 8.6.
3. Program executes first pass of logic tests, subtest iterations inhibited.
4. Program reports any errors it detects.
5. Program reports "END PASS 1".
6. Program executes second pass of logic tests, only this time it will loop on each test 2000 times.
7. Program then reports "END PASS 2".
8. Program will continue executing steps 6 and 7 until stopped.

6.0 ERRORS

6.1 Error Printout

Printout varies with the error detected. The error PC typed out is the actual location of the error call.

6.2 Non-Standard Error Halt

Bus errors will cause a Halt in the routine "IOTRD". The address that caused this trap will be in "TRTO".

7.0 RESTRICTIONS

7.1 Starting Restriction

If a free-running clock, such as 60Hz from the power supply, is attached to the "BEVNT" bus line on both REV level C/D and E systems, an interrupt to location 100 will occur when using the "G" and "L" commands prior to executing the first instruction. Therefore this program can not disable the BEVNT bus line by inhibiting interrupts.

User systems requiring a free-running clock attached to the BEVNT bus line can temporarily avoid this situation by setting the PSW(RS) to 200, loading the PC with the starting address instead of using the "G" command, and then using the "P" command. Before using the "L" command, the PSW(RS) can be set to 200, thereby inhibiting interrupts, to avoid receiving the event interrupt after loading the ABS loader.

7.2 Possible Program "BOMBS"

The first two tests of this program check to see if the IBV11-A responds to the address the program thinks its at. If the IBV11-A does not respond, a bus error occurs.

For more information on the next subject, see JAN. 1976 LSI-11 engineering bulletin issued by the Digital Components Group.

Bus errors may alter the preset contents of location 4 before the trap is executed, thereby transferring program control to area in the program that was not set up to handle the trap. If this happens, the program will "BOMB" and possibly rewrite parts of itself.

8.0 MISCELLANEOUS

8.1 Power Fail

After a power failure occurs, the program execution will continue at the point where the power occurred. The program will type "POWER".

8.2 XXDP, ACT, APT

The program is chainable under XXDP, ACT, or APT. Although "APT HOOKS" have been installed, they have not been tested.

8.3 Execution Time

0.1 Minutes (6 sec) Iteration Inhibited - No Errors
0.5 Minutes (30 sec) With Iterations - No Errors

8.4 LSI-11 "ODT" Commands

| <u>FORMAT</u> | <u>DESCRIPTION</u> |
|----------------|--|
| <CR> RETURN | Close opened location and accept next command. |
| <LF> LINE FEED | Close current location; open next sequential location. |
| ↑(UPARROW) | Open previous location. |
| < (LEFT ARROW) | Take contents of opened location, indexed by contents of PC, and open that location. |
| Ⓜ | Take contents of opened location as absolute address and open that location. |
| R/ | Open the word at location R. |
| / | Reopen the last location. |
| SN/ or RN/ | Open general register N(0-7) or S(PS register). |
| R:G or RG | GOTO location R and start program. |
| NL | Execute Bootstrap loader using N as device CSR. Console device is 177560. |
| :P or P | Proceed with program execution. |
| RUBOUT | Erases previous numeric character. Response is a backslash (). |

8.5 Entering LSI-11 "ODT"

The halt or ODT microcode state of the KD11F (LSI-11 module) can be entered in five different ways (others are a subset of these) from the run state:

1. Execution of a LSI-11 halt instruction.
2. A double bus error.
3. As a power up option.
4. ASCII break with DLV11 framing error asserting the B halt line (enabled by jumper of DLV11).

Upon entering the halt state, the KD11F responds through the set of command listed in section 8.4.

8.6 Use of Program Software SWR

The software switch register may be changed by typing \uparrow G (control and letter G keys typed simultaneously). When \uparrow G is typed, the program responds by typing "SWR=XXXXXX" where XXXXXX equals the former contents of the switch register.

If you wish to keep the current value, type <CR>. If you wish to change the value, type the new value followed by a <CR>.

It is important to note that the diagnostic is not running after the \uparrow G until a <CR> is typed.

8.7 Trap Catcher

The Trap Catcher in this diagnostic employs a new concept. This concept will enable the user of this diagnostic to gain more knowledge of the events that lead the program to this area.

The Trap Catch consists of PC+2 and JSR PC,RO. (i.e., Location 300 would contain 302 and location 302 would contain 4700).

When a device interrupts to the Trap Catcher, it would pick up the PC+2 of the trap as an address of the interrupt service routine.

The program would then pick up "4700" as the new PSW. Bit 7 of the new PSW having been 1, would cause further interrupts from happening. When the CPU attempts to execute "4700" (JSR PC,RO), a Bus-time-out trap will occur to location 4. Location 4 contains a pointer to "IOTRD", a routine that will report the trap as an error.

To guard against "Real" Bus errors routing us through location 4 to "IOTRD", we check to see if the trap that brought us to location 4 really came from the Trap Catcher area. If not we'll halt and leave the Trap Address in "TRTO".

More about the interrupt error can be found in the description of the error in the program listing in the routine "IOTRD".

| | |
|------|---|
| 22 | OPERATIONAL SWITCH SETTINGS |
| 34 | TRAP CATCHER |
| 53 | BASIC DEFINITIONS |
| 169 | ACT11 HOOKS |
| 182 | RPT PARAMETER BLOCK |
| 205 | COMMON TAGS |
| 249 | RPT MAILBOX-ETABLE |
| 298 | ERROR POINTER TABLE |
| 369 | REG ADDRESS AND COMMON TAGS |
| 401 | PROGRAM START |
| 405 | INITIALIZE THE COMMON TAGS |
| 505 | TYPE PROGRAM NAME |
| 510 | GET VALUE FOR SOFTWARE SWITCH REGISTER |
| 526 | T1 *TEST THE ADDRESSABILITY OF THE IBS, IBO REGISTERS |
| 581 | T2 *TEST THAT BASE ADDRESSES +4, +6 RESPOND WHEN ADDRESSED |
| 634 | T3 *TEST THAT IBS IS CLEAR AT INIT OF TESTING |
| 656 | T4 *TEST THAT IBO IS CLEAR AT INIT OF TESTING |
| 679 | T5 *TEST THAT BASE ADDRESSES +4, +6 RETURN ZERO WHEN READ |
| 718 | T6 *TEST THAT WE CAN SET TCS, TCS SETS CMD |
| 755 | T7 *TEST THAT EOP WILL SET |
| 792 | T10 *TEST THAT RE WILL SET + CLEAR |
| 829 | T11 *TEST THAT IBC WILL SET AND CLEAR |
| 872 | T12 *TEST THAT TON (BIT05) AND TKR SET AND CLEAR |
| 910 | T13 *MAKE SURE WE CAN SET AND CLEAR BIT06 (IE) |
| 954 | T14 *TEST THAT BIT 7 (ACC) CAN BE SET AND CLEARED |
| 993 | T15 *TEST THAT IBO BIT 0 CAN BE SET + CLEARED |
| 1033 | T16 *TEST THAT IBO BIT 1 CAN BE SET + CLEARED |
| 1073 | T17 *TEST THAT IBO BIT 2 CAN BE SET + CLEARED |
| 1113 | T20 *TEST THAT IBO BIT 3 CAN BE SET + CLEARED |
| 1153 | T21 *TEST THAT IBO BIT 4 CAN BE SET + CLEARED |
| 1193 | T22 *TEST THAT IBO BIT 5 CAN BE SET + CLEARED |
| 1233 | T23 *TEST THAT IBO BIT 6 CAN BE SET + CLEARED |
| 1273 | T24 *TEST THAT IBO BIT 7 CAN BE SET + CLEARED |
| 1313 | T25 *TEST THAT NO DATA GETS XFERRERD, IF NOT ENABLED |
| 1337 | T26 *TEST IBO BITS DAC, AND DAY |
| 1402 | T27 *TEST THAT REN SETS WHEN REM SETS, ALSO TEST CLEAR |
| 1443 | T30 *TEST THAT IFC SETS WHEN IBS SETS, ALSO TEST CLEAR |
| 1487 | T31 *TEST THAT ATN SETS WHEN TCS SETS, ALSO TEST CLEAR |
| 1528 | T32 *TEST THAT EOI SETS WHEN EOP SETS, ALSO TEST CLEAR |
| 1567 | T33 *TEST THAT RFD SET WHEN CSR CLEAR, CLEAR WHEN ACC SET |
| 1601 | T34 *TEST THAT WE CAN GENERATE AN ER2 |
| 1624 | T35 *TEST THAT BUS INIT CLEARS ACC, TON, LON, REM, EIP, TCS |
| 1646 | T36 *TEST IBC CLEARS ACC, TON, LON, REM AND EOP |
| 1670 | T37 *TEST THAT BUS INIT INDIRECTLY CLEARS IBO |
| 1693 | |
| 1694 | INTERRUPT TESTS |
| 1695 | |
| 1697 | T40 *TEST THAT CMD CAN GENERATE AN INTERRUPT B |
| 1731 | T41 *TEST THAT TKR AND LNR CAN GENERATE INTERRUPTS |
| 1796 | T42 *TEST THAT ER2 CAN GENERATE AN INTERRUPT |
| 1838 | |
| 1839 | SECOND MODULE TESTS |
| 1840 | |
| 1842 | T43 *TEST THAT MODULE PASSES "BIAKI" |

MAINDEC-11-DVIBA-A MACY11 27(663) 29-MAR-77 12:57
 DVIBA.P11 TABLE OF CONTENTS

SEQ 0015

| | | |
|------|-----|---|
| 1893 | T44 | *TEST THAT SRQ CAN GENERATE AN INTERRUPT |
| 1944 | T45 | *TEST THAT ERROR1 IS GENERATED IF ATN IS ON THE IB BUS |
| 1975 | T46 | *TEST THAT ERROR 1 IS GENERATED IF IFC IS PUT ON IB BUS BY SECONUD MODULE |
| 2008 | T47 | *TEST THAT ERROR 1 IS GENERATED IF REN IS ON IB BUS |
| 2038 | T50 | *TEST THAT AN ERROR 1 CAN GENERATE AN INTERRUPT |
| 2088 | T51 | *TEST THAT DATA CAN BE XFERRERD BETWEEN THE MODULE UNDER TEST AND THE KGM |
| 2132 | T52 | *TEMP END OF TESTS |
| 2140 | | SYSMAC ROUTINES: |
| 2142 | | END OF PASS ROUTINE |
| 2197 | | ERROR HANDLER ROUTINE |
| 2247 | | ERROR MESSAGE TYPEOUT ROUTINE |
| 2294 | | SCOPE HANDLER ROUTINE |
| 2360 | | TTY INPUT ROUTINE |
| 2499 | | BINARY TO OCTAL (ASCII) AND TYPE |
| 2576 | | CONVERT BINARY TO DECIMAL AND TYPE ROUTINE |
| 2643 | | TYPE ROUTINE |
| 2722 | | APT COMMUNICATIONS ROUTINE |
| 2779 | | POWER DOWN AND UP ROUTINES |
| 2903 | | TRAP DECODER |
| 2926 | | TRAP TABLE |
| 2946 | | MESSAGES AND TABLES |

1
2
3
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
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54

167400

000001

000000

000004 012066 000200
000174 000000
000176 000000
000100 000104 000200 000002
000200 000200 001422

```
.NLIST MC,MD,CND
.LIST ME
.ENABL ABS
.ENABL AMA
.MCALL .HEADER .SETUP .SETTRAP .TRMTRP .STRAP .SRDOCT
.MCALL .STYPBIN, TYPOCS, SPOWER, SCATCH, STYPOCT, .EQUAT
.MCALL .SCMTAG, SWRHI, SEOP, SERROR, SERRTYP
.MCALL .STYPDEC, SSCOPE, SREAD, STYPE
.MCALL .SACT11, SAPTHDR, SAPTYPE
SSWR=167400
```

```
.TITLE MAINDEC-11-DVIBA-A
*COPYRIGHT (C) 1976
*DIGITAL EQUIPMENT CORP.
*MAYNARD, MASS. 01754
*
*PROGRAM BY EDWARD C. BADGER
*
*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
```

```
$TM=1
; THIS VERSION LAST EDITED - OCT. 6, 1976
```

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

```
.SBTTL TRAP CATCHER
.=0
*ALL UNUSED LOCATIONS FROM 4-776 CONTAIN A ".+2"
*AND "JSR PC,R0" SEQUENCE TO CATCH ILLEGAL INTERRUPTS,
*AND INTERRUPTS TO THE WRONG VECTOR.
*LOCATION 0 CONTAINS A 0 TO CATCH IMPROPERLY LOADED
*VECTORS
.=4
.WORD IOTRD,200 ;HANDLE BUSS ERROR.
.=174
DISPREG: .WORD 0 ;; SOFTWARE DISPLAY REGISTER.
SWREG: .WORD 0 ;; SOFTWARE SWITCH REGISTER.
.=100
.WORD 104,200,2 ; IF "B EVENT" ON 0-BUS IS
; CONNECTED, WE NEED A WAY OF
; IGNORING ITS INTERRUPTS.
.=200
JMP START
```

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
101
102
103
104
105
106
107
108

.SBTTL BASIC DEFINITIONS

;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***

001100

STACK= 1100

.EQUIV EMT,ERROR

::: BASIC DEFINITION OF ERROR CALL

.EQUIV IOT,SCOPE

::: BASIC DEFINITION OF SCOPE CALL

;*MISCELLANEOUS DEFINITIONS

000011

HT= 11

::: CODE FOR HORIZONTAL TAB

000012

LF= 12

::: CODE FOR LINE FEED

000015

CR= 15

::: CODE FOR CARRIAGE RETURN

000200

CRLF= 200

::: CODE FOR CARRIAGE RETURN-LINE FEED

177776

PS= 177776

::: PROCESSOR STATUS WORD

177774

.EQUIV PS,PSW

177774

STKLMT= 177774

::: STACK LIMIT REGISTER

177772

PIRQ= 177772

::: PROGRAM INTERRUPT REQUEST REGISTER

177570

DSWR= 177570

::: HARDWARE SWITCH REGISTER

177570

DDISP= 177570

::: HARDWARE DISPLAY REGISTER

;*GENERAL PURPOSE REGISTER DEFINITIONS

000000

R0= %0

::: GENERAL REGISTER

000001

R1= %1

::: GENERAL REGISTER

000002

R2= %2

::: GENERAL REGISTER

000003

R3= %3

::: GENERAL REGISTER

000004

R4= %4

::: GENERAL REGISTER

000005

R5= %5

::: GENERAL REGISTER

000006

R6= %6

::: GENERAL REGISTER

000007

R7= %7

::: GENERAL REGISTER

000006

SP= %6

::: STACK POINTER

000007

PC= %7

::: PROGRAM COUNTER

;*PRIORITY LEVEL DEFINITIONS

000000

PR0= 0

::: PRIORITY LEVEL 0

000040

PR1= 40

::: PRIORITY LEVEL 1

000100

PR2= 100

::: PRIORITY LEVEL 2

000140

PR3= 140

::: PRIORITY LEVEL 3

000200

PR4= 200

::: PRIORITY LEVEL 4

000240

PR5= 240

::: PRIORITY LEVEL 5

000300

PR6= 300

::: PRIORITY LEVEL 6

000340

PR7= 340

::: PRIORITY LEVEL 7

;*SWITCH REGISTER SWITCH DEFINITIONS

100000

SW15= 100000

040000

SW14= 40000

020000

SW13= 20000

010000

SW12= 10000

004000

SW11= 4000

002000

SW10= 2000

001000

SW09= 1000

000400

SW08= 400

000200

SW07= 200

000100

SW06= 100

000040

SW05= 40

109 000020
 110 000010
 111 000004
 112 000002
 113 000001
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126 100000
 127 040000
 128 020000
 129 010000
 130 004000
 131 002000
 132 001000
 133 000400
 134 000200
 135 000100
 136 000040
 137 000020
 138 000010
 139 000004
 140 000002
 141 000001
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154 000004
 155 000010
 156 000014
 157 000014
 158 000014
 159 000020
 160 000024
 161 000030
 162 000034

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

.*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

```

163          000060          TKVEC= 60          ;;TTY KEYBOARD VECTOR
164          000064          TPVEC= 64          ;;TTY PRINTER VECTOR
165          000240          PIRQVEC=240        ;;PROGRAM INTERRUPT REQUEST VECTOR
166
167          160150          ABASE= 160150
168          000640          AVECT1= 640
169          000200          APRIOR= 200
170          000001          $TN=1
171
172          .SBTTL ACT11 HOOKS
173
174          ;;*****
175          ;;HOOKS REQUIRED BY ACT11
176          000204          $SVPC=.          ;SAVE PC
177          000046          .=46
178          000046          SENDAD          ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
179          000052          .=52
180          000052          .WORD 0          ;;2)SET LOC.52 TO ZERO
181          000204          .=$SVPC          ;; RESTORE PC
182
183          001000          .=1000
184
185          .SBTTL APT PARAMETER BLOCK
186
187          ;;*****
188          ;;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
189          ;;*****
190          001000          .SX=.          ;;SAVE CURRENT LOCATION
191          000024          .=24          ;;SET POWER FAIL TO POINT TO START OF PROGRAM
192          000024          200          ;;FOR APT START UP
193          000044          .=44          ;;POINT TO APT INDIRECT ADDRESS PNTR.
194          000044          $APTHDR          ;;POINT TO APT HEADER BLOCK
195          001000          .=$X          ;;RESET LOCATION COUNTER
196
197          ;;*****
198          ;;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
199          ;;INTERFACE SPEC.
200
201          001000          $APTHD:
202          001000          $SHIBTS: .WORD 0          ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
203          001002          $MADR: .WORD $MAIL          ;; ADDRESS OF APT MAILBOX (BITS 0-15)
204          001004          $TSTM: .WORD 60          ;; RUN TIM OF LONGEST TEST
205          001006          $PASTM: .WORD 120.          ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
206          001010          $UNITH: .WORD 120.          ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
207          001012          .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

```

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
256
257
258
259
260
261

001100
000000
000
000
000000
000000
000000
000000
000000
000000
000
001
000000
000000
000000
000000
000000
000000
000000
000000
000000
000000
000
000
000000
177570
177570
177560
177562
177564
177566
000
002
012
000
000000
000000
177607
077
015
000012

000377

.SBTTL COMMON TAGS

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

SCMTAG: .=1100 ;: START OF COMMON TAGS
STSTNM: .WORD 0 ;: CONTAINS THE TEST NUMBER
SERFLG: .BYTE 00 ;: CONTAINS ERROR FLAG
SICNT: .WORD 00 ;: CONTAINS SUBTEST ITERATION COUNT
SLPADR: .WORD 00 ;: CONTAINS SCOPE LOOP ADDRESS
SLPERR: .WORD 00 ;: CONTAINS SCOPE RETURN FOR ERRORS
SERTIL: .WORD 00 ;: CONTAINS TOTAL ERRORS DETECTED
SITEMB: .BYTE 00 ;: CONTAINS ITEM CONTROL BYTE
SERMAX: .BYTE 1 ;: CONTAINS MAX. ERRORS PER TEST
SERAPC: .WORD 0 ;: CONTAINS PC OF LAST ERROR INSTRUCTION
SGDADR: .WORD 0 ;: CONTAINS ADDRESS OF 'GOOD' DATA
SBDADR: .WORD 0 ;: CONTAINS ADDRESS OF 'BAD' DATA
SGDDAT: .WORD 0 ;: CONTAINS 'GOOD' DATA
SBDAT: .WORD 0 ;: CONTAINS 'BAD' DATA
SAUTOB: .WORD 0 ;: RESERVED--NOT TO BE USED
SINTAG: .BYTE 0 ;: AUTOMATIC MODE INDICATOR
SWR: .WORD 0 ;: INTERRUPT MODE INDICATOR
DISPLAY: .WORD DSWR ;: ADDRESS OF SWITCH REGISTER
STKS: 177560 ;: ADDRESS OF DISPLAY REGISTER
STKB: 177562 ;: TTY KBD STATUS
STPS: 177564 ;: TTY KBD BUFFER
STPB: 177566 ;: TTY PRINTER STATUS REG. ADDRESS
SNUL: .BYTE 0 ;: TTY PRINTER BUFFER REG. ADDRESS
SFILLS: .BYTE 2 ;: CONTAINS NULL CHARACTER FOR FILLS
SFILLC: .BYTE 12 ;: CONTAINS # OF FILLER CHARACTERS REQUIRED
STPFLG: .BYTE 0 ;: INSERT FILL CHARS. AFTER A "LINE FEED"
STIMES: 0 ;: "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
SESCAPE: 0 ;: MAX. NUMBER OF ITERATIONS
SBELL: .ASCIZ <207><377><377> ;: ESCAPE ON ERROR ADDRESS
SQUES: .ASCII /?/ ;: CODE FOR BELL
SCRLF: .ASCII <15> ;: QUESTION MARK
SLF: .ASCIZ <12> ;: CARRIAGE RETURN
 ;: LINE FEED

.SBTTL APT MAILBOX-ETABLE

EVEN
SMAIL: ;: APT MAILBOX
SMSGTY: .WORD AMSGTY ;: MESSAGE TYPE CODE
SFATAL: .WORD AFATAL ;: FATAL ERROR NUMBER
STESTN: .WORD ATESTN ;: TEST NUMBER
SPASS: .WORD APASS ;: PASS COUNT
SDEVCT: .WORD ADEVCT ;: DEVICE COUNT

| | | | | | | |
|-----|--------|--------|-----------|-------|--------|--|
| 262 | 001206 | 000000 | \$UNIT: | .WORD | AUNIT | :: I/O UNIT NUMBER |
| 263 | 001210 | 000000 | \$MSGAD: | .WORD | AMSGAD | :: MESSAGE ADDRESS |
| 264 | 001212 | 000000 | \$MSGLG: | .WORD | AMSGLG | :: MESSAGE LENGTH |
| 265 | 001214 | | \$ETABLE: | | | :: APT ENVIRONMENT TABLE |
| 266 | 001214 | 000 | \$ENV: | .BYTE | AENV | :: ENVIRONMENT BYTE |
| 267 | 001215 | 000 | \$ENVM: | .BYTE | AENVM | :: ENVIRONMENT MODE BITS |
| 268 | 001216 | 000000 | \$SWREG: | .WORD | ASWREG | :: APT SWITCH REGISTER |
| 269 | 001220 | 000000 | \$USWR: | .WORD | AUSWR | :: USER SWITCHES |
| 270 | 001222 | 000000 | \$CPUOP: | .WORD | ACPUOP | :: CPU TYPE, OPTIONS |
| 271 | | | * | | | BITS 15-11=CPU TYPE |
| 272 | | | * | | | 11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05 |
| 273 | | | * | | | 11/70=06, P00=07, Q=10 |
| 274 | | | * | | | BIT 10=REAL TIME CLOCK |
| 275 | | | * | | | BIT 9=FLOATING POINT PROCESSOR |
| 276 | | | * | | | BIT 8=MEMORY MANAGEMENT |
| 277 | 001224 | 000 | \$MAMS1: | .BYTE | AMAMS1 | :: HIGH ADDRESS, M.S. BYTE |
| 278 | 001225 | 000 | \$MTYP1: | .BYTE | AMTYP1 | :: MEM. TYPE, BLK#1 |
| 279 | | | * | | | MEM. TYPE BYTE -- (HIGH BYTE) |
| 280 | | | * | | | 900 NSEC CORE=001 |
| 281 | | | * | | | 300 NSEC BIPOLAR=002 |
| 282 | | | * | | | 500 NSEC MOS=003 |
| 283 | 001226 | 000000 | \$MADR1: | .WORD | AMADR1 | :: HIGH ADDRESS, BLK#1 |
| 284 | | | * | | | MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE |
| 285 | 001230 | 000 | \$MAMS2: | .BYTE | AMAMS2 | :: HIGH ADDRESS, M.S. BYTE |
| 286 | 001231 | 000 | \$MTYP2: | .BYTE | AMTYP2 | :: MEM. TYPE, BLK#2 |
| 287 | 001232 | 000000 | \$MADR2: | .WORD | AMADR2 | :: MEM. LAST ADDRESS, BLK#2 |
| 288 | 001234 | 000 | \$MAMS3: | .BYTE | AMAMS3 | :: HIGH ADDRESS, M.S. BYTE |
| 289 | 001235 | 000 | \$MTYP3: | .BYTE | AMTYP3 | :: MEM. TYPE, BLK#3 |
| 290 | 001236 | 000000 | \$MADR3: | .WORD | AMADR3 | :: MEM. LAST ADDRESS, BLK#3 |
| 291 | 001240 | 000 | \$MAMS4: | .BYTE | AMAMS4 | :: HIGH ADDRESS, M.S. BYTE |
| 292 | 001241 | 000 | \$MTYP4: | .BYTE | AMTYP4 | :: MEM. TYPE, BLK#4 |
| 293 | 001242 | 000000 | \$MADR4: | .WORD | AMADR4 | :: MEM. LAST ADDRESS, BLK#4 |
| 294 | 001244 | 000640 | \$VECT1: | .WORD | AVECT1 | :: INTERRUPT VECTOR#1, BUS PRIORITY#1 |
| 295 | 001246 | 000000 | \$VECT2: | .WORD | AVECT2 | :: INTERRUPT VECTOR#2, BUS PRIORITY#2 |
| 296 | 001250 | 160150 | \$BASE: | .WORD | ABASE | :: BASE ADDRESS OF EQUIPMENT UNDER TEST |
| 297 | 001252 | 000000 | \$DEVN: | .WORD | ADEVN | :: DEVICE MAP |
| 298 | 001254 | 000000 | \$CDW1: | .WORD | ACDW1 | :: CONTROLLER DESCRIPTION WORD#1 |
| 299 | 001256 | | \$ETEND: | | | |
| 300 | | | .MEXIT | | | |

301
302
303
304
305
306
307
308
309
310
311
312
313
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

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

;ITEM 1

EM1 ;IBS FUNCTION ERROR
DH1 ;TEST ERRPC IB ADDR
DT1 ;\$TESTN,\$ERRPC,IBS
DF0 ;ALL NUMBERS ARE IN OCTAL FORM.

;ITEM 2

EM2 ;IBD FUNCTION ERROR
DH1 ;TEST ERRPC IB ADDR
DT1 ;\$TESTN,\$ERRPC,IBS
DF0 ;ALL NUMBERS ARE IN OCTAL FORM.

;ITEM 3

EM3 ;IBS DATA ERROR
DH3 ;TEST ERRPC GOOD BAD
DT3 ;\$TESTN,\$ERRPC,\$GDDAT,\$BDDAT
DF0 ;ALL NUMBERS ARE IN OCTAL FORM.

;ITEM 4

EM4 ;IBD DATA ERROR
DH3 ;TEST ERRPC GOOD BAD
DT3 ;\$TESTN,\$ERRPC,\$GDDAT,\$BDDAT
DF0 ;ALL NUMBERS ARE IN OCTAL FORM.

;ITEM 5

EM5 ;IBS/IBD ADDRESS ERROR
DH5 ;TEST ERROR PC ADDRESS
DT5 ;\$TESTN,\$ERRPC,IBS
DF0 ;ALL NUMBERS ARE IN OCTAL FORM.

;ITEM 6

001256

001256 012216
001260 012437
001262 012640
001264 012710

001266 012244
001270 012437
001272 012640
001274 012710

001276 012272
001300 012505
001302 012654
001304 012710

001306 012314
001310 012505
001312 012654
001314 012710

001316 012336
001320 012542
001322 012666
001324 012710

355 001326 012366
356 001330 012505
357 001332 012654
358 001334 012710
359
360

EM6
DH3
DT3
DFO

; IBC/IBCA DATA ERROR
; TEST ERRPC GOOD BAD
; \$TESTN, \$ERRPC, \$GOODAT, \$BADAT
; ALL NUMBERS ARE IN OCTAL FORM.

361 001336 012415
362 001340 012573
363 001342 012676
364 001344 012710
365
366
367
368
369
370
371

; ITEM 7
EM7
DH7
DT7
DFO

; INTERRUPT ERROR
; TEST ERRPC TO FROM ADDR.
; \$TSTNM, \$ERRPC, \$TRTO, \$TRFRO
; ALL NUMBERS ARE IN OCTAL FORM.

372
373
374
375

.SBTTL REG ADDRESS AND COMMON TAGS
; WARNING IF DEVICE # IS AT DIFFERENT ADDRESS OR VECTOR
; DO NOT PATCH THESE LOCATIONS - SEE PROGRAM DOCUMENTATION.

376 001346 160150
377 001350 160152
378 001352 160154
379 001354 160156
380 001356 000640
381 001360 000644
382 001362 000650
383 001364 000654
384 001366 160160
385 001370 160162
386 001372 000660
387 001374 000664
388 001376 000670
389 001400 000674
390
391
392

IBS: .WORD ABASE ;>NO ;< CONTROL AND STATUS REGISTER.
IBD: .WORD ABASE+2 ;>PATCHES ;< DATA REGISTER.
IBWC: .WORD ABASE+4 ;< ADDRESS RESERVED FOR
IBCA: .WORD ABASE+6 ;< FUTURE USE
VECTA: .WORD AVECT1 ;>ALLOWED ;< VECTOR ADDRESS.
VECTB: .WORD AVECT1+4 ;>HERE! ;< VECTOR ADDR. +4.
VECTC: .WORD AVECT1+10
VECTD: .WORD AVECT1+14
IBS2: .WORD ABASE+10
IBD2: .WORD ABASE+12
VECTA2: .WORD AVECT1+20
VECTB2: .WORD AVECT1+24
VECTC2: .WORD AVECT1+30
VECTD2: .WORD AVECT1+34

; VECTOR ADDRESSES +2 LOCATIONS.

393 001402 000642
394 001404 000646
395 001406 000652
396 001410 000656
397

PRA: .WORD AVECT1+2 ; NOTE: DO NOT ATTEMPT TO PATCH
PRB: .WORD AVECT1+6 ; THESE LOCATIONS IF A VECTOR
PRC: .WORD AVECT1+12 ; VARYIES - ALTER LOCATION
PRD: .WORD AVECT1+16 ; "\$VECT1:".

398 001412 000662
399 001414 000666
400 001416 000672
401 001420 000676
402
403
404

PRA2: .WORD AVECT1+22 ; IF TEST MODULE VECTOR IS
PRB2: .WORD AVECT1+26 ; DIFFERENT, YOU MUST CHANGE
PRC2: .WORD AVECT1+32 ; LOCATION "\$VECTA2:".
PRD2: .WORD AVECT1+36 ;

.SBTTL PROGRAM START

405
406
407 001422
408

START:
.SBTTL INITIALIZE THE COMMON TAGS

```

409          ;; CLEAR THE COMMON TAGS ($CMTAG) AREA
410 001422 012706 001100      MOV    $CMTAG,R6      ;; FIRST LOCATION TO BE CLEARED
411 001426 005026             CLR    (R6)+          ;; CLEAR MEMORY LOCATION
412 001430 022706 001140      CMP    $SWR,R6      ;; DONE?
413 001434 001374             BNE    -6             ;; LOOP BACK IF NO
414 001436 012706 001100      MOV    $STACK,SP    ;; SETUP THE STACK POINTER
415          ;; INITIALIZE A FEW VECTORS
416 001442 012737 007450 000020  MOV    $SCOPE,$IOTVEC ;; IOT VECTOR FOR SCOPE ROUTINE
417 001450 012737 000340 000022  MOV    $340,$IOTVEC+2 ;; LEVEL 7
418 001456 012737 007126 000030  MOV    $,$ERRVEC    ;; ERR VECTOR FOR ERROR ROUTINE
419 001464 012737 000340 000032  MOV    $340,$ERRVEC+2 ;; LEVEL 7
420 001472 012737 012136 000034  MOV    $,$TRAPVEC   ;; TRAP VECTOR FOR TRAP CALLS
421 001500 012737 000340 000036  MOV    $340,$TRAPVEC+2 ;; LEVEL 7
422 001506 012737 011710 000024  MOV    $SPWRON,$PWRVEC ;; POWER FAILURE VECTOR
423 001514 012737 000340 000026  MOV    $340,$PWRVEC+2 ;; LEVEL 7
424 001522 005037 001160             CLR    $TIMES       ;; INITIALIZE NUMBER OF ITERATIONS
425 001526 005037 001162             CLR    $ESCAPE      ;; CLEAR THE ESCAPE ON ERROR ADDRESS
426 001532 112737 000001 001115  MOV    $1,$ERRMAX   ;; ALLOW ONE ERROR PER TEST
427 001540 012737 001540 001106  MOV    $,$SLPADR    ;; INITIALIZE THE LOOP ADDRESS FOR SCOPE
428 001546 012737 001546 001110  MOV    $,$SLPERR    ;; SETUP THE ERROR LOOP ADDRESS
429          ;; SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
430          ;; EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
431 001554 013746 000004             MOV    $ERRVEC,-(SP) ;; SAVE ERROR VECTOR
432 001560 012737 001614 000004  MOV    $64,$ERRVEC  ;; SET UP ERROR VECTOR
433 001566 012737 177570 001140  MOV    $OSWR,$SWR   ;; SETUP FOR A HARDWARE SWICH REGISTER
434 001574 012737 177570 001142  MOV    $ODISP,$DISPLAY ;; AND A HARDWARE DISPLAY REGISTER
435 001602 022777 177777 177330  CMP    #-1,$SWR    ;; TRY TO REFERENCE HARDWARE SWR
436 001610 001012             BNE    66$         ;; BRANCH IF NO TIMEOUT TRAP OCCURRED
437          ;; AND THE HARDWARE SWR IS NOT = -1
438 001612 000403             BR     65$        ;; BRANCH IF NO TIMEOUT
439 001614 012716 001622 64$: MOV    $65$,(SP)    ;; SET UP FOR TRAP RETURN
440 001620 000002             RTI
441 001622 012737 000176 001140 65$: MOV    $SWREG,$SWR  ;; POINT TO SOFTWARE SWR
442 001630 012737 000174 001142  MOV    $DISPREG,$DISPLAY
443 001636 012637 000004 66$: MOV    (SP)+,$ERRVEC ;; RESTORE ERROR VECTOR
444          ;;
445 001642 005037 001202             CLR    $PASS        ;; CLEAR PASS COUNT
446 001646 132737 000200 001215  BITB  $APTSIZE,$ENVM ;; TEST USER SIZE UNDER APT
447 001654 001403             BEQ    67$         ;; YES, USE NON-APT SWITCH
448 001656 012737 001216 001140 67$: MOV    $SSWREG,$SWR ;; NO, USE APT SWITCH REGISTER
449 001664             ;;
450 001664 012737 012066 000004  MOV    $IOTRD,$ERRVEC ;; SET TO HANDLE BUS ERRORS.
451 001672 012737 000200 000006  MOV    $200,$ERRVEC+2
452          ;;
453 001700 013737 001250 001346  MOV    $BASE,$IBS  ;; GET BASE ADDR.
454 001706 013737 001346 001350  MOV    $IBS,$IB0   ;; FIX DATA BUFFER=
455 001714 062737 000002 001350  ADD    $2,$IB0     ;; CSR+2
456 001722 013737 001350 001352  MOV    $IB0,$IBWC
457 001730 062737 000002 001352  ADD    $2,$IBWC
458 001736 013737 001352 001354  MOV    $IBWC,$IBCA
459 001744 062737 000002 001354  ADD    $2,$IBCA
460 001752 013737 001244 001356  MOV    $VECT1,$VECTA ;; GET VECTOR ADDR.
461 001760 042737 170000 001356  BIC    $170000,$VECTA ;; STRIP JUNK
462 001766 013737 001346 012646  MOV    $IBS,$IBSA

```

```

463 001774 013737 001350 012650      MOV      I80,I80A
464 002002 013737 001366 001370      MOV      I85,I802
465 002010 062737 000002 001370      ADD      #2,I802
466 002016 013737 001356 001360      MOV      VECTA,VECTB
467 002024 062737 000004 001360      ADD      #4,VECTB
468 002032 013737 001360 001362      MOV      VECTB,VECTC
469 002040 062737 000004 001362      ADD      #4,VECTC
470 002046 013737 001362 001364      MOV      VECTC,VECTD
471 002054 062737 000004 001364      ADD      #4,VECTD
472 002062 013737 001372 001374      MOV      VECTA2,VECTB2
473 002070 062737 000004 001374      ADD      #4,VECTB2
474 002076 013737 001374 001376      MOV      VECTB2,VECTC2
475 002104 062737 000004 001376      ADD      #4,VECTC2
476 002112 013737 001376 001400      MOV      VECTC2,VECTD2
477 002120 062737 000004 001400      ADD      #4,VECTD2
478
479 002126 013737 001356 001402      MOV      VECTA,PRA          ;SET UP VECTOR+2 ADDRESSES.
480 002134 062737 000002 001402      ADD      #2,PRA
481 002142 013737 001402 001404      MOV      PRA,PRB
482 002150 062737 000004 001404      ADD      #4,PRB
483 002156 013737 001404 001406      MOV      PRB,PRC
484 002164 062737 000004 001406      ADD      #4,PRC
485 002172 013737 001406 001410      MOV      PRC,PRD
486 002200 062737 000004 001410      ADD      #4,PRD
487 002206 013737 001372 001412      MOV      VECTA2,PRA2
488 002214 062737 000002 001412      ADD      #2,PRA2
489 002222 013737 001412 001414      MOV      PRA2,PRB2
490 002230 062737 000004 001414      ADD      #4,PRB2
491 002236 013737 001414 001416      MOV      PRB2,PRC2
492 002244 062737 000004 001416      ADD      #4,PRC2
493 002252 013737 001416 001420      MOV      PRC2,PRD2
494 002260 062737 000004 001420      ADD      #4,PRD2
495 002266
496
497 002266 013777 001402 177062      MOV      PRA,@VECTA ;/RESTORE VECTOR FOR
498 002274 012777 004700 177100      MOV      #4700,@PRA ;/ILLEGAL INTRO.
499
500 002302 013777 001404 177050      MOV      PRB,@VECTB ;/RESTORE VECTOR FOR
501 002310 012777 004700 177066      MOV      #4700,@PRB ;/ILLEGAL INTRO.
502
503 002316 013777 001406 177036      MOV      PRC,@VECTC ;/RESTORE VECTOR FOR
504 002324 012777 004700 177054      MOV      #4700,@PRC ;/ILLEGAL INTRO.
505
506 002332 013777 001410 177024      MOV      PRD,@VECTD ;/RESTORE VECTOR FOR
507 002340 012777 004700 177042      MOV      #4700,@PRD ;/ILLEGAL INTRO.
508
509
510 002346 005227 177777      .SBTTL TYPE PROGRAM NAME
511 002352 001033      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
512 002354 104401 002422      INC #1 ;:FIRST TIME?
513
514 002360 005737 000042      BNE 645 ;:BRANCH IF NO
515 002364 001012      TYPE ,655 ;:TYPE ASCIZ STRING
516 002366 123727 001214 000001      .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
TST @#42 ;:ARE WE RUNNING UNDER XXDP/ACT?
BNE 665 ;:BRANCH IF YES
CMPB $ENV,#1 ;:ARE WE RUNNING UNDER APT?

```

```

517 002374 001406          BEQ      66$          ;; BRANCH IF YES
518 002376 023727 001140 000176  CMP      SWR, #SWREG ;; SOFTWARE SWITCH REG SELECTED?
519 002404 001005          BNE      67$          ;; BRANCH IF NO
520 002406 104406          GTSWR                    ;; GET SOFT-SWR SETTINGS
521 002410 000403          BR      67$
522 002412 112737 000001 001134 66$:  MOVB    #1, SAUTOB    ;; SET AUTO-MODE INDICATOR
523 002420 000403          BR      67$
524 002420 000410          BR      64$          ;; GET OVER THE ASCIZ
525                                     ;; 65$: .ASCIZ <CRLF>#MD11-DVIBA-A#<CRLF>
526 002442                                     64$:
527 002442 000005          RESET
528
529                                     ;; *****
530                                     ;; *TEST 1 *TEST THE ADDRESSABILITY OF THE IBS, IBD REGISTERS
531                                     ;; *****
532 002444 000240          †ST1:  NOP
533 002446 012737 000050 001160  MOV     #50, $TIMES    ;; DO 50 ITERATIONS
534 002454 012737 002504 001106  MOV     #1$, $SLPADR   ;; SET SCOPE LOOP ADDRESS
535
536 002462 012737 000001 001102  MOV     #1, $STSTM     ;; SET TEST #1.
537 002470 012737 000001 001200  MOV     #1, $TESTN     ;; DON'T FORGET APT!
538 002476 012737 002504 001110  MOV     #1$, $SLPERR
539
540 002504 013746 000004          1$:  MOV     ERRVEC, -(SP)  ;; SAVE CONTENTS OF ADDR. 4
541 002510 012737 002536 000004  MOV     #2$, ERRVEC   ;; SET TIME-OUT TRAP VECTOR TO HANDLE
542                                     ;; IN CASE WE TIME OUT WHEN
543                                     ;; WE ADDR. THE IBV-11
544
545 002516 005777 176624          TST     @IBS          ;; ADDR THE IBS, IF NO RESPONSE,
546                                     ;; WILL TRAP TO 2$ FROM HERE
547 002522 012737 002544 000004  MOV     #3$, ERRVEC   ;; CHANGE FOR ADDRESSING THE IBD REG.
548
549 002530 005777 176614          TST     @IBD          ;; ADDR THE IBD REG.
550                                     ;; WE'LL TRAP TO 3$ FROM HERE IF BAD.
551 002534 000406          BR      4$
552 002536                                     2$:
553 002536 062706 000004          ADD     #4, SP        ; /ADD #4 TO STACK POINTER.
554
                                     ;; *****
                                     ;; *****
558                                     ERROR 5
559 002542 104005          ;; /MODULE FAULT DETECTED:
560                                     ;; IBS REGISTER COULD NOT BE
561                                     ;; ADDRESSED
562
563                                     ;; *****
564                                     ;; *****
564 002544                                     3$:
565 002544 062706 000004          ADD     #4, SP        ; /ADD #4 TO STACK POINTER.
566
                                     ;; *****
                                     ;; *****

```

571 002550 104005
572

ERROR 5 ;/MODULE FAULT DETECTED:
;ADDRESSED

575 002552 012637 000004
576
577 002556 012746 000000
578 002562 012746 002570
579 002566 000002
580 002570
581
582
583
584
585
586
587
588
589
590

4S: ;:SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
MOV (SP)+,ERRVEC ;RESTORE CONTENTS OF LOC 4.
;PR
MOV #0,-(SP) ;/SET CPU PRIORITY ON RETURN
MOV #64S,-(SP) ;/SHOW RETURN ADDRESS
RTI ;/CAUSE A RETURN (PUTS NEW STATUS
;IN STATUS REG.)
64S:

;TEST 2 *TEST THAT BASE ADDRESSES +4,+6 RESPOND WHEN ADDRESSED

;*EVEN THOUGH THE BASE ADDRESS +4 AND +6 ARE NOT USED,
;*THE IBV11A SHOULD RESPOND TO THEM
;*

591 002570 000004
592 002572 012737 000010 001160
593
594 002600 013746 000004
595 002604 012737 002632 000004
596
597
598 002612 005777 176534
599
600
601 002616 012737 002642 000004
602
603 002624 005777 176524
604 002630 000407
605
606 002632
607 002632 062706 000004
608

↑ST2: SCOPE
MOV #10,STIMES ;;DO 10 ITERATIONS
MOV ERRVEC,-(SP) ;SAVE CONTENTS OF ADDR 4.
MOV #1S,ERRVEC ;SET TIME OUT TRAP VECTOR TO HANDLE
;IN CASE WE TIME OUT WHEN WE
;ADDRESS THE IBV-11 ADDRESSES +4,+6.
TST @IBWC ;TEST BASE ADDRESS +4, IF NO RESPONSE
;WILL TRAP TO 1S FROM HERE
MOV #2S,ERRVEC ;CHANGE FOR ADDRESSING +6 ADDR.
TST @IBCA ;ADDR THE +6 ADDR. - TRAP IF BAD.
BR 3S ;CONTINUE IF GOOD.
1S: ADD #4,SP ;/ADD #4 TO STACK POINTER.

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

612
613 002636 104005
614
615

ERROR 5 ;/MODULE FAULT DETECTED:
;BASE ADDR+4 COULD NOT
;BE ADDRESSED.

618 002640 000403
619
620 002642
621 002642 062706 000004
622

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
BR 3S
2S: ADD #4,SP ;/ADD #4 TO STACK POINTER.

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

626
627 002646 104005          ERROR 5          ;/MODULE FAULT DETECTED:
628                                     ;BASE ADDR+6 COULD NOT
629                                     ;BE ADDRESSED.

; ;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

632
633 002650 012637 000004    3$:  MOV (SP)+,ERRVEC ;RESTORE CONTENTS OF LOC 4.
634
635 ;:*****
636 ;*TEST 3 *TEST THAT IBS IS CLEAR AT INIT OF TESTING
637 ;:*****
638 002654 000004          †ST3: SCOPE
639 002656 012737 000001 001160  MOV #1,STIMES ;;DO 1 ITERATION
640
641 002664 000005          RESET ;ISSUE SYSTEM INIT.
642
643 002666 005037 001124    CLR $GDDAT ;EXPECT ZERO CSR.
644 002672 017737 176450 001126  MOV @IBS,$BDDAT ;READ CSR.
645 002700 001401          BEQ TST4 ;
646

; ;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

650
651 002702 104003          ERROR 3          ;/MODULE FAULT DETECTED:
652                                     ;IBS NOT CLEAR ON INT.

; ;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

655
656 ;:*****
657 ;*TEST 4 *TEST THAT IBO IS CLEAR AT INIT OF TESTING
658 ;:*****
659 002704 000004          †ST4: SCOPE
660 002706 012737 000001 001160  MOV #1,STIMES ;;DO 1 ITERATION
661
662 002714 000005          RESET ;ISSUE SYSTEM INITIALIZE.
663
664 002716 005037 001124    CLR $GDDAT ;EXPECT ZERO CSR.
665 002722 117737 176422 001126  MOVB @IBO,$BDDAT ;READ DBR
666 002730 001401          BEQ TST5 ;
667

; ;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

671
672 002732 104004          ERROR 4          ;/MODULE FAULT DETECTED:
673                                     ;IBO NOT CLEAR ON INIT.

; ;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

676
677 ;:*****
678 ;:*****

```

MAINDEC-11-DVIBA-A
DVIBA.P11 TS

MACY11 27(663) 29-MAR-77 12:57 PAGE 14
*TEST THAT BASE ADDRESSES +4,+6 RETURN ZERO WHEN READ

SEQ 0029

| | | | | |
|-----|--------|--------|--------|--------|
| 679 | | | | |
| 680 | | | | |
| 681 | | | | |
| 682 | | | | |
| 683 | | | | |
| 684 | | | | |
| 685 | 002734 | 000004 | | |
| 686 | 002736 | 012737 | 000010 | 001160 |
| 687 | | | | |
| 688 | 002744 | 005037 | 001124 | |
| 689 | 002750 | 017737 | 176376 | 001126 |
| 690 | 002756 | 001402 | | |
| 691 | | | | |

; *TEST 5 *TEST THAT BASE ADDRESSES +4,+6 RETURN ZERO WHEN READ

; *
; *BASE ADDRESS +4 AND +6 SHOULD RETURN A ZERO WHEN
; *READ, IN THIS TEST WE WILL TRY THAT.
; *

```

↑ST5: SCOPE
      MOV     #10,STIMES      ;;DO 10 ITERATIONS
      CLR     $GDOAT         ;EXPECT ZERO RETURN
      MOV     @IBWC,$BDOAT   ;READ BASE ADDRESS+4
      BEQ     IS            ;IF ZERO - GOOD.

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

| | | | | |
|-----|--------|--------|--|--|
| 695 | | | | |
| 696 | 002760 | 104006 | | |
| 697 | | | | |
| 698 | | | | |

ERROR 6 ;/MODULE FAULT DETECTED:
;SHOULD HAVE READ BACK ZERO FROM
;THIS ADDR.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

| | | | | |
|-----|--------|--------|--------|--------|
| 701 | 002762 | 000405 | | |
| 702 | | | | |
| 703 | 002764 | 017737 | 176364 | 001126 |
| 704 | 002772 | 001401 | | |
| 705 | | | | |

```

BR     TST6                ;;
↑ST6: MOV     @IBCA,$BDOAT  ;READ BASE ADDR+6, SHOULD BE ZERO
      BEQ     TST6                ;;

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

| | | | | |
|-----|--------|--------|--|--|
| 709 | | | | |
| 710 | 002774 | 104006 | | |
| 711 | | | | |
| 712 | | | | |

ERROR 6 ;/MODULE FAULT DETECTED:
;SHOULD HAVE READ BACK ZERO FROM
;BASE ADDR+6.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

| | | | | |
|-----|--------|--------|--------|--------|
| 715 | | | | |
| 716 | | | | |
| 717 | | | | |
| 718 | | | | |
| 719 | 002776 | 000004 | | |
| 720 | | | | |
| 721 | 003000 | 005077 | 176342 | |
| 722 | 003004 | 052777 | 000001 | 176334 |
| 723 | 003012 | 052737 | 002001 | 001124 |
| 724 | 003020 | 017737 | 176322 | 001126 |
| 725 | 003026 | 023737 | 001124 | 001126 |
| 726 | 003034 | 000402 | | |
| 727 | | | | |

```

*****
; *TEST 6 *TEST THAT WE CAN SET TCS, TCS SETS CMD
*****
↑ST6: SCOPE

```

```

      CLR     @IBS          ;CLEAR CLR
      BIS     @BIT0,@IBS   ;SET TCS.
      BIS     @BIT0!BIT10,$GDOAT ;EXPECT ONLY TCS AND CMD TO SET
      MOV     @IBS,$BDOAT  ;READ THE IBS.
      CMP     $GDOAT,$BDOAT ;DID TCS AND CMD SET?
      BR     IS            ;YES - CONTINUE

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

| | | | | |
|-----|--------|--------|--|--|
| 731 | | | | |
| 732 | 003036 | 104003 | | |

ERROR 3 ;/MODULE FAULT DETECTED:

733 ;TCS AND/OR CMD FAILED TO SET

```

736 003040 000412      ;:SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
737      BR      TST7      ;;
738 003042 042777 000001 176276 1S: BIC      @BIT0,@IBS      ;CLEAR TCS.
739 003050 005037 001124      CLR      $GDDAT      ;EXPECT TCS AND CMD TO CLEAR
740 003054 017737 176266 001126 MOV      @IBS,$BDDAT ;READ IBS, DID THEY CLEAR?
741 003062 001401      BEQ      TST7      ;;
742

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

746 003064 104003      ERROR 3      ;/MODULE FAULT DETECTED:
747      ;TCS AND/OR CMD FAILED TO CLEAR.
748

```

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

```

751
752      ;:*****
753      ;*TEST 7      *TEST THAT EOP WILL SET
754      ;:*****
755 003066 000004      †TST7: SCOPE
756
757 003070 J05077 176252      CLR      @IBS      ;CLEAR CSR.
758 003074 052777 000002 176244 BIS      @BIT1,@IBS ;SET EOP
759 003102 012737 000002 001124 MOV      @BIT1,$GDDAT ;EXPECT ONLY EOP TO SET.
760 003110 017737 176232 001126 MOV      @IBS,$BDDAT ;READ IBS
761 003116 023737 001124 001126 CMP      $GDDAT,$BDDAT ;DID EOP SET?
762 003124 001402      BEQ      1S      ;YES - CONTINUE
763

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

767 003126 104003      ERROR 3      ;/MODULE FAULT DETECTED:
768      ;EOP BIT SETTING ERROR.
769

```

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

```

772 003130 000412      BR      TST10      ;;
773
774 003132 042777 000002 176206 1S: BIC      @BIT1,@IBS      ;CLEAR EOP
775 003140 005037 001124      CLR      $GDDAT      ;EXPECT A ZERO CSR.
776 003144 017737 176176 001126 MOV      @IBS,$BDDAT ;READ IBS, IS IT CLEAR?
777 003152 001401      BEQ      TST10      ;;
778

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

782 003154 104003      ERROR 3      ;/MODULE FAULT DETECTED:
783      ;IBS FAILED TO CLEAR
784

```

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

787
788
789
790
791
792
793
794
795
796
797
798
799

003156 000004
003160 005077 176162
003164 052777 000004 176154
003172 012737 000004 001124
003200 017737 176142 001126
003206 023737 001126 001124
003214 001402

```

*****
*TEST 10      *TEST THAT RE WILL SET + CLEAR
*****
TST10: SCOPE
CLR      QIBS      ;CLEAR CSR.
BIS      #BIT02,QIBS ;SET REM
MOV      #BIT02,$GDDAT ;EXPECT ONLY REM TO SET.
MOV      QIBS,$BDDAT ;READ IBS.
CMP      $BDDAT,$GDDAT ;DID REM AND ONLY REM SET?
BEQ      IS        ;YES - CONTINUE

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

803
804
805

003216 104003

```

ERROR 3 ;/MODULE FAULT DETECTED:
        ;REM BIT SETTING ERROR.

```

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

808
809
810
811
812
813
814

003220 000412
003222 042777 000004 176116 IS:
003230 005037 001124
003234 017737 176106 001126
003242 001401

```

BR      TST11      ;
BIC      #BIT02,QIBS ;CLEAR REM BIT.
CLR      $GDDAT    ;EXPECT ZERO CSR.
MOV      QIBS,$BDDAT ;READ IBS - IS IT CLEAR?
BEQ      TST11      ;

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

818
819
820

003244 104003

```

ERROR 3 ;/MODULE FAULT DETECTED:
        ;IBS FAILED TO CLEAR.

```

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839

003246 000004
003250 005077 176072
003254 052777 000010 176064
003262 012737 000010 001124
003270 017737 176052 001126
003276 023737 001124 001126
003304 001414
003306 012777 000010 176032
003314 017737 176026 001126
003322 023737 001124 001126
003330 001402

```

*****
*TEST 11      *TEST THAT IBC WILL SET AND CLEAR
*****
TST11: SCOPE
CLR      QIBS      ;CLEAR CSR.
BIS      #BIT03,QIBS ;SET IBC
MOV      #BIT03,$GDDAT ;EXPECT ONLY IBC TO BE SET
MOV      QIBS,$BDDAT ;READ IBS.
CMP      $GDDAT,$BDDAT ;DID IBS SET?
BEQ      IS        ;YES CONTINUE
MOV      #BIT03,QIBS ;TRY SETTING IBC AGAIN.
MOV      QIBS,$BDDAT ;MEMORY REFRESH MIGHT HAVE
CMP      $GDDAT,$BDDAT ;GOT IN THE WAY.
BEQ      IS

```

```

843
844 003332 104003 ERROR 3 ;/MODULE FAULT DETECTED:
845 ;IBS BIT SETTING ERROR.

```

```

848 003334 000416 ;:SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
849 BR TST12 ;:
850 003336 004537 007106 15: JSR R5,DEL50 ;DELAY 150 US.
851 003342 000006 .WORD 6
852
853 003344 012737 002001 001124 MOV #BIT10:BIT0,$GDDAT ;EXP CMD AND TCS.
854 003352 017737 175770 001126 MOV @IBS,$BDDAT ;READ IBS - IS IT CLEAR?
855 003350 023737 001124 001126 CMP $GDDAT,$BDDAT
856 003356 001401 BEQ TST12 ;:
857

```

```

861
862 003370 104003 ERROR 3 ;/MODULE FAULT DETECTED:
863 ;IBS NOT CLEAR AFTER IBC

```

```

866 ;:SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
867 ;:*****
868 ;:TEST 12 *TEST THAT TON (BIT05) AND TKR SET AND CLEAR
869 ;:*****
870 TST12: SCOPE
871 003374 005077 175746 CLR @IBS ;CLEAR THE CSR.
872 003400 052777 000040 175740 BIS #BIT5,@IBS ;SET TON.
873 003406 012737 001040 001124 MOV #BIT5:BIT9,$GDDAT ;EXPECT ONLY TON AND TKR TO SET.
874 003414 017737 175726 001126 MOV @IBS,$BDDAT ;READ CSR.
875 003422 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID THEY BOTH SET?
876 003430 001402 BEQ 15 ;YES - CONTINUE.
877
878

```

```

882
883 003432 104003 ERROR 3 ;/MODULE FAULT DETECTED:
884 ;ERROR IN SETTING TON BIT.

```

```

887 003434 000412 ;:SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
888 BR TST13 ;:
889 003436 042777 000040 175702 15: BIC #BIT5,@IBS ;WHEN TON CLEARED, TKR SHOULD CLEAR.
890 003444 005037 001124 CLR $GDDAT ;EXPECT ZERO CSR.
891 003450 017737 175672 001126 MOV @IBS,$BDDAT ;DID IT CLEAR?
892 003456 001401 BEQ TST13 ;:
893

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

897
898 003460 104003
899

ERROR 3 ;/MODULE FAULT DETECTED:
;CSR FAILED TO CLEAR.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

902
903
904
905
906 003462 000004
907
908

;TEST 13 *MAKE SURE WE CAN SET AND CLEAR BIT06 (IE)

↑TST13: SCOPE

909 003464 012746 000340
910 003470 012746 003476
911 003474 000002
912 003476
913

64\$:
MOV #340,-(SP) ;/PR
MOV #64\$,-(SP) ;/SET CPU PRIORITY ON RETURN
RTI ;/SHOW RETURN ADDRESS
; /CAUSE A RETURN (PUTS NEW STATUS
; IN STATUS REG.)

914 003476 005077 175644
915
916 003502 052777 000100 175636
917 003510 012737 000100 001124
918 003516 017737 175624 001126
919 003524 023737 001124 001126
920 003532 001402
921

CLR #IBS ;CLEAR CSR.
BIS #BIT6,#IBS ;SET IE.
MOV #BIT6,\$GDDAT ;EXPECT ONLY BIT 6 TO SET.
MOV #IBS,\$BDDAT ;READ IBS.
CMP \$GDDAT,\$BDDAT ;DID IE SET?
BEQ IS ;YES - CONTINUE.

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

925
926 003534 104003
927

ERROR 3 ;/MODULE FAULT DETECTED:
;ERROR IN SETTING IE BIT.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

930 003536 000412
931
932 003540 005037 001124
933 003544 042777 000100 175574
934 003552 017737 175570 001126
935 003560 001401
936

15: BR TST14 ;;
CLR \$GDDAT ;EXPECT ZERO CSR AFTER.
BIC #BIT6,#IBS ;IE IS CLEARED.
MOV #IBS,\$BDDAT ;READ CSR - IS IT CLEAR?
BEQ TST14 ;;

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

940
941 003562 104003
942

ERROR 3 ;/MODULE FAULT DETECTED:
;FAILED TO CLEAR CSR.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

945
946
947
948

;TEST 14 *TEST THAT BIT 7 (ACC) CAN BE SET AND CLEARED

MAINDEC-11-DVIBA-A
DVIBA.P11 T14

MACY11 27(663) 29-MAR-77 12:57 PAGE 19
*TEST THAT BIT 7 (ACC) CAN BE SET AND CLEARED

SEQ 0034

```

949 003564 000004          TST14: SCOPE
950
951 003566 005077 175554    CLR      @IBS          ;CLEAR CSR.
952 003572 052777 000200 175546  BIS      @BIT7,@IBS   ;SET ACC.
953 003600 012737 000200 001124  MOV      @BIT7,$GDDAT ;EXPECT ONLY ACC TO SET.
954 003606 017737 175534 001126  MOV      @IBS,$BDDAT  ;READ IBS.
955 003614 023737 001124 001126  CMP      $GDDAT,$BDDAT ;DID ACC SET?
956 003622 001402          BEQ      IS           ;YES - CONTINUE.
957

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

961 003624 104003          ERROR 3              ;/MODULE FAULT DETECTED:
962                                     ;FAILURE IN SETTING BIT 7 (ACC).
963

```

```

966 003626 000412          BR      TST15        ;
967
968 003630 042777 000200 175510 1S:  BIC      @BIT7,@IBS   ;TRY CLEARING ACC.
969 003636 005037 001124          CLR      $GDDAT      ;EXPECT ZERO CSR.
970 003642 017737 175500 001126  MOV      @IBS,$BDDAT ;READ IBS, IS IT CLEAR?
971 003650 001401          BEQ      TST15        ;
972

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

976 003652 104003          ERROR 3              ;/MODULE FAULT DETECTED:
977                                     ;IBS FAILED TO CLEAR.
978

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

981
982
983
984
985
986
987 003654 000004          TST15: SCOPE
988
989
990 003656 012777 000060 175462    MOV      @BIT4!BITS,@IBS ;/MACRO BIT
991 003664 012737 000001 001124    MOV      @BIT0,$GDDAT   ;/SET TON AND LON.
992 003672 013777 001124 175450    MOV      $GDDAT,@IB0    ;/WE'RE GONNA TEST BIT 0.
993                                     ;/SET THE BIT.
994 003700 117737 175444 001126    MOV      @IB0,$BDDAT    ;/READ THE IBO.
995 003706 123737 001124 001126    CMP      $GDDAT,$BDDAT  ;/DID IT GET THRU OK?
996 003714 001402          BEQ      IS           ;/YES - CONTINUE.
997
998

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

MAINDEC-11-DVIBA-A
DVIBA.P11 T15

MACY11 27(663) 29-MAR-77 12:57 PAGE 20
*TEST THAT IBO BIT 0 CAN BET SET + CLEARED

SEQ 0035

1003 003716 104004
1004

ERROR 4 ;/MODULE FAULT DETECTED:
;/ERROR IN SETTING IBO BIT 0.

1007 003720 000412
1008 003722 005037
1009 003726 042777
1010 003734 117737
1011 003742 001401
1012

001124 175414 1S:
000001 175410 001126

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
BR TST16 ;/;
CLR \$GDDAT ;/EXPECT ZERO IBO WHEN
BIC #BIT0, IBO ;/BIT 0 IS CLEARED.
MOVB IBO, \$BDDAT ;/READ IBO, IS IT CLEAR?
BEG TST16 ;/;

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1016
1017 003744 104004
1018

ERROR 4 ;/MODULE FAULT DETECTED:
;/FAILED TO CLEAR IBO.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

1021
1022
1023
1024
1025
1026 003746 000004
1027
1028
1029 003750 012777 000060 175370
1030 003756 012737 000002 001124
1031 003764 013777 001124 175356
1032
1033 003772 117737 175352 001126
1034 004000 123737 001124 001126
1035 004006 001402
1036
1037

: *TEST 16 *TEST THAT IBO BIT 1 CAN BET SET + CLEARED
: *****
TST16: SCOPE

MOV #BIT4:BITS, IBS ;/MACRO BOT
MOV #BIT1, \$GDDAT ;/SET TON AND LON.
MOV \$GDDAT, IBO ;/WE'RE GONNA TEST BIT 1.
;/SET THE BIT.
MOVB IBO, \$BDDAT ;/READ THE IBO.
CMPB \$GDDAT, \$BDDAT ;/DID IT GET THRU OK?
BEG IS ;/YES - CONTINUE.

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1041
1042 004010 104004
1043

ERROR 4 ;/MODULE FAULT DETECTED:
;/ERROR IN SETTING IBO BIT 1.

1046 004012 000412
1047 004014 005037
1048 004020 042777
1049 004026 117737
1050 004034 001401
1051

001124 175322 1S:
000002 175316 001126

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
BR TST17 ;/;
CLR \$GDDAT ;/EXPECT ZERO IBO WHEN
BIC #BIT1, IBO ;/BIT 1 IS CLEARED.
MOVB IBO, \$BDDAT ;/READ IBO, IS IT CLEAR?
BEG TST17 ;/;

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1055
1056 004036 104004

ERROR 4 ;/MODULE FAULT DETECTED:

1057

;FAILED TO CLEAR IBO.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076

004040 000004

; *TEST 17 *TEST THAT IBO BIT 2 CAN BET SET + CLEARED

↑ST17: SCOPE

004042 012777 000060 175276
004050 012737 000004 001124
004056 013777 001124 175264
004064 117737 175260 001126
004072 123737 001124 001126
004100 001402

MOV #BIT4!BITS, @IBS ;/MACRO BOT
MOV #BIT2, \$GDOAT ;/SET TON AND LON.
MOV \$GDOAT, @IBO ;/WE'RE GONNA TEST BIT 2.
; /SET THE BIT.
MOVB @IBO, \$BDOAT ;/READ THE IBO.
CMPB \$GDOAT, \$BDOAT ;/DID IT GET THRU OK?
BEQ IS ;/YES - CONTINUE.

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1080
1081
1082

004102 104004

ERROR 4 ;/MODULE FAULT DETECTED:
;/ERROR IN SETTING IBO BIT 2.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

1085
1086
1087
1088
1089
1090

004104 000412
004106 005037 001124
004112 042777 000004 175230
004120 117737 175224 001126
004126 001401

IS:

BR TST20 ;/;
CLR \$GDOAT ;/EXPECT ZERO IBO WHEN
BIC #BIT2, @IBO ;/BIT 2 IS CLEARED.
MOVB @IBO, \$BDOAT ;/READ IBO, IS IT CLEAR?
BEQ TST20 ;/;

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1094
1095
1096

004130 104004

ERROR 4 ;/MODULE FAULT DETECTED:
;/FAILED TO CLEAR IBO.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110

004132 000004

; *TEST 20 *TEST THAT IBO BIT 3 CAN BET SET + CLEARED

↑ST20: SCOPE

004134 012777 000060 175204
004142 012737 000010 001124
004150 013777 001124 175172

MOV #BIT4!BITS, @IBS ;/MACRO BOT
MOV #BIT3, \$GDOAT ;/SET TON AND LON.
MOV \$GDOAT, @IBO ;/WE'RE GONNA TEST BIT 3.
; /SET THE BIT.

MAINDEC-11-DVIBA-A
DVIBA.P11 T20

MACY11 27(663) 29-MAR-77 12:57 PAGE 22
*TEST THAT IBO BIT 3 CAN BET SET + CLEARED

SEQ 0037

```

1111 004156 117737 175166 001126      MOVB   @IBO,$BDDAT    ;/READ THE IBO.
1112 004164 123737 001124 001126      CMPB   $GDDAT,$BDDAT ;/DID IT GET THRU OK?
1113 004172 001402                      BEQ    1$             ;/YES - CONTINUE.
1114
1115

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

1119
1120 004174 104004      ERROR   4             ;/MODULE FAULT DETECTED:
1121                                     ;/ERROR IN SETTING IBO BIT 3.

```

..SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

```

1124 004176 000412      BR      TST21          ;/
1125 004200 005037 001124      CLR     $GDDAT         ;/EXPECT ZERO IBO WHEN
1126 004204 042777 000010 175136 1$:   BIC     @BIT3,@IBO    ;/BIT 3 IS CLEARED.
1127 004212 117737 175132 001126      MOVB   @IBO,$BDDAT    ;/READ IBO, IS IT CLEAR?
1128 004220 001401      BEQ    TST21          ;/
1129

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

1133
1134 004222 104004      ERROR   4             ;/MODULE FAULT DETECTED:
1135                                     ;/FAILED TO CLEAR IBO.

```

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

```

1138
1139
1140
1141
1142
1143 004224 000004      *TEST 21      *TEST THAT IBO BIT 4 CAN BET SET + CLEARED
1144
1145
1146 004226 012777 000060 175112      MOV     @BIT4!BITS,@IBS ;/MACRO BIT
1147 004234 012737 000020 001124      MOV     @BIT4,$GDDAT    ;/SET TON AND LON.
1148 004242 013777 001124 175100      MOV     $GDDAT,@IBO    ;/WE'RE GONNA TEST BIT 4.
1149                                     ;/SET THE BIT.
1150 004250 117737 175074 001126      MOVB   @IBO,$BDDAT    ;/READ THE IBO.
1151 004256 123737 001124 001126      CMPB   $GDDAT,$BDDAT  ;/DID IT GET THRU OK?
1152 004264 001402                      BEQ    1$             ;/YES - CONTINUE.
1153
1154

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

1158
1159 004266 104004      ERROR   4             ;/MODULE FAULT DETECTED:
1160                                     ;/ERROR IN SETTING IBO BIT 4.

```

..SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

```

1163 004270 000412      BR      TST22          ;/
1164 004272 005037 001124      CLR     $GDDAT         ;/EXPECT ZERO IBO WHEN

```

MAINDEC-11-DVIBA-A
DVIBA.P11 T21

MACY11 27(663) 29-MAR-77 12:57 PAGE 23
*TEST THAT IBO BIT 4 CAN BET SET + CLEARED

SEQ 0038

| | | | | | | | |
|------|--------|--------|--------|--------|------|--------------|--------------------------|
| 1165 | 004276 | 042777 | 000020 | 175044 | BIC | #BIT4, @IBO | ;/BIT 4 IS CLEARED. |
| 1166 | 004304 | 117737 | 175040 | 001126 | MOVB | @IBO, @SBOAT | ;/READ IBO, IS IT CLEAR? |
| 1167 | 004312 | 001401 | | | BEQ | TST22 | ;; |
| 1168 | | | | | | | |

;;SSSSSSSSSS)) ERROR <<<SSSSSSSSSS

| | | | | | | | |
|------|--------|--------|--|--|-------|---|--------------------------|
| 1172 | | | | | | | |
| 1173 | 004314 | 104004 | | | ERROR | 4 | ;/MODULE FAULT DETECTED: |
| 1174 | | | | | | | ;/FAILED TO CLEAR IBO. |

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

| | | | | | | | |
|------|--------|--------|--|--|--|--|--|
| 1177 | | | | | | | |
| 1178 | | | | | | | |
| 1179 | | | | | | | |
| 1180 | | | | | | | ***** |
| 1181 | | | | | | | ;*TEST 22 *TEST THAT IBO BIT 5 CAN BET SET + CLEARED |
| 1182 | 004316 | 000004 | | | | | ***** |
| 1183 | | | | | | | ↑ST22: SCOPE |

| | | | | | | | |
|------|--------|--------|--------|--------|------|------------------|---------------------------|
| 1184 | | | | | | | ;/MACRO BOT |
| 1185 | 004320 | 012777 | 000060 | 175020 | MOV | #BIT4!BITS, @IBS | ;/SET ION AND LON. |
| 1186 | 004326 | 012737 | 000040 | 001124 | MOV | @BITS, @SBOAT | ;/WE'RE GONNA TEST BIT 5. |
| 1187 | 004334 | 013777 | 001124 | 175006 | MOV | @SBOAT, @IBO | ;/SET THE BIT. |
| 1188 | | | | | | | |
| 1189 | 004342 | 117737 | 175002 | 001126 | MOVB | @IBO, @SBOAT | ;/READ THE IBO. |
| 1190 | 004350 | 123737 | 001124 | 001126 | CMPS | @SBOAT, @SBOAT | ;/DID IT GET THRU OK? |
| 1191 | 004356 | 001402 | | | BEQ | IS | ;/YES - CONTINUE. |
| 1192 | | | | | | | |
| 1193 | | | | | | | |

;;SSSSSSSSSS)) ERROR <<<SSSSSSSSSS

| | | | | | | | |
|------|--------|--------|--|--|-------|---|-------------------------------|
| 1197 | | | | | | | |
| 1198 | 004360 | 104004 | | | ERROR | 4 | ;/MODULE FAULT DETECTED: |
| 1199 | | | | | | | ;/ERROR IN SETTING IBO BIT 5. |

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

| | | | | | | | |
|------|--------|--------|--------|--------|------|--------------|--------------------------|
| 1202 | 004362 | 000412 | | | | | |
| 1203 | 004364 | 005037 | 001124 | | 15: | | |
| 1204 | 004370 | 042777 | 000040 | 174752 | CLR | @SBOAT | ;/EXPECT ZERO IBO WHEN |
| 1205 | 004376 | 117737 | 174746 | 001126 | BIC | #BITS, @IBO | ;/BIT 5 IS CLEARED. |
| 1206 | 004404 | 001401 | | | MOVB | @IBO, @SBOAT | ;/READ IBO, IS IT CLEAR? |
| 1207 | | | | | BEQ | TST23 | ;; |

;;SSSSSSSSSS)) ERROR <<<SSSSSSSSSS

| | | | | | | | |
|------|--------|--------|--|--|-------|---|--------------------------|
| 1211 | | | | | | | |
| 1212 | 004406 | 104004 | | | ERROR | 4 | ;/MODULE FAULT DETECTED: |
| 1213 | | | | | | | ;/FAILED TO CLEAR IBO. |

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

| | | | | | | | |
|------|--|--|--|--|--|--|-------|
| 1216 | | | | | | | |
| 1217 | | | | | | | |
| 1218 | | | | | | | ***** |

MAINDEC-11-DVIBA-A
DVIBA.P11 T23

MACY11 27(663) 29-MAR-77 12:57 PAGE 24
*TEST THAT IBO BIT 6 CAN BET SET + CLEARED

SEQ 0039

```

1219 ;*TEST 23 *TEST THAT IBO BIT 6 CAN BET SET + CLEARED
1220 ;*****
1221 TST23: SCOPE
1222
1223 ;/MACRO BOT
1224 004410 000004 MOV #BIT4:BITS, @IBS ;/SET TON AND LON.
1225 004412 012777 000060 174726 MOV #BIT6, $GDDAT ;/WE'RE GONNA TEST BIT 6.
1226 004420 012737 000100 001124 MOV $GDDAT, @IBO ;/SET THE BIT.
1227 004426 013777 001124 174714
1228 004434 117737 174710 001126 MOV @IBO, $BDDAT ;/READ THE IBO.
1229 004442 123737 001124 001126 CMPB $GDDAT, $BDDAT ;/DID IT GET THRU OK?
1230 004450 001402 BEQ IS ;/YES - CONTINUE.
1231
1232

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

1236 ERROR 4 ;/MODULE FAULT DETECTED:
1237 004452 104004 ;/ERROR IN SETTING IBO BIT 6.
1238

```

;;SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

```

1241 004454 000412 BR TST24 ;/
1242 004456 005037 001124 174660 15: CLR $GDDAT ;/EXPECT ZERO IBO WHEN
1243 004462 042777 000100 174660 BIC #BIT6, @IBO ;/BIT 6 IS CLEARED.
1244 004470 117737 174654 001126 MOV @IBO, $BDDAT ;/READ IBO, IS IT CLEAR?
1245 004476 001401 BEQ TST24 ;/
1246

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

1250 ERROR 4 ;/MODULE FAULT DETECTED:
1251 004500 104004 ;/FAILED TO CLEAR IBO.
1252

```

;;SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

```

1255 ;*****
1256 ;*TEST 24 *TEST THAT IBO BIT 7 CAN BET SET + CLEARED
1257 ;*****
1258 TST24: SCOPE
1259
1260 004502 000004 ;/MACRO BOT
1261
1262 004504 012777 000060 174634 MOV #BIT4:BITS, @IBS ;/SET TON AND LON.
1263 004512 012737 000200 001124 MOV #BIT7, $GDDAT ;/WE'RE GONNA TEST BIT 7.
1264 004520 013777 001124 174622 MOV $GDDAT, @IBO ;/SET THE BIT.
1265
1266 004526 117737 174616 001126 MOV @IBO, $BDDAT ;/READ THE IBO.
1267 004534 123737 001124 001126 CMPB $GDDAT, $BDDAT ;/DID IT GET THRU OK?
1268 004542 001402 BEQ IS ;/YES - CONTINUE.
1269
1270
1271

```

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS
1275
1276 004544 104004 ERROR 4 ;/MODULE FAULT DETECTED:
1277 ;/ERROR IN SETTING IBO BIT 7.

..SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
1280 004546 000412 BR TST25 ;
1281 004550 005037 001124 15: CLR $GDDAT ;/EXPECT ZERO IBO WHEN
1282 004554 042777 000200 174566 BIC #BIT7,IBO ;/BIT 7 IS CLEARED.
1283 004562 117737 174562 001126 MOVB IBO,$B00AT ;/READ IBO, IS IT CLEAR?
1284 004570 001401 BEQ TST25 ;
1285

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS
1289
1290 004572 104004 ERROR 4 ;/MODULE FAULT DETECTED:
1291 ;/FAILED TO CLEAR IBO.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
1294
1295
1296 *****
1297 *TEST 25 *TEST THAT NO DATA GETS XFERRERD, IF NOT ENABLED
1298 *****
1299 TST25: SCOPE
1300
1301 004576 005077 174544 CLR IBS ;CLEAR CSR
1302 004602 112777 000252 174540 MOVB #252,IBO ;TRY XFERRING DATA
1303 004610 005037 001124 CLR $GDDAT ;NO DATA SHOULD XFERR
1304 004614 117737 174530 001126 MOVB IBO,$B00AT ;READ BUFFER REG.
1305 004622 001401 BEQ TST26 ;
1306

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS
1310
1311 004624 104002 ERROR 2 ;/MODULE FAULT DETECTED:
1312 ;/DATA WAS XFERRERD THROUGH IBO
1313 ;/EVEN THOUGH TON AND LON CLEARED.
1314 ;/SIGNAL "ENB XFER L" PROBABLY
1315 ;/STUCK LOW.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
1318
1319 *****
1320 *TEST 25 *TEST IBO BITS DAC, AND DAV
1321 *****
1322 TST26: SCOPE
1323
1324 004630 005077 174512 CLR IBS ;CLEAR CSR.
1325 004634 005077 174510 CLR IBO ;CLEAR DATA REG.
1326 004640 032777 000400 174502 BIT #BIT8,IBO ;IS DAC SET?

```

1327 004646 001002
1328

BNE 45 ;YES (GOOD)

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1332 004650 104002
1333
1334

ERROR 2 ;/MODULE FAULT DETECTED:
;DAC NOT SET.

..SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

1337 004652 000437
1338 004654
1339 004654 052777 000260 174464
1340 004662 012777 000252 174460
1341 004670 017737 174454 001126
1342 004676 032737 001000 001126
1343 004704 001002
1344

45:

BR TST27 ;;
BIS #BITS!BIT4!BIT7,2IBS ;SET TON AND LON
MOV #252,2IB0 ;PUT DATA IN IBO.
MOV 2IB0,SBOCT ;READ IBO.
BIT #BIT9,SBUAT ;DID DAV SET?
BNE 15 ;YES - CONTINUE.

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1348 004706 104002
1349
1350

ERROR 2 ;/MODULE FAULT DETECTED:
;DAV FAILED TO SET.

..SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

1353 004710 000420
1354
1355 004712 012777 000060 174426
1356 004720 105777 174424
1357 004724 032777 000400 174416
1358 004732 001402
1359

15:
25:

BR TST27 ;;
MOV #BIT4!BITS,2IBS ;CLEAR ACC.
TSTB 2IB0 ;READ LOW BYTE OF IBO.
BIT #BIT8,2IB0 ;DID DAC CLEAR?
BEQ 35 ;YES - CONTINUE.

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1363 004734 104002
1364
1365

ERROR 2 ;/MODULE FAULT DETECTED:
;DAC FAILED TO CLEAR.

..SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

1368 004736 000405
1369 004740 032777 001000 174402
1370 004746 001401
1371

35:

BR TST27 ;;
BIT #BIT9,2IB0 ;DID DAV CLEAR?
BEQ TST27 ;;

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1375 004750 104002
1376
1377

ERROR 2 ;/MODULE FAULT DETECTED:
;DAV FAILED TO CLEAR.

..SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

1380

;/MACRO -SIGC-

1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397

004752 000004
004754 005077 174366
004760 052777 000004 174360
004766 032777 010000 174354
004774 001011
004776 052777 000004 174342
005004 032777 010000 174336
005012 001002

```
*****  
*TEST 27 *TEST THAT REN SETS WHEN REN SETS, ALSO TEST CLEAR  
*****  
TST27: SCOPE  
CLR @IBS ;/CLEAR CSR.  
BIS #BIT2,@IBS ;/SET REN, SHOULD SET REN.  
BIT #BIT12,@IBD ;/DID REN SET?  
BNE IS ;/YES - LETS TRY CLEARING IT.  
BIS #BIT2,@IBS ;/SET REN, MEMORY  
;/REFRESH COULD HAVE  
;/INTERRUPTED US.  
BIT #BIT12,@IBD ;/DID REN SET THIS TIME?  
BNE IS
```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1401
1402
1403

005014 104002

```
ERROR 2 ;/MODULE FAULT DETECTED:  
;/REN FILED TO SET WHEN REN SET.
```

;;SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

1406
1407
1408
1409
1410

005016 000410
005020 042777 000004 174320
005026 032777 010000 174314
005034 001401

```
BR TST30 ;/  
BIC #BIT2,@IBS ;/CLEAR REN, SHOULD CLEAR REN.  
BIT #BIT12,@IBD ;/DID REN CLEAR?  
BEQ TST30 ;/
```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1414
1415
1416

005036 104002

```
ERROR 2 ;/MODULE FAULT DETECTED:  
;/REN FAILED TO CLEAR.
```

;;SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434

005040 000004
005042 005077 174300
005046 052777 000010 174272
005054 032777 020000 174266
005062 001011
005064 052777 000010 174254

```
*****  
*TEST 30 *TEST THAT IFC SETS WHEN IBS SETS, ALSO TEST CLEAR  
*****  
TST30: SCOPE  
CLR @IBS ;/CLEAR CSR.  
BIS #BIT3,@IBS ;/SET IBS, SHOULD SET IFC.  
BIT #BIT13,@IBD ;/DID IFC SET?  
BNE IS ;/YES - LETS TRY CLEARING IT.  
BIS #BIT3,@IBS ;/SET IBS, MEMORY  
;/REFRESH COULD HAVE  
;/INTERRUPTED US.
```

;/MACRO -SIGC-

MAINDEC-11-DVIBA-A
DVIBA.P11 T30

MACY11 27(663) 29-MAR-77 12:57 PAGE 28
*TEST THAT IFC SETS WHEN IBS SETS, ALSO TEST CLEAR

SEQ 0043

1435 005072 032777 020000 174250 BIT #BIT13, IBS ;/DID IFC SET THIS TIME?
1436 005100 001002 BNE IS
1437

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1441 005102 104002 ERROR 2 ;/MODULE FAULT DETECTED:
1442 ;/IFC FILED TO SET WHEN IBS SET.
1443

1446 005104 000411 BR TST31 ;
1447 ;

1448 005106 032777 000010 174232 IS: BIT #BIT3, IBS ;/WAIT FOR IBS TO CLEAR.
1449 005114 001374 BNE IS
1450

1451 005116 032777 020000 174224 BIT #BIT13, IBS ;/IBS CLEAR, DID IFC CLEAR?
1452 005124 001401 BEQ TST31 ;
1453

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1457 005126 104002 ERROR 2 ;/MODULE FAULT DETECTED:
1458 ;/IFC FAILED TO CLEAR.
1459

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1462 ;/MACRO -SIGC-
1463
1464
1465

1466 ;*****
1467 ;*TEST 31 *TEST THAT ATN SETS WHEN TCS SETS, ALSO TEST CLEAR
1468 ;*****
1469 TST31: SCOPE

1471 005132 005077 174210 CLR IBS ;/CLEAR CSR.
1472 005136 052777 000001 174202 BIS #BIT0, IBS ;/SET TCS, SHOULD SET ATN.
1473 005144 032777 040000 174176 BIT #BIT14, IBS ;/DID ATN SET?
1474 005152 001011 BNE IS ;/YES - LETS TRY CLEARING IT.
1475 005154 052777 000001 174164 BIS #BIT0, IBS ;/SET TCS, MEMORY
1476 ;/REFRESH COULD HAVE
1477 ;/INTERRUPTED US.
1478 005162 032777 040000 174160 BIT #BIT14, IBS ;/DID ATN SET THIS TIME?
1479 005170 001002 BNE IS
1480

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1484 005172 104002 ERROR 2 ;/MODULE FAULT DETECTED:
1485 ;/ATN FILED TO SET WHEN TCS SET.
1486

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

MAINDEC-11-DVIBA-A
DVIBA.P11 T31

MACY11 27(663) 29-MAR-77 12:57 PAGE 29
*TEST THAT ATN SETS WHEN TCS SETS, ALSO TEST CLEAR

SEQ 0044

| | | | | | | | |
|------|--------|--------|--------|--------|-----|-------------|--------------------------------|
| 1489 | 005174 | 000410 | | | BR | TST32 | ;; |
| 1490 | 005176 | 042777 | 000001 | 174142 | BIC | #BIT0,2IBS | ;/CLEAR TCS, SHOULD CLEAR ATN. |
| 1491 | 005204 | 032777 | 040000 | 174136 | BIT | #BIT14,2IBD | ;/DID ATN CLEAR? |
| 1492 | 005212 | 001401 | | | BEQ | TST32 | ;; |
| 1493 | | | | | | | |

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

| | | | | | | | |
|------|--------|--------|--|--|-------|---|--------------------------|
| 1497 | | | | | | | |
| 1498 | 005214 | 104002 | | | ERROR | 2 | ;/MODULE FAULT DETECTED: |
| 1499 | | | | | | | ;/ATN FAILED TO CLEAR. |

;;SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

| | | | | | | | |
|------|--|--|--|--|--|--|----------------|
| 1502 | | | | | | | |
| 1503 | | | | | | | |
| 1504 | | | | | | | ;/MACRO -SIGC- |
| 1505 | | | | | | | |
| 1506 | | | | | | | |

```

;*****
;TEST 32 *TEST THAT EOI SETS WHEN EOP SETS, ALSO TEST CLEAR
;*****

```

| | | | | | | | |
|------|--------|--------|--------|--------|--------|-------------|-------------------------------|
| 1508 | | | | | ↑ST32: | SCOPE | |
| 1509 | 005216 | 000004 | | | | | |
| 1510 | | | | | | | |
| 1511 | 005220 | 005077 | 174122 | | CLR | 2IBS | ;/CLEAR CSR. |
| 1512 | 005224 | 052777 | 000002 | 174114 | BIS | #BIT1,2IBS | ;/SET EOP, SHOULD SET EOI. |
| 1513 | 005232 | 032777 | 100000 | 174110 | BIT | #BIT15,2IBD | ;/DID EOI SET? |
| 1514 | 005240 | 001011 | | | BNE | IS | ;/YES - LETS TRY CLEARING IT. |
| 1515 | 005242 | 052777 | 000002 | 174076 | BIS | #BIT1,2IBS | ;/SET EOP, MEMORY |
| 1516 | | | | | | | ;/REFRESH COULD HAVE |
| 1517 | | | | | | | ;/INTERRUPTED US. |
| 1518 | 005250 | 032777 | 100000 | 174072 | BIT | #BIT15,2IBD | ;/DID EOI SET THIS TIME? |
| 1519 | 005256 | 001002 | | | BNE | IS | |
| 1520 | | | | | | | |

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

| | | | | | | | |
|------|--------|--------|--|--|-------|---|----------------------------------|
| 1524 | | | | | | | |
| 1525 | 005260 | 104002 | | | ERROR | 2 | ;/MODULE FAULT DETECTED: |
| 1526 | | | | | | | ;/EOI FILED TO SET WHEN EOP SET. |

;;SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

| | | | | | | | |
|------|--------|--------|--------|--------|-----|-------------|--------------------------------|
| 1529 | 005262 | 000410 | | | BR | TST33 | ;; |
| 1530 | 005264 | 042777 | 000002 | 174054 | BIC | #BIT1,2IBS | ;/CLEAR EOP, SHOULD CLEAR EOI. |
| 1531 | 005272 | 032777 | 100000 | 174050 | BIT | #BIT15,2IBD | ;/DID EOI CLEAR? |
| 1532 | 005300 | 001401 | | | BEQ | TST33 | ;; |
| 1533 | | | | | | | |

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

| | | | | | | | |
|------|--------|--------|--|--|-------|---|--------------------------|
| 1537 | | | | | | | |
| 1538 | 005302 | 104002 | | | ERROR | 2 | ;/MODULE FAULT DETECTED: |
| 1539 | | | | | | | ;/EOI FAILED TO CLEAR. |

;;SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

1542

```

1543
1544
1545
1546
1547 005304 000004
1548
1549 005306 005077 174034 CLR @IBS ;CLEAR CSR.
1550 005312 032777 002000 174030 BIT @BIT10,@IBD ;DID RFD SET?
1551 005320 001002 BNE IS ;YES CONTINUE.
1552

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

1556
1557 005322 104002 ERROR 2 ;/MODULE FAULT DETECTED:
1558 ;RFD FAILED TO SET.

```

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

```

1561 005324 000410 BR TST34 ;
1562
1563 005326 052777 000200 174012 IS: BIS @BIT7,@IBS ;NOW SET ACC,RFD SHOULD CLEAR.
1564 005334 032777 002000 174006 BIT @BIT10,@IBD ;DID IT CLEAR?
1565 005342 001401 BEQ TST34 ;
1566

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

1570
1571 005344 104002 ERROR 2 ;/MODULE FAULT DETECTED:
1572 ;RFD FAILED TO CLEAR.

```

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

```

1575
1576
1577
1578
1579
1580 005346 000004
1581
1582 005350 005077 173772 CLR @IBS ;CLEAR THE STATUS REG.
1583 005354 052777 000041 173764 BIS @BITS!BIT0,@IBS ;SET TON; THIS SHOULD CAUSE AN
1584 ;ERROR SENCE NO LISTENERS ARE ON
1585 005362 105077 173762 CLRB @IBD ;AND WE SENT DATA TO THE BUS.
1586 ;BUS.
1587 005366 032777 040000 173752 BIT @BIT14,@IBS ;DID ER2 SET?
1588 005374 001001 BNE TST35 ;
1589

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

1593
1594 005376 104001 ERROR 1 ;/MODULE FAULT DETECTED:
1595 ;ER2 FAILED TO SET.

```

1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609

005400 000004
005402 012737 000005 001160
005410 012777 000367 173730
005416 000005
005420 105777 173722
005424 001401

```

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
:*****
: *TEST 35 *TEST THAT BUS INIT CLEARS ACC,TON,LON,REM,EIP,TCS
:*****
↑ST35: SCOPE
MOV #5,STIMES ;;DO 5 ITERATIONS
MOV #367,@IBS ;SET ACC,TON,LON,REM,EOP, AND TCS.
RESET ;ISSUE SYS INIT.
TSTB @IBS ;DID THEY ALL CLEAR?
BEQ TST36 ;;

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1613
1614
1615
1616

005426 104001

ERROR 1 ;/MODULE FAULT DETECTED:
;BUS INIT FAILED TO CLEAR CSR.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632

005430 000004
005432 012737 000005 001160
005440 012777 000266 173700
005446 052777 000010 173672
005454 032777 000010 173664
005462 001374
005464 032777 000266 173654
005472 001401

```

:*****
: *TEST 36 *TEST IBC CLEARS ACC,TON,LON,REM AND EOP
:*****
↑ST36: SCOPE
MOV #5,STIMES ;;DO 5 ITERATIONS
MOV #266,@IBS ;SET ACC,TON,LON,REM, AND EOP.
BIS #BIT3,@IBS ;SET IBC, THIS SHOULD CLEAR ABOVE BITS.
15: BIT #BIT3,@IBS ;WAIT TILL IBS CLEARS
BNE 15 ;
BIT #266,@IBS ;DID THEY CLEAR?
BEQ TST37 ;;

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1636
1637
1638
1639

005474 104001

ERROR 1 ;/MODULE FAULT DETECTED:
;ACC,TON,LON,REM, AND/OR EOP
;FAILED TO CLEAR ON IBC

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

1642
1643
1644
1645
1646
1647
1648
1649
1650

005476 000004
005500 012737 000005 001160
005506 012777 000260 173632
005514 012777 000377 173626

```

:*****
: *TEST 37 *TEST THAT BUS INIT INDIRECTLY CLEARS IBD
:*****
↑ST37: SCOPE
MOV #5,STIMES ;;DO 5 ITERATIONS
MOV #BIT7!BITS!BIT4,@IBS ;SET ACC,TON, AND LON.
MOV #377,@IBD ;LOAD IBD

```

MAINDEC-11-DVIBA-A
DVIBA.P11 T37

MACY11 27(663) 29-MAR-77 12:57 PAGE 32
*TEST THAT BUS INIT INDIRECTLY CLEARS IBO

SEQ 0047

1651 005522 000005
1652 005524 105777 173620
1653 005530 001401
1654

RESET ;ISSUE SYS INIT.
TSTB @IBO ;DID IT CLEAR?
BEQ TST40 ;

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1658
1659 005532 104002
1660
1661

ERROR 2 ;/MODULE FAULT DETECTED:
;FAILED TO CLEAR LOW BYTE OF IBO ON
;SYSTEM INIT.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

1664
1665
1666
1667
1668
1669
1670
1671
1672 005534 000004
1673
1674 005536 005077 173604
1675 005542 012777 000200 173636
1676 005550 012777 005606 173604
1677 005556 052777 000101 173562
1678
1679 005564 012746 000000
1680 005570 012746 005576
1681 005574 000002
1682 005576
1683 005576 000240
1684 005600 000240
1685

.SBTTL
.SBTTL INTERRUPT TESTS
.SBTTL

;TEST 40 *TEST THAT CMD CAN GENERATE AN INTERRUPT B

↑ST40: SCOPE

CLR @IBS ;CLEAR THE CSR.
MOV #200,@PRC
MOV #15,@VECTC ;SET UP INTERRUPT VECTOR
BIS @BIT0!BIT6,@IBS ;SET TCS, SHOULD CAUSE
;PR
MOV #0,-(SP) ;SET CPU PRIORITY ON RETURN
MOV #64\$,-(SP) ;SHOW RETURN ADDRESS
RTI ;CAUSE A RETURN (PUTS NEW STATUS
;IN STATUS REG.)
NOP ;CMD TO SET AND GIVE US AN
NOP ;INTERRUPT.

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1687
1688 005602 104001
1689

ERROR 1 ;/MODULE FAULT DETECTED:
;CMD FAILED TO GENERATE AN INTERRUPT.

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

1694 005604 000402
1695 005606
1696 005606 062706 000004
1697 005612 005077 173530
1698
1699 005616 013777 001406 173536
1700 005624 012777 004700 173554
1701
1702
1703
1704

BR 2\$
1\$: ADD #4,SP ;/ADD #4 TO STACK POINTER.
2\$: CLR @IBS ;CLEAR INTERRUPT
;/-RESV-
MOV PRC,@VECTC ;/RESTORE VECTOR FOR
MOV #4700,@PRC ;/ILLEGAL INTRO.

;TEST 41 *TEST THAT TCR AND LNR CAN GENERATE INTERRUPTS

```

1705 005632 000004          TST41: SCOPE
1706 005634 012777 000200 173544      MOV      #200, @PRC
1707 005642 012777 005726 173512      MOV      #1$, @VECTC ;SET UP INTERRUPT VECTOR FOR TKR INTERRUPT
1708 005650 012777 000060 173470      MOV      #BIT4!BITS, @IBS ;SET TON AND LON
1709 005656 052777 000100 173462      BIS      #BIT6, @IBS ;ALLOW INTERRUPT
1710                                ;/PR
1711 005664 012746 000000          MOV      #0, -(SP) ;/SET CPU PRIORITY ON RETURN
1712 005670 012746 005676          MOV      #64$, -(SP) ;/SHOW RETURN ADDRESS
1713 005674 000002          RTI      ;/CAUSE A RETURN (PUTS NEW STATUS
1714 005676                                ;/IN STATUS REG.)
1715 005676 000240          NOP
1716 005700 000240          NOP
1717

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

1721
1722 005702 104001          ERROR 1 ;/MODULE FAULT DETECTED:
1723                                ;/FAILED TO GENERATE A TKR INTERRUPT.

```

;;SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

```

1726 005704 005077 173436      CLR      @IBS ;CLR CSR
1727                                ;/-RESV-
1728 005710 013777 001406 173444      MOV      PRC, @VECTC ;/RESTORE VECTOR FOR
1729 005716 012777 004700 173462      MOV      #4700, @PRC ;/ILLEGAL INTRO.
1730 005724 000443          BR       TST42 ;
1731
1732 005726                                15:
1733 005726 062706 000004          ADD      #4, SP ;/ADD #4 TO STACK POINTER.
1734                                ;/-RESV-
1735 005732 013777 001406 173422      MOV      PRC, @VECTC ;/RESTORE VECTOR FOR
1736 005740 012777 004700 173440      MOV      #4700, @PRC ;/ILLEGAL INTRO.
1737 005746 012777 000200 173434      MOV      #200, @PRD
1738 005754 012777 006010 173402      MOV      #2$, @VECTD ;SET UP FOR LNR INTERRUPT.
1739                                ;/PR
1740 005762 012746 000000          MOV      #0, -(SP) ;/SET CPU PRIORITY ON RETURN
1741 005766 012746 005774          MOV      #65$, -(SP) ;/SHOW RETURN ADDRESS
1742 005772 000002          RTI      ;/CAUSE A RETURN (PUTS NEW STATUS
1743 005774                                ;/IN STATUS REG.)
1744 005774 105277 173350          INCB    @IBD ;SEND DATA - CLRS TKR SETS LNR
1745                                ;/FOR INTERRUPT .
1746 006000 000240          NOP
1747 006002 000240          NOP
1748

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

1752
1753 006004 104001          ERROR 1 ;/MODULE FAULT DETECTED:
1754                                ;/FAILED TO GENERATE LNR INTERRUPT

```

;;SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

```

1757 006006 000402          BR       35
1758

```

```

1759 006010 25:
1760 006010 062706 000004 ADD #4,SP ;/ADD #4 TO STACK POINTER.
1761 006014 005077 173326 CLR #18 ;CLEAR THE STATUS REG.
1762 ;/RESV-
1763 006020 013777 001410 173336 MOV PRA,#VECTA ;/RESTORE VECTOR FOR
1764 006026 012777 004700 173354 MOV #4700,#PRA ;/ILLEGAL INTRO.
1765
1766 ;*****
1767 ;*TEST 42 *TEST THAT ER2 CAN GENERATE AN INTERRUPT
1768 ;*****
1769 006034 000004 †ST42: SCOPE
1770
1771 006036 005077 173304 CLR #18 ;START WITH CSR CLEAR
1772 006042 012777 000200 173332 MOV #200,#PRA
1773 006050 012777 006126 173300 MOV #18,#VECTA ;SET UP INTERRUPT VECTOR
1774 ;/PR
1775 006056 012746 000200 MOV #200,-(SP) ;/SET CPU PRIORITY ON RETURN
1776 006062 012746 006070 MOV #648,-(SP) ;/SHOW RETURN ADDRESS
1777 006066 000002 RTI ;/CAUSE A RETURN (PUTS NEW STATUS
1778 006070 645: ;/IN STATUS REG.)
1779 006070 052777 000140 173250 BIS #BIT5!BIT6,#18 ;SET TON - NO LISTNERS ON
1780 006076 105077 173246 CLR #18 ;BUS BUT DATA PUT ON
1781 006102 000240 NOP ;BUS - THEREFORE AN INTERRUPT
1782 006104 000240 NOP ;SHOULD BE POSTED.
1783 ;/PR
1784 006106 012746 000000 MOV #0,-(SP) ;/SET CPU PRIORITY ON RETURN
1785 006112 012746 006120 MOV #655,-(SP) ;/SHOW RETURN ADDRESS
1786 006116 000002 RTI ;/CAUSE A RETURN (PUTS NEW STATUS
1787 006120 655: ;/IN STATUS REG.)
1788 006120 000240 NOP
1789
1790

```

:: SSSSSSSSSS))) ERROR (((SSSSSSSSSS

```

1794
1795 006122 104001 ERROR 1 ;/MODULE FAULT DETECTED:
1796 ;/FAILED TO INTERRUPT ON ERROR2

```

:: SSSSSSSSSS))) ERROR (((SSSSSSSSSS

```

1799 006124 000402 BR 25
1800 006126 15:
1801 006126 062706 000004 ADD #4,SP ;/ADD #4 TO STACK POINTER.
1802 006132 005077 173210 CLR #18 ;CLEAR CSR
1803 ;/RESV-
1804 006136 013777 001402 173212 MOV PRA,#VECTA ;/RESTORE VECTOR FOR
1805 006144 012777 004700 173230 MOV #4700,#PRA ;/ILLEGAL INTRO.
1806
1807 .SBTTL
1808 .SBTTL SECOND MODULE TESTS
1809 .SBTTL
1810
1811 ;*****
1812 ;*TEST 43 *TEST THAT MODULE PASSES "BIAKI"

```

```

1813                                     ;:*****
1814 006152 000004                       †ST43: SCOPE
1815
1816
1817                                     ;:WARNING! THIS TEST IS DESIGNED TO BE EXERCISED WITH A
1818                                     ;:SECOND MODULE (IBV-11) WITH SWITCH "ERI INH" SET.
1819                                     ;:ADDRESS OF THE SECOND MODULE IS IN LOCATION "IBS2" VECTOR
1820                                     ;:ADDRESS IS IN LOCATION "VECTA2". THE SECOND IBV-11 SHOULD BE ELECTRICALLY SEC
1821                                     ;:TO INHIBIT THE USE OF TESTING WITH A SECOND MODULE, MAKE
1822                                     ;:LOCATION "SCDW1" ZERO.
1823                                     ;:
1824
1825 006154 005737 001254                   TST      $CDW1      ;TESTING WITH
1826 006160 001002                       BNE      3$      ;SECOND IBV11?
1827 006162 000137 006764                   JMP      EOP     ;NO-END PASS.
1828 006166
1829
1830                                     3$:
1831 006166 005077 173154                   CLR      @IBS     ;CLEAR CSR.
1832 006172 005077 173170                   CLR      @IBS2   ;CLEAR SECOND MODULE.
1833 006176 012777 000200 173212           MOV      #200,@PRC2
1834 006204 012777 006242 173164           MOV      #15,@VECTC2 ;SET UP VECTOR ADDR.
1835 006212 012746 000000                       /PR
1836 006216 012746 006224                       /SET CPU PRIORITY ON RETURN
1837 006222 000002                       MOV      #0,-(SP) ;SHOW RETURN ADDRESS
1838 006224                                     RTI          ;CAUSE A RETURN (PUTS NEW STATUS
1839 006224 012777 000140 173134           MOV      #BIT6:BIT5,@IBS2 ;/IN STATUS REG.)
1840 006232 000240                                     ;SET INTR ENABLE AND TOM ON SECOND
1841 006234 000240                                     ;IBV - SHOULD CAUSE A TKR INTERRUPT.
1842
                                     ;:*****
                                     ;:*****
1846 006236 104001                       ERROR 1      ;/MODULE FAULT DETECTED:
1847                                     ;:ASSUMING SECOND MODULE IS GOOD,
1848                                     ;:MODULE (IBV-11) UNDER TEST FAILED
1849                                     ;:TO PASS @ BUSS SIGNAL "BIAKI"
1850
                                     ;:*****
                                     ;:*****
1853 006240 000402                       BR      2$
1854 006242                                     1$:
1855 006242 062706 000004                   ADD      #4,SP   ;/ADD #4 TO STACK POINTER.
1856 006246 005077 173114                   CLR      @IBS2   ;CLEAR SECOND MODULE
1857                                     ;:-RESV-
1858 006252 013777 001416 173116           MOV      PRC2,@VECTC2 ;/RESTORE VECTOR FOR
1859 006260 012777 004700 173130           MOV      #4700,@PRC2 ;/ILLEGAL INTRO.
1860
1861                                     ;:*****
1862                                     ;:TEST 44 *TEST THAT SRQ CAN GENERATE AN INTERRUPT
1863                                     ;:*****
1864 006266 000004                       †ST44: SCOPE
1865                                     ;:WARNING! THIS TEST IS DESIGNED TO BE EXERCISED WITH A
1866                                     ;:SECOND MODULE (IBV-11) WITH SWITCH "ERI INH" SET.

```

M04

MAINDEC-11-DVIBA-A
DVIBA.P11 T44

MACY11 27(663) 29-MAR-77 12:57 PAGE 36
*TEST THAT SRQ CAN GENERATE AN INTERRUPT

SEQ 0051

```

1867 ;*ADDRESS OF THE SECONUD MODULE IS IN LOCATION "IBS2" VECTOR
1868 ;*ADDRESS IS IN LOCATION "VECTA2". THE SECONUD IBV-11 SHOULD BE ELECTRICALLY SEC
1869 ;*TO INHIBIT THE USE OF TESTING WITH A SECONUD MODULE, MAKE
1870 ;*LOCATION "SCDW1" ZERO.
1871 ;*
1872
1873 006270 005077 173052 CLR @IBS ;CLEAR CSRS.
1874 006274 005077 173066 CLR @IBS2
1875
1876 006300 012777 000200 173076 MOV #200,@PRB ;SET UP INTERRUPT VECTOR.
1877 006306 012777 006352 173044 MOV #15,@VECTB
1878
1879 006314 012746 000000 MOV #0,-(SP) ;/PR
1880 006320 012746 006326 MOV #64$,-(SP) ;/SET CPU PRIORITY ON RETURN
1881 006324 000002 RTI ;/SHOW RETURN ADDRESS
1882 ;/CAUSE A RETURN (PUTS NEW STATUS
1883 ;/IN STATUS REC.)
1884 006326 012777 000100 173012 MOV #100,@IBS ;ENABLE INTERRUPTS
1885 006334 052777 100000 173024 BIS #BIT15,@IBS2 ;SETTING SRQ IN THE "CDW" MODULE
1886 ;WILL PUT SRQ ON THE IB BUS
1887 006342 000240 NOP ;IS ERI INH SW IS SET.
1888 006344 000240 NOP
1889

```

64\$:

::\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$

```

1893
1894 006346 104001 ERROR 1 ;/MODULE FAULT DETECTED:
1895 ;SRQ FAILED TO GENERATE
1896 ;AN INTERRUPT

```

::\$\$\$\$\$\$\$\$\$\$↑↑↑ ERROR ↑↑↑\$\$\$\$\$\$\$\$\$\$

```

1899 006350 000402 BR 25
1900
1901 006352 062706 000004 1$: ADD #4,SP ;/ADD #4 TO STACK POINTER.
1902
1903
1904 006356 2$:
1905
1906 006356 013777 001404 172774 MOV PRB,@VECTB ;/-RESV-
1907 006364 012777 004700 173012 MOV #4700,@PRB ;/RESTORE VECTOR FOR
1908 006372 005077 172750 CLR @IBS ;/ILLEGAL INTRO.
1909 006376 005077 172764 CLR @IBS2 ;CLEAR CSRS
1910

```

```

;*****
; *TEST 45 *TEST THAT ERROR1 IS GENERATED IF ATN IS ON THE IB BUS
;*****
↑ST45: SCOPE

```

```

; *WARNING! THIS TEST IS DESIGNED TO BE EXERCISED WITH A
; *SECONUD MODULE (IBV-11) WITH SWITCH "ERI INH" SET.
; *ADDRESS OF THE SECONUD MODULE IS IN LOCATION "IBS2" VECTOR
; *ADDRESS IS IN LOCATION "VECTA2". THE SECONUD IBV-11 SHOULD BE ELECTRICALLY SEC
; *TO INHIBIT THE USE OF TESTING WITH A SECONUD MODULE, MAKE

```

1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931

006404 005077 172756
006410 005277 172752

006414 032777 020000 172724
006422 001001

```
;;#LOCATION "SCDW1" ZERO.
;*
CLR  @IBS2      ;CLR CSR OF 2ND MODULE.
INC  @IBS2      ;ASSERT ATN ON IB BUS
                        ;ASSERTED ATN ON IBV UNDER TEST-
                        ;THIS SHOULD CAUSE AN ERROR 1
                        ;SENCE THE 2ND IBV HAS ATN SET.
BIT  @BIT13,@IBS ;DID ERROR 1 SET?
BNE  TST46      ;;
```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1935
1936
1937

006424 104001

```
ERROR 1 ;/MODULE FAULT DETECTED:
        ;FAILED TO GENERATE ERROR 1
```

;;SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

1940
1941
1942
1943

006426 000004
006430 012737 000005 001160

```
*****
;#TEST 46      *TEST THAT ERROR 1 IS GENERATED IF IFC IS PUT ON IB BUS BY SECONUD MODUL
*****
```

```
TST46: SCOPE
MOV  #5,$TIMES ;:DO 5 ITERATIONS
;#WARNING! THIS TEST IS DESIGNED TO BE EXERCISED WITH A
;#SECONUD MODULE (IBV-11) WITH SWITCH "ERI INH" SET.
;#ADDRESS OF THE SECONUD MODULE IS IN LOCATION "IBS2" VECTOR
;#ADDRESS IS IN LOCATION "VECTA2". THE SECONUD IBV-11 SHOULD BE ELECTRICALLY SEC
;#TO INHIBIT THE USE OF TESTING WITH A SECONUD MODULE, MAKE
;#LOCATION "SCDW1" ZERO.
;#
```

1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961

006436 005077 172704
006442 012777 000010 172716
006450 032777 020000 172670

006456 001010
006460 012777 000010 172700
006466 032777 020000 172652
006474 001001

```
CLR  @IBS      ;CLEAR CSR
MOV  @BIT3,@IBS2 ;ASSERT IFC FROM TESTOR
BIT  @BIT13,@IBS ;DID ERROR 1 GET SET?
                        ;IF SO - NEXT TEST
BNE  TST47      ;
MOV  @BIT3,@IBS2 ;IF NOT WE'LL TRY AGAIN SENCE MEMORY
BIT  @BIT13,@IBS ;REFRESH COULD HAVE GO IN THE WAY.
BNE  TST47      ;;
```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1965
1966
1967
1968
1969

006476 104001

```
ERROR 1 ;/MODULE FAULT DETECTED:
        ;ERROR 1 FAILED TO SET WHEN
        ;IFC WAS ON IB-BUS AND MODULE
        ;UNDER TEST DIDN'T PUT IT THERE.
```

;;SSSSSSSSSS+++ ERROR +++SSSSSSSSSS

1972
1973
1974

```
*****
;#TEST 47      *TEST THAT ERROR 1 IS GENERATED IF REN IS ON IB BUS
```

```

1975          ::*****
1976 006500 000004      †T47: SCOPE
1977
1978          ;#WARNING! THIS TEST IS DESIGNED TO BE EXERCISED WITH A
1979          ;#SECOND MODULE (IBV-11) WITH SWITCH "ERI INH" SET.
1980          ;#ADDRESS OF THE SECOND MODULE IS IN LOCATION "IBS2" VECTOR
1981          ;#ADDRESS IS IN LOCATION "VECTA2". THE SECOND IBV-11 SHOULD BE ELECTRICALLY SEC
1982          ;#TO INHIBIT THE USE OF TESTING WITH A SECOND MODULE, MAKE
1983          ;#LOCATION "SCDW1" ZERO.
1984          ;#
1985
1986 006502 005077 172640      CLR      @IBS          ;CLEAR CSRS.
1987 006506 005077 172654      CLR      @IBS2
1988 006512 052777 000004 172646  BIS      @BIT2,@IBS2  ;ASSERT REN ON IB BUS FROM 2ND
1989                                ;MODULE. 1ST IBV-11 SHOULD
1990 006520 032777 020000 172620  BIT      @BIT13,@IBS  ;GENERATE AN ERROR 1;DID IT??
1991 006526 001001                                BNE     TST50          ;
1992
          ;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

1996
1997 006530 104001      ERROR 1          ;/MODULE FAULT DETECTED:
1998                                ;FAILED TO GENERATE AN ERROR 1.

          ;;SSSSSSSSSS††† ERROR †††SSSSSSSSSS

2001
2002          ::*****
2003          ;#TEST 50      *TEST THAT AN ERROR 1 CAN GENERATE AN INTERRUPT
2004          ;#*****
2005 006532 000004      †T50: SCOPE
2006
2007          ;#WARNING! THIS TEST IS DESIGNED TO BE EXERCISED WITH A
2008          ;#SECOND MODULE (IBV-11) WITH SWITCH "ERI INH" SET.
2009          ;#ADDRESS OF THE SECOND MODULE IS IN LOCATION "IBS2" VECTOR
2010          ;#ADDRESS IS IN LOCATION "VECTA2". THE SECOND IBV-11 SHOULD BE ELECTRICALLY SEC
2011          ;#TO INHIBIT THE USE OF TESTING WITH A SECOND MODULE, MAKE
2012          ;#LOCATION "SCDW1" ZERO.
2013          ;#
2014
2015 006534 005077 172626      CLR      @IBS2        ;CLEAR CSRS.
2016 006540 005077 172602      CLR      @IBS
2017
2018 006544 012777 000200 172630  MOV      @200,@PRA
2019 006552 012777 006616 172576  MOV      @18,@VECTA  ;SET UP VECTOR ADDR.
2020
2021 006560 052777 000100 172560  BIS      @BIT06,@IBS ;SET INTERRUPT ENABLE
2022                                ;/PR
2023 006566 012746 000000                                ;/SET CPU PRIORITY ON RETURN
2024 006572 012746 006600                                ;/SHOW RETURN ADDRESS
2025 006576 000002                                ;/CAUSE A RETURN (PUTS NEW STATUS
2026 006600                                ;/IN STATUS REG.)
2027 006600 052777 000004 172560  BIS      @BIT2,@IBS2 ;GENERATE AN ERROR 1 AS PER LAST TEST.
2028 006606 000240      NOP

```

645:

MAINDEC-11-DVIBA-A
DVIBA.P11 T50

MACY11 27(663) 29-MAR-77 12:57 PAGE 39
*TEST THAT AN ERROR 1 CAN GENERATE AN INTERRUPT

SEQ 0054

2029 006610 000240
2030

NOP

:::SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

2034 006612 104001
2035
2036

ERROR 1 ;/MODULE FAULT DETECTED:
;ERROR 1 FAILED TO GENERATE AN INTR.

:::SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS

2039 006614 000402
2040

BR 25

2041 006616 062706 000004
2042 006616
2043 006622

15: ADD #4,SP ;/ADD #4 TO STACK POINTER.
25:

2045 006622 013777 001402 172526
2046 006630 012777 004700 172544
2047 006636 005077 172524
2048 006642 005077 172500

;/-RESV-
MOV PRA, VECTA ;/RESTORE VECTOR FOR
MOV #4700, PRA ;/ILLEGAL INTRO.
CLR JIBS2 ;CLEAR CSRS.
CLR JIBS

2049
2050

:::*****
; *TEST 51 *TEST THAT DATA CAN BE XFERRED BETWEEN THE MODULE UNDER TEST AND THE KGM
; * NOTE: KGM = KNOWN GOOD MODULE
; * IN THIS TEST WE'LL MAKE THE KGM A LISTENER
; * AND THE MODULE UNDER TEST A TALKER.
; * WE'VE ALREADY XFERRED DATA TO AND FROM THE IB-BUS
; * VIA THE MODULE UNDER TEST. THE ONLY UNKNOWN
; * IS THE CABLE CONNECTING THE KGM TO THE MODULE UNDER TEST,
; * AS WELL AS THE KGM.
; *

2051
2052
2053
2054
2055
2056
2057
2058
2059
2060

:::*****

2061 006646 000004
2062

↑T51: SCOPE

2064 006650 012737 006670 001110
2065 006656 012737 000000 001124
2066 006664 005037 001126

MOV #15, \$LPERR ;SET ERROR LOOP.
MOV #0, \$GDDAT ;START PATTERN.
CLR \$B0DAT

2068 006670 005077 172452
2069 006674 005077 172466
2070 006700 052777 000041 172440

15: CLR JIBS ;CLEAR CSRS.
CLR JIBS2
BIS #BITS, BIT0, JIBS ;SET TON AND TCS.
BIS #BIT4, JIBS2 ;SET LON ON KGM.
MOV \$GDDAT, JIB0 ;SEND PATTERN.
MOVB JIB02, \$B0DAT ;READ ATA FROM KGM.
CMPB \$GDDAT, \$B0DAT ;DATA SENT = DATA RECEIVED?
BEQ 25 ;YES, CONTINUE

2071 006706 052777 000020 172452
2072 006714 013777 001124 172426
2073 006722 117737 172442 001126
2074 006730 123737 001124 001126
2075 006736 001402

:::SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

2080 006740 104004
2081
2082

ERROR 4 ;/MODULE FAULT DETECTED:
;ERROR - BAD DATA PASSED BETWEEN

2083

;MODULE UNDER TEST AND KGM.

```

2086 006742 000407          BR      TST52          ;;
2087                                ;;
2088 006744 105237 001124    25:    INCB   $CDOAT      ;CHANGE PATTERN.
2089 006750 001347          BNE     IS           ;IF NOT DONE, CONTINUE.
2090
2091 006752 005077 172370    CLR     @IBS        ;CLEAR CSR'S
2092 006756 005077 172404    CLR     @IBS2
2093
2094                                ;*****
2095                                ;*TEST 52      *TEMP END OF TESTS
2096                                ;*****
2097 006762 000004          TST52: SCOPE
2098
2099 006764          EOP:
2100
2101                                .SBTTL  SYSMAC ROUTINES:
2102
2103                                .SBTTL  END OF PASS ROUTINE
2104
2105                                ;*****
2106                                ;*INCREMENT THE PASS NUMBER ($PASS)
2107                                ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
2108                                ;*IF THERES A MONITOR GO TO IT
2109                                ;*IF THERE ISN'T JUMP TO RSTART
2110
2111          SEOP:
2112          SCOPE
2113          CLR     $STSTM          ;; ZERO THE TEST NUMBER
2114          CLR     $TIMES         ;; ZERO THE NUMBER OF ITERATIONS
2115          INC     $PASS          ;; INCREMENT THE PASS NUMBER
2116          BIC     @10000,$PASS  ;; DON'T ALLOW A NEG. NUMBER
2117          DEC     (PC)+         ;; LOOP?
2118          SEOPCT: .WORD 1
2119          BGT     $DOAGN         ;; YES
2120          MOV     (PC)+,@(PC)+   ;; RESTORE COUNTER
2121          SENDCT: .WORD 1
2122          SEOPCT
2123          TYPE    $SENDMG        ;; TYPE "END PASS #"
2124          MOV     $PASS,-(SP)    ;; SAVE $PASS FOR TYPEOUT
2125          TYPDS   TYPE          ;; GO TYPE--DECIMAL ASCII WITH SIGN
2126          TYPE    $NULL         ;; TYPE A NULL CHARACTER
2127          MOV     @42,R0        ;; GET MONITOR ADDRESS
2128          BEQ    $DOAGN        ;; BRANCH IF NO MONITOR
2129          RESET  $DOAGN        ;; CLEAR THE WORLD
2130          JSR    PC,(R0)        ;; GO TO MONITOR
2131          NOP
2132          NOP
2133          NOP
2134          SDOAGN:
2135          JMP     @(PC)+         ;; RETURN
2136          SRTNAD: .WORD RSTART

```

```

2137 007066 377 377 000 $ENULL: .BYTE -1,-1,0 ;:NULL CHARACTER STRING
2138 007071 015 042412 042116 $ENDMG: .ASCIZ <15><12>/END PASS #/
2139 007076 050040 051501 020123
2140 007104 000043
2141 ;/DELMA
2142
2143 ;/ROUTINE TO PROVIDE DELAYS IN INCREMENTS OF 25 US
2144 ;/
2145 ;/ CALL= JSR RS,DEL50
2146 ;/ WORD X (# OF 25 US TO DELAY)
2147 ;/ RETURNS HERE
2148 ;/
2149
2150 007106 012500 DEL50: MOV (5)+,R0 ;/GET # OF 25 US DELAYS
2151 007110 012701 000007 15: MOV #2.,R1 ;/# FOR LOOP TO DO 50 US.
2152 007114 005301 25: DEC R1 ;/DEC IT
2153 007116 001376 BNE 25 ;/WAITED 25. TIMES?
2154 007120 005300 DEC R0 ;/DONE # OF 50 US DELAY DESIRED?
2155 007122 001372 BNE 15 ;/NO - NEXT ONE.
2156 007124 000205 RTS R5 ;/YES - EXIT.
2157
2158 .SBTTL ERROR HANDLER ROUTINE
2159
2160 ;:*****
2161 ;:THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2162 ;:SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2163 ;:AND GO TO $ERRTYP ON ERROR
2164 ;:THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2165 ;:*SW15=1 HALT ON ERROR
2166 ;:*SW13=1 INHIBIT ERROR TYPEOUTS
2167 ;:*SW10=1 BELL ON ERROR
2168 ;:*SW09=1 LOOP ON ERROR
2169 ;:*CALL
2170 ;:* ERROR N ;:ERROR=EMT AND N=ERROR ITEM NUMBER
2171
2172 007126 $ERROR:
2173 007126 104407 CKSWR ;:TEST FOR CHANGE IN SOFT-SWR
2174 007130 105237 001103 75: INCB $ERFLG ;:SET THE ERROR FLAG
2175 007134 001775 BEQ 75 ;:DON'T LET THE FLAG GO TO ZERO
2176 007136 013777 001102 171776 MOV $STNM,$DISPLAY ;:DISPLAY TEST NUMBER AND ERROR FLAG
2177 007144 032777 002000 171766 BIT #BIT10,$SWR ;:BELL ON ERROR?
2178 007152 001402 BEQ 15 ;:NO - SKIP
2179 007154 104401 001164 TYPE $BELL ;:RING BELL
2180 007160 005237 001112 15: INC $ERTTL ;:COUNT THE NUMBER OF ERRORS
2181 007164 011637 001116 MOV (SP),$ERRPC ;:GET ADDRESS OF ERROR INSTRUCTION
2182 007170 162737 000002 001116 SUB #2,$ERRPC
2183 007176 117737 171714 001114 MOVB $ERRPC,$ITEMB ;:STRIP AND SAVE THE ERROR ITEM CODE
2184 007204 032777 020000 171726 BIT #BIT13,$SWR ;:SKIP TYPEOUT IF SET
2185 007212 001004 BNE 205 ;:SKIP TYPEOUTS
2186 007214 004737 007314 JSR PC,$ERRTYP ;:GO TO USER ERROR ROUTINE
2187 007220 104401 001171 TYPE $CRLF
2188 007224
2189 007224 122737 000001 001214 205: CMPB #APTENV,$ENV ;:RUNNING IN APT MODE
2190 007232 001007 BNE 25 ;:NO,SKIP APT ERROR REPORT

```

```

2191 007234 113737 001114 007246      MOVB    $ITEMB,21$      ;; SET ITEM NUMBER AS ERROR NUMBER
2192 007242 004737 011460              JSR     PC,$ATY4       ;; REPORT FATAL ERROR TO APT
2193 007246      000          21$: .BYTE    0
2194 007247      000          .BYTE    0
2195 007250 000777              22$: BR     22$          ;; APT ERROR LOOP
2196 007252 005777 171662          2$: TST    2$SWR       ;; HALT ON ERROR
2197 007256 100002          BPL     3$            ;; SKIP IF CONTINUE
2198 007260 000000          HALT                    ;; HALT ON ERROR!
2199 007262 104407          CKSWR                    ;; TEST FOR CHANGE IN SOFT-SWR
2200 007264 032777 001000 171646 3$: BIT    2$BIT09,2$SWR  ;; LOOP ON ERROR SWITCH SET?
2201 007272 001402          BEQ    4$            ;; BR IF NO
2202 007274 013716 001110          MOV    $LPCRR,(SP)     ;; FUDGE RETURN FOR LOOPING
2203 007300 005737 001162          4$: TST    $ESCAPE      ;; CHECK FOR AN ESCAPE ADDRESS
2204 007304 001402          BEQ    5$            ;; BR IF NONE
2205 007306 013716 001162          MOV    $ESCAPE,(SP)   ;; FUDGE RETURN ADDRESS FOR ESCAPE
2206 007312
2207 007312 000002          5$: RTI                    ;; RETURN
2208 .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
2209
2210 ;; *****
2211 ;; THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
2212 ;; ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
2213 ;; AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
2214
2215 007314      SERRTYP:
2216 007314 104401 001171          TYPE   $CRLF          ;; "CARRIAGE RETURN" & "LINE FEED"
2217 007320 010046          MOV    R0,-(SP)       ;; SAVE R0
2218 007322 005000          CLR    R0            ;; PICKUP THE ITEM INDEX
2219 007324 153700 001114          BISB  2$ITEMB,R0
2220 007330 001004          BNE    1$
2221
2222 007332 013746 001116          MOV    $ERRPC,-(SP)   ;; IF ITEM NUMBER IS ZERO, JUST
2223
2224 007336 104402          TYPOC                    ;; TYPE THE PC OF THE ERROR
2225 007340 000426          BR     6$            ;; SAVE $ERRPC FOR TYPEOUT
2226 007342 005300          1$: DEC    R0          ;; ERROR ADDRESS
2227 007344 006300          ASL   R0            ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
2228 007346 006300          ASL   R0            ;; GET OUT
2229 007350 006300          ASL   R0            ;; ADJUST THE INDEX SO THAT IT WILL
2230 007352 062700 001256          ADD    2$ERRTB,R0     ;; WORK FOR THE ERROR TABLE
2231 007356 012037 007366          MOV    (R0)+,2$
2232 007362 001404          BEQ    3$
2233 007364 104401          TYPE                    ;; FORM TABLE POINTER
2234 007366 000000          2$: .WORD  0          ;; PICKUP "ERROR MESSAGE" POINTER
2235 007370 104401 001171          TYPE   $CRLF          ;; SKIP TYPEOUT IF NO POINTER
2236 007374 012037 007404          3$: MOV    (R0)+,4$     ;; TYPE THE "ERROR MESSAGE"
2237 007400 001404          BEQ    5$            ;; "ERROR MESSAGE" POINTER GOES HERE
2238 007402 104401          TYPE                    ;; "CARRIAGE RETURN" & "LINE FEED"
2239 007404 000000          4$: .WORD  0          ;; PICKUP "DATA HEADER" POINTER
2240 007406 104401 001171          TYPE   $CRLF          ;; SKIP TYPEOUT IF 0
2241 007412 011000          5$: MOV    (R0),R0     ;; TYPE THE "DATA HEADER"
2242 007414 001004          BNE    7$            ;; "DATA HEADER" POINTER GOES HERE
2243 007416 012600          6$: MOV    (SP)+,R0    ;; "CARRIAGE RETURN" & "LINE FEED"
2244 007420 104401 001171          TYPE   ,CRLF
    
```

```

2245 007424 000207          RTS      PC          ;;RETURN
2246 007426                7$:      MOV      2(R0)+,-(SP)    ;;SAVE 2(R0)+ FOR TYPEOUT
2247 007426 013046          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2248 007430 104402          TST      (R0)        ;;IS THERE ANOTHER NUMBER?
2249 007432 005710          BEQ      6$          ;;BR IF NO
2250 007434 001770          TYPE     6$          ;;TYPE TWO(2) SPACES
2251 007436 104401 007444          BR      7$          ;;LOOP
2252 007442 000771          .ASCIZ  / /          ;;TWO(2) SPACES
2253 007444 020040 000          .EVEN
2254 007450          .SBTTL SCOPE HANDLER ROUTINE
2255
2256
2257          ;*****
2258          ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
2259          ;AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
2260          ;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
2261          ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2262          ;*SW14=1 LOOP ON TEST
2263          ;*SW11=1 INHIBIT ITERATIONS
2264          ;*SW09=1 LOOP ON ERROR
2265          ;*SW08=1 LOOP ON TEST IN SWR<7:0>
2266          ;*CALL
2267          ;* SCOPE          ;;SCOPE=IOT
2268
2269          $SCOPE:
2270 007450 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
2271 007452 104407          CKSWR
2272 007454 032777 040000 171456 1$: BIT      #BIT14,2SWR    ;;LOOP ON PRESENT TEST?
2273 007462 001114          BNE      $OVER     ;;YES IF SW14=1
2274          ;****START OF CODE FOR THE XOR TESTER****
2275 007464 000416          $XTSTR: BR      6$    ;;IF RUNNING ON THE "XOR" TESTER CHANGE
2276          ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
2277 007466 013746 000004          MOV      2#ERRVEC, -(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
2278 007472 012737 007512 000004          MOV      #55,2#ERRVEC   ;;SET FOR TIMEOUT
2279 007500 005737 177060          TST      2#177060      ;;TIME OUT ON XOR?
2280 007504 012637 000004          MOV      (SP)+,2#ERRVEC ;;RESTORE THE ERROR VECTOR
2281 007510 000463          BR      $$VLAD       ;;GO TO THE NEXT TEST
2282 007512 022626          5$: CMP      (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
2283 007514 012637 000004          MOV      (SP)+,2#ERRVEC ;;RESTORE THE ERROR VECTOR
2284 007520 000423          BR      7$          ;;LOOP ON THE PRESENT TEST
2285 007522          6$;****END OF CODE FOR THE XOR TESTER****
2286 007522 032777 000400 171410          BIT      #BIT08,2SWR    ;;LOOP ON SPEC. TEST?
2287 007530 001404          BEQ      2$          ;;BR IF NO
2288 007532 127737 171402 001102          CMPB    2SWR,$TSTNM    ;;ON THE RIGHT TEST? SWR<7:0>
2289 007540 001465          BEQ      $OVER     ;;BR IF YES
2290 007542 105737 001103          2$: TSTB    $ERFLG      ;;HAS AN ERROR OCCURRED?
2291 007546 001421          BEQ      3$          ;;BR IF NO
2292 007550 123737 001115 001103          CMPB    $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
2293 007556 101015          BHI      3$          ;;BR IF NO
2294 007560 032777 001000 171352          BIT      #BIT09,2SWR    ;;LOOP ON ERROR?
2295 007566 001404          BEQ      4$          ;;BR IF NO
2296 007570 013737 001110 001106          7$: MOV      $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
2297 007576 000446          BR      $OVER     ;;
2298 007600 105037 001103          4$: CLRB    $ERFLG    ;;ZERO THE ERROR FLAG

```

```

2299 007604 005037 001160 CLR $TIMES ;; CLEAR THE NUMBER OF ITERATIONS TO MAKE
2300 007610 000415 BR $S ;; ESCAPE TO THE NEXT TEST
2301 007612 032777 004000 171320 3$: BIT #BIT11,$SWR ;; INHIBIT ITERATIONS?
2302 007620 001011 BNE $S ;; BR IF YES
2303 007622 005737 001202 TST $PASS ;; IF FIRST PASS OF PROGRAM
2304 007626 001406 BEQ $S ;; INHIBIT ITERATIONS
2305 007630 005237 001104 INC $ICNT ;; INCREMENT ITERATION COUNT
2306 007634 023737 001160 001104 CMP $TIMES,$ICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
2307 007642 002024 BGE $OVER ;; BR IF MORE ITERATION REQUIRED
2308 007644 012737 000001 001104 1$: MOV #1,$ICNT ;; REINITIALIZE THE ITERATION COUNTER
2309 007652 013737 007730 001160 MOV $MXCNT,$TIMES ;; SET NUMBER OF ITERATIONS TO DC
2310 007660 105237 001102 $SVLAD: INCB $STNM ;; COUNT TEST NUMBERS
2311 007664 113737 001102 001200 MOVB $STNM,$STNM ;; SET TEST NUMBER IN APT MAILBOX
2312 007672 011637 001106 MOV (SP),$LPAOR ;; SAVE SCOPE LOOP ADDRESS
2313 007676 011637 001110 MOV (SP),$LPERR ;; SAVE ERROR LOOP ADDRESS
2314 007702 005037 001162 CLR $ESCAPE ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
2315 007706 112737 000001 001115 MOVB #1,$ERMAX ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
2316 007714 013777 001102 171220 $OVER: MOV $STNM,$DISPLAY ;; DISPLAY TEST NUMBER
2317 007722 013716 001106 MOV $LPAOR,(SP) ;; FUDGE RETURN ADDRESS
2318 007726 000002 RTI ;; FIXES PS
2319 007730 003720 $MXCNT: 2000. ;; MAX. NUMBER OF ITERATIONS
2320 .SBTTL TTY INPUT ROUTINE
2321
2322 ;;*****
2323 .ENABL LSB
2324
2325 ;;*****
2326 ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
2327 ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
2328 ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
2329 ;*WHEN OPERATING IN TTY FLAG MODE.
2330 007732 022737 000176 001140 $CKSWR: CMP #SWREG,$SWR ;; IS THE SOFT-SWR SELECTED?
2331 007740 001074 BNE $S ;; BRANCH IF NO
2332 007742 105777 171176 TSTB $STKS ;; CHAR THERE?
2333 007746 100071 BPL $S ;; IF NO, DON'T WAIT AROUND
2334 007750 117746 171172 MOVB $STKB,-(SP) ;; SAVE THE CHAR
2335 007754 042716 177600 BIC #177,(SP) ;; STRIP-OFF THE ASCII
2336 007760 022726 000007 CMP #7,(SP)+ ;; IS IT A CONTROL G?
2337 007764 001062 BNE $S ;; NO, RETURN TO USER
2338 007766 123727 001134 000001 CMPB $AUTOB,#1 ;; ARE WE RUNNING IN AUTO-MODE?
2339 007774 001456 BEQ $S ;; BRANCH IF YES
2340
2341 007776 104401 010457 $GTSWR: TYPE $CNTLG ;; ECHO THE CONTROL-G (?G)
2342 010002 104401 010464 TYPE $MSWR ;; TYPE CURRENT CONTENTS
2343 010006 013746 000176 MOV $SWREG,-(SP) ;; SAVE SWREG FOR TYPEOUT
2344 010012 104402 TYPEOC ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
2345 010014 104401 010475 TYPE $MNEW ;; PROMPT FOR NEW SWR
2346 010020 005046 19$: CLR -(SP) ;; CLEAR COUNTER
2347 010022 005046 CLR -(SP) ;; THE NEW SWR
2348 010024 105777 171114 7$: TSTB $STKS ;; CHAR THERE?
2349 010030 100375 BPL 7$ ;; IF NOT TRY AGAIN
2350
2351 010032 117746 171110 MOVB $STKB,-(SP) ;; PICK UP CHAR
2352 010036 042716 177600 BIC #177,(SP) ;; MAKE IT 7-BIT ASCII

```

```

2353
2354
2355
2356 010042 021627 000025      9$:    CMP      (SP),#25      ;; IS IT A CONTROL-U?
2357 010046 001005                BNE      10$          ;; BRANCH IF NOT
2358 010050 104401 010452          TYPE    $CNTLU      ;; YES, ECHO CONTROL-U (IU)
2359 010054 062706 000006      20$:    ADD      #6,SP      ;; IGNORE PREVIOUS INPUT
2360 010060 000757                BR       19$          ;; LET'S TRY IT AGAIN
2361
2362
2363 010062 021627 000015      10$:    CMP      (SP),#15      ;; IS IT A <CR>?
2364 010066 001022                BNE      16$          ;; BRANCH IF NO
2365 010070 005766 000004          TST     4(SP)        ;; YES, IS IT THE FIRST CHAR?
2366 010074 001403                BEQ     11$          ;; BRANCH IF YES
2367 010076 016677 000002 171034      MOV     2(SP),@SWR   ;; SAVE NEW SWR
2368 010104 062706 000006      11$:    ADD      #6,SP      ;; CLEAR UP STACK
2369 010110 104401 001171      14$:    TYPE    $CRLF      ;; ECHO <CR> AND <LF>
2370 010114 123727 001135 000001      CMPB   $INTAG,#1    ;; RE-ENABLE TTY KBD INTERRUPTS?
2371 010122 001003                BNE     15$          ;; BRANCH IF NOT
2372 010124 012777 000100 171012      MOV     #100,@STKS  ;; RE-ENABLE TTY KBD INTERRUPTS
2373 010132 000002                RTI                    ;; RETURN
2374 010134 004737 011372      15$:    JSR     PC,$TYPEC  ;; ECHO CHAR
2375 010140 021627 000060      16$:    CMP     (SP),#60   ;; CHAR < 0?
2376 010144 002420                BLT     18$          ;; BRANCH IF YES
2377 010146 021627 000067      CMP     (SP),#67   ;; CHAR > 7?
2378 010152 003015                BGT     18$          ;; BRANCH IF YES
2379 010154 042726 000060      BIC     #60,(SP)+  ;; STRIP-OFF ASCII
2380 010160 005766 000002          TST     2(SP)        ;; IS THIS THE FIRST CHAR
2381 010164 001403                BEQ     17$          ;; BRANCH IF YES
2382 010166 006316                ASL     (SP)        ;; NO, SHIFT PRESENT
2383 010170 006316                ASL     (SP)        ;; CHAR OVER TO MAKE
2384 010172 006316                ASL     (SP)        ;; ROOM FOR NEW ONE.
2385 010174 005266 000002      17$:    INC     2(SP)        ;; KEEP COUNT OF CHAR
2386 010200 056616 177776      BIS     -2(SP),(SP) ;; SET IN NEW CHAR
2387 010204 000707                BR      7$           ;; GET THE NEXT ONE
2388 010206 104401 001170      18$:    TYPE    $QUES      ;; TYPE ?<CR><LF>
2389 010212 000720                BR      20$          ;; SIMULATE CONTROL-U
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401 010214 011646 000004 000002      $R0CHR: MOV     (SP),-(SP) ;; PUSH DOWN THE PC
2402 010216 016666 000004 170714      MOV     4(SP),2(SP) ;; SAVE THE PS
2403 010224 105777 170714      1$:    TSTB   @STKS      ;; WAIT FOR
2404 010230 100375                BPL     1$           ;; A CHARACTER
2405 010232 117766 170710 000004      MOVB   @STKB,4(SP) ;; READ THE TTY
2406 010240 042766 177600 000004      BIC     #1C<177>,4(SP) ;; GET RID OF JUNK IF ANY

```

```

*****
; THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
; CALL:
;   R0CHR
;   RETURN HERE
; INPUT A SINGLE CHARACTER FROM THE TTY
; CHARACTER IS ON THE STACK
; WITH PARITY BIT STRIPPED OFF

```

```

2407 010246 026627 000004 000023      CMP      4(SP),#23      ;; IS IT A CONTROL-5?
2408 010254 001013                    BNE      3$           ;; BRANCH IF NO
2409 010256 105777 170662      2$:    TSTB     2$TKS     ;; WAIT FOR A CHARACTER
2410 010262 100375                    BPL      2$           ;; LOOP UNTIL ITS THERE
2411 010264 117746 170656      MOV      2$TKB,-(SP)   ;; GET CHARACTER
2412 010270 042716 177600      BIC      8+C177,(SP)  ;; MAKE IT 7-BIT ASCII
2413 010274 022627 000021      CMP      (SP)+,#21   ;; IS IT A CONTROL-0?
2414 010300 001366                    BNE      2$           ;; IF NOT DISCARD IT
2415 010302 000750                    BR       1$           ;; YES, RESUME
2416 010304 026627 000004 000140 3$:    CMP      4(SP),#140  ;; IS IT UPPER CASE?
2417 010312 002407                    BLT      4$           ;; BRANCH IF YES
2418 010314 026627 000004 000175      CMP      4(SP),#175  ;; IS IT A SPECIAL CHAR?
2419 010322 003003                    BGT      4$           ;; BRANCH IF YES
2420 010324 042766 000040 000004      BIC      #40,4(SP)   ;; MAKE IT UPPER CASE
2421 010332 000002      4$:    RTI              ;; GO BACK TO USER
2422                                     *****
2423                                     *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2424                                     *CALL:
2425                                     *
2426                                     *   RDLIN
2427                                     *   RETURN HERE
2428                                     *
2429                                     ;; INPUT A STRING FROM THE TTY
2430                                     ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2431                                     ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
2432
2433 $RDLIN: MOV      R3,-(SP)      ;; SAVE R3
2434 1$:    MOV      #1$TTYIN,R3  ;; GET ADDRESS
2435 2$:    CMP      #1$TTYIN+8.,R3 ;; BUFFER FULL?
2436                    BLOS     4$           ;; BR IF YES
2437                    ROCHR    ;; GO READ ONE CHARACTER FROM THE TTY
2438                    MOV      (SP)+,(R3)  ;; GET CHARACTER
2439 10$:   CMP      #177,(R3)    ;; IS IT A RUBOUT
2440                    BNE      3$           ;; SKIP IF NOT
2441                    TYPE     $QUES    ;; TYPE A '?'
2442                    BR       1$           ;; CLEAR THE BUFFER AND LOOP
2443 3$:    MOV      (R3),9$      ;; ECHO THE CHARACTER
2444                    TYPE     9$
2445                    CMP      #15,(R3)+  ;; CHECK FOR RETURN
2446                    BNE      2$           ;; LOOP IF NOT RETURN
2447                    CLRB    -1(R3)     ;; CLEAR RETURN (THE 15)
2448                    TYPE     $LF      ;; TYPE A LINE FEED
2449                    MOV      (SP)+,R3  ;; RESTORE R3
2450                    MOV      (SP),-(SP) ;; ADJUST THE STACK AND PUT ADDRESS OF THE
2451                    MOV      4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
2452                    RTI
2453 9$:    .BYTE   0              ;; RETURN
2454                    .BYTE   0              ;; STORAGE FOR ASCII CHAR. TO TYPE
2455                    .BLKB   8.           ;; TERMINATOR
2456                    $TTYIN: .ASCIZ /U/<15><12> ;; RESERVE 8 BYTES FOR TTY INPUT
2457                    $CNTLU: .ASCIZ /G/<15><12> ;; CONTROL "U"
2458                    $MSWR:  .ASCIZ <15><12>/SWR = / ;; CONTROL "G"
2459
2460                    $MNEW:  .ASCIZ / NEW = /
2461
2462                    .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

```

```

2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484 010506 017646 000000
2485 010512 116637 000001 010731
2486 010520 112637 010733
2487 010524 062716 000002
2488 010530 000406
2489 010532 112737 000001 010731
2490 010540 112737 000006 010733
2491 010546 112737 000005 010730
2492 010554 010346
2493 010556 010446
2494 010560 010546
2495 010562 113704 010733
2496 010566 005404
2497 010570 062704 000006
2498 010574 110437 010732
2499 010600 113704 010731
2500 010604 016605 000012
2501 010610 005003
2502 010612 006105 1$:
2503 010614 000404 BR 3$
2504 010616 006105 2$:
2505 010620 006105 ROL 5$
2506 010622 006105 ROL 5$
2507 010624 010503 ROL 5$
2508 010626 006103 3$:
2509 010630 105337 010732 DECB 7$
2510 010634 100016 BPL 7$
2511 010636 042703 177770 BIC #177770,R3
2512 010642 001002 BNE 4$
2513 010644 005704 TST R4
2514 010646 001403 BEQ 5$

```

```

*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*STYPOS---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
*
*STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*STYPOS OR STYPOC
*CALL:
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPON   ;;CALL FOR TYPEOUT
*
*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
*CALL:
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPOC   ;;CALL FOR TYPEOUT
*
*STYPOS: MOV     2(SP),-(SP)  ;;PICKUP THE MODE
*         MOVB   1(SP),SOFILL ;;LOAD ZERO FILL SWITCH
*         MOVB   (SP)+,SOMODE+1 ;;NUMBER OF DIGITS TO TYPE
*         ADD    #2,(SP)     ;;ADJUST RETURN ADDRESS
*         BR     STYPON
*STYPOC: MOVB   #1,SOFILL    ;;SET THE ZERO FILL SWITCH
*         MOVB   #6,SOMODE+1 ;;SET FOR SIX(6) DIGITS
*STYPON: 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   SOFILL,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    3$        ;;GO DO MSB
*         ROL   R5        ;;FORM THIS DIGIT
*         ROL   R5
*         ROL   R5
*         MOV   R5,R3
*         ROL   R3        ;;GET LSB OF THIS DIGIT
*         DECB  SOMODE    ;;TYPE THIS DIGIT?
*         BR   7$        ;;BR IF NO
*         BIC  #177770,R3 ;;GET RID OF JUNK
*         BNE  4$        ;;TEST FOR 0
*         TST  R4        ;;SUPPRESS THIS 0?
*         BEQ  5$        ;;BR IF YES

```

| | | | | | | | |
|------|--------|--------|---------------|----------|-------|-------------|------------------------------------|
| 2515 | 010650 | 005204 | | 4\$: | INC | R4 | :: DON'T SUPPRESS ANYMORE 0'S |
| 2516 | 010652 | 052703 | 000060 | | BIS | #'0,R3 | :: MAKE THIS DIGIT ASCII |
| 2517 | 010656 | 052703 | 000040 | 5\$: | BIS | #' R3 | :: MAKE ASCII IF NOT ALREADY |
| 2518 | 010662 | 110337 | 010726 | | MOVB | R3,B\$ | :: SAVE FOR TYPING |
| 2519 | 010666 | 104401 | 010726 | | TYPE | B\$ | :: GO TYPE THIS DIGIT |
| 2520 | 010672 | 105337 | 010730 | 7\$: | DECB | \$OCNT | :: COUNT BY 1 |
| 2521 | 010676 | 003347 | | | BGT | 2\$ | :: BR IF MORE TO DO |
| 2522 | 010700 | 002402 | | | BLT | 6\$ | :: BR IF DONE |
| 2523 | 010702 | 005204 | | | INC | R4 | :: INSURE LAST DIGIT ISN'T A BLANK |
| 2524 | 010704 | 000744 | | | BR | 2\$ | :: GO DO THE LAST DIGIT |
| 2525 | 010706 | 012605 | | 6\$: | MOV | (SP)+,R5 | :: RESTORE R5 |
| 2526 | 010710 | 012604 | | | MOV | (SP)+,R4 | :: RESTORE R4 |
| 2527 | 010712 | 012603 | | | MOV | (SP)+,R3 | :: RESTORE R3 |
| 2528 | 010714 | 016666 | 000002 000004 | | MOV | 2(SP),4(SP) | :: SET THE STACK FOR RETURNING |
| 2529 | 010722 | 012616 | | | MOV | (SP)+,(SP) | |
| 2530 | 010724 | 000002 | | | RTI | | :: RETURN |
| 2531 | 010726 | 000 | | 8\$: | .BYTE | 0 | :: STORAGE FOR ASCII DIGIT |
| 2532 | 010727 | 000 | | | .BYTE | 0 | :: TERMINATOR FOR TYPE ROUTINE |
| 2533 | 010730 | 000 | | \$OCNT: | .BYTE | 0 | :: OCTAL DIGIT COUNTER |
| 2534 | 010731 | 000 | | \$OFILL: | .BYTE | 0 | :: ZERO FILL SWITCH |
| 2535 | 010732 | 000000 | | \$OMODE: | .WORD | 0 | :: NUMBER OF DIGITS TO TYPE |

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589

010734
010734 010046
010736 010146
010740 010246
010742 010346
010744 010546
010746 012746 020200
010752 016605 000020
010756 100004
010760 005405
010762 112766 000055 000001
010770 005000
010772 012703 011150
010776 112723 000040
011002 005002
011004 016001 011140
011010 160105
011012 002402
011014 005202
011016 000774
011020 060105
011022 005702
011024 001002
011026 105716
011030 100407
011032 106316
011034 103003
011036 116663 000001 177777
011044 052702 000060
011050 052702 000040
011054 110223
011056 005720
011060 020027 000010
011064 002746
011066 003002
011070 019502
011072 000764
011074 105726
011076 100003
011100 116663 177777 177776
011106 105013
011110 012605

```
*****  
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT  
*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE  
*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED  
*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE  
*REPLACED WITH SPACES.  
*CALL:  
*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK  
*      TYPDS                    ;;GO TO THE ROUTINE  
*  
STYPDS:  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      R5,-(SP)      ;;PUSH R5 ON STACK  
          MOV      #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN  
          MOV      20(SP),R5     ;;GET THE INPUT NUMBER  
          BPL      1$           ;;BR IF INPUT IS POS.  
          NEG      R5           ;;MAKE THE BINARY NUMBER POS.  
          MOV      #'-,1(SP)    ;;MAKE THE ASCII NUMBER NEG.  
          CLR      R0           ;;ZERO THE CONSTANTS INDEX  
          MOV      #S0BLK,R3    ;;SETUP THE OUTPUT POINTER  
          MOV      #' ,(R3)+    ;;SET THE FIRST CHARACTER TO A BLANK  
          CLR      R2           ;;CLEAR THE BCD NUMBER  
          MOV      $OTBL(R0),R1  ;;GET THE CONSTANT  
          SUB      R1,R5        ;;FORM THIS BCD DIGIT  
          BLT      4$           ;;BR IF DONE  
          INC      R2           ;;INCREASE THE BCD DIGIT BY 1  
          BR      3$  
          ADD      R1,R5        ;;ADD BACK THE CONSTANT  
          TST      R2           ;;CHECK IF BCD DIGIT=0  
          BNE      5$           ;;FALL THROUGH IF 0  
          TSTB    (SP)         ;;STILL DOING LEADING 0'S?  
          BMI      7$           ;;BR IF YES  
          ASLB    (SP)         ;;MSD?  
          BCC      6$           ;;BR IF NO  
          MOV      1(SP),-1(R3)  ;;YES--SET THE SIGN  
          BIS      #'0,R2       ;;MAKE THE BCD DIGIT ASCII  
          BIS      #' ,R2       ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT  
          MOV      R2,(R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER  
          TST      (R0)+       ;;JUST INCREMENTING  
          CMP      R0,#10      ;;CHECK THE TABLE INDEX  
          BLT      2$           ;;GO DO THE NEXT DIGIT  
          BGT      8$           ;;GO TO EXIT  
          MOV      R5,R2       ;;GET THE LSD  
          BR      6$           ;;GO CHANGE TO ASCII  
          TSTB    (SP)+       ;;WAS THE LSD THE FIRST NON-ZERO?  
          BPL      9$           ;;BR IF NO  
          MOV      -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING  
          CLR      (R3)        ;;SET THE TERMINATOR  
          MOV      (SP)+,R5     ;;POP STACK INTO R5  
1$:  
2$:  
3$:  
4$:  
5$:  
6$:  
7$:  
8$:  
9$:
```

| | | | | |
|------|--------|--------|--------|--------|
| 2600 | 011112 | 012603 | | |
| 2601 | 011114 | 012602 | | |
| 2602 | 011116 | 012601 | | |
| 2603 | 011120 | 012600 | | |
| 2604 | 011122 | 104401 | 011150 | |
| 2605 | 011126 | 016666 | 000002 | 000004 |
| 2606 | 011134 | 012616 | | |
| 2607 | 011136 | 000002 | | |
| 2608 | 011140 | 023420 | | |
| 2609 | 011142 | 001750 | | |
| 2610 | 011144 | 000144 | | |
| 2611 | 011146 | 000012 | | |
| 2612 | 011150 | 000004 | | |

```

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
TYPE $DBLK        ;; NOW TYPE THE NUMBER
MOV 2(SP),4(SP)   ;; ADJUST THE STACK
MOV (SP)+,(SP)
RTI                ;; RETURN TO USER

```

SDTBL: 10000.
1000.
100.
10.

SOBLK: .BLKW 4
.SETTL TYPE ROUTINE

```

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

| | | | | |
|------|--------|--------|--------|--------|
| 2620 | 011160 | 105737 | 001157 | |
| 2621 | 011164 | 100002 | | |
| 2622 | 011166 | 000000 | | |
| 2623 | 011170 | 000430 | | |
| 2624 | 011172 | 010046 | | |
| 2625 | 011174 | 017600 | 000002 | |
| 2626 | 011200 | 122737 | 000001 | 001214 |
| 2627 | 011206 | 001011 | | |
| 2628 | 011210 | 132737 | 000100 | 001215 |
| 2629 | 011216 | 001405 | | |
| 2630 | 011220 | 010037 | 011230 | |
| 2631 | 011224 | 004737 | 011450 | |
| 2632 | 011230 | 000000 | | |
| 2633 | 011232 | 132737 | 000040 | 001215 |
| 2634 | 011240 | 001003 | | |
| 2635 | 011242 | 112046 | | |
| 2636 | 011244 | 001005 | | |
| 2637 | 011246 | 005726 | | |
| 2638 | 011250 | 012600 | | |
| 2639 | 011252 | 062716 | 000002 | |
| 2640 | 011256 | 000002 | | |
| 2641 | 011260 | 122716 | 000011 | |
| 2642 | 011264 | 001430 | | |
| 2643 | 011266 | 122716 | 000200 | |

```

$TYPE: TSTB $TPFLG      ;; IS THERE A TERMINAL?
BPL 1$      ;; BR IF YES
HALT      ;; HALT HERE IF NO TERMINAL
BR 3$      ;; LEAVE
1$: MOV RO,-(SP)      ;; SAVE RO
MOV 22(SP),RO      ;; GET ADDRESS OF ASCIZ STRING
CMPB #APTENV,$ENV      ;; RUNNING IN APT MODE
BNE 62$      ;; NO GO CHECK FOR APT CONSOLE
BITB #APTPOOL,$ENV      ;; SPOOL MESSAGE TO APT
BEQ 62$      ;; NO GO CHECK FOR CONSOLE
MOV RO,61$      ;; SETUP MESSAGE ADDRESS FOR APT
JSR PC,$ATY3      ;; SPOOL MESSAGE TO APT
61$: .WORD 0      ;; MESSAGE ADDRESS
62$: BITB #APTCSUP,$ENV      ;; APT CONSOLE SUPPRESSED
BNE 60$      ;; YES, SKIP TYPE OUT
2$: MOVB (RO)+,-(SP)      ;; PUSH CHARACTER TO BE TYPED ONTO STACK
BNE 4$      ;; BR IF IT ISN'T THE TERMINATOR
TST (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
60$: MOV (SP)+,RO      ;; RESTORE RO
3$: ADD #2,(SP)      ;; ADJUST RETURN PC
RTI      ;; RETURN
4$: CMPB #HT,(SP)      ;; BRANCH IF <HT>
BEQ 8$
CMPB #CRLF,(SP)      ;; BRANCH IF NOT <CRLF>

```

```

2644 011272 001006      BNE      5$
2645 011274 005726      TST      (SP)+      ;; POP <CR><LF> EQUIV
2646 011276 104401      TYPE
2647 011300 001171      SCRLF      ;; TYPE A CR AND LF
2648 011302 105037 011436      CLR      $CHARCNT      ;; CLEAR CHARACTER COUNT
2649 011306 000755      BR      2$      ;; GET NEXT CHARACTER
2650 011310 004737 011372      5$: JSR      PC,$TYPEC      ;; GO TYPE THIS CHARACTER
2651 011314 123726 001156      6$: CMPB   $FILLC,(SP)+      ;; IS IT TIME FOR FILLER CHARS.?
2652 011320 001350      BNE      2$      ;; IF NO GO GET NEXT CHAR.
2653 011322 013746 001154      MOV      $NULL,-(SP)      ;; GET # OF FILLER CHARS. NEEDED
2654                                ;; AND THE NULL CHAR.
2655 011326 105366 000001      7$: DECB   1(SP)      ;; DOES A NULL NEED TO BE TYPED?
2656 011332 002770      BLT      6$      ;; BR IF NO--GO POP THE NULL OFF OF STACK
2657 011334 004737 011372      JSR      PC,$TYPEC      ;; GO TYPE A NULL
2658 011340 105337 011436      DECB   $CHARCNT      ;; DO NOT COUNT AS A COUNT
2659 011344 000770      BR      7$      ;; LOOP
2660
2661                                ;HORIZONTAL TAB PROCESSOR
2662
2663 011346 112716 000040      8$: MOVB   #' (SP)      ;; REPLACE TAB WITH SPACE
2664 011352 004737 011372      9$: JSR      PC,$TYPEC      ;; TYPE A SPACE
2665 011356 132737 000007 011436      BITB   #',$CHARCNT      ;; BRANCH IF NOT AT
2666 011364 001372      BNE      9$      ;; TAB STOP
2667 011366 005726      TST      (SP)+      ;; POP SPACE OFF STACK
2668 011370 000724      BR      2$      ;; GET NEXT CHARACTER
2669 011372 105777 167552      $TYPEC: TSTB  2$TPS      ;; WAIT UNTIL PRINTER IS READY
2670 011376 100375      BPL      $TYPEC
2671 011400 116677 000002 167544      MOVB   2(SP),2$TPB      ;; LOAD CHAR TO BE TYPED INTO DATA REG.
2672 011406 122766 000015 000002      CMPB   #CR,2(SP)      ;; IS CHARACTER A CARRIAGE RETURN?
2673 011414 001003      BNE      1$      ;; BRANCH IF NO
2674 011416 105037 011436      CLR      $CHARCNT      ;; YES--CLEAR CHARACTER COUNT
2675 011422 000406      BR      $TYPEX      ;; EXIT
2676 011424 122766 000012 000002      1$: CMPB   #LF,2(SP)      ;; IS CHARACTER A LINE FEED?
2677 011432 001402      BEQ      $TYPEX      ;; BRANCH IF YES
2678 011434 105227      INCB   (PC)+      ;; COUNT THE CHARACTER
2679 011436 000000      $CHARCNT: WORD 0      ;; CHARACTER COUNT STORAGE
2680 011440 000207      $TYPEX: RTS      PC
2681
2682                                .SBTTL APT COMMUNICATIONS ROUTINE
2683
2684                                ;*****
2685 011442 112737 000001 011706      $ATY1: MOVB   #1,$FFLG      ;; TO REPORT FATAL ERROR
2686 011450 112737 000001 011704      $ATY3: MOVB   #1,$MFLG      ;; TO TYPE A MESSAGE
2687 011456 000403      BR      $ATYC
2688 011460 112737 000001 011706      $ATY4: MOVB   #1,$FFLG      ;; TO ONLY REPORT FATAL ERROR
2689 011466      $ATYC:
2690 011466 010046      MOV      R0,-(SP)      ;; PUSH R0 ON STACK
2691 011470 010146      MOV      R1,-(SP)      ;; PUSH R1 ON STACK
2692 011472 105737 011704      TSTB   $MFLG      ;; SHOULD TYPE A MESSAGE?
2693 011476 001450      BEQ      5$      ;; IF NOT: BR
2694 011500 122737 000001 001214      CMPB   #APTENV,$ENV      ;; OPERATING UNDER APT?
2695 011506 001031      BNE      3$      ;; IF NOT: BR
2696 011510 132737 000100 001215      BITB   #APTPOOL,$ENVM      ;; SHOULD SPOOL MESSAGES?
2697 011516 001425      BEQ      3$      ;; IF NOT: BR

```

```

2698 011520 017600 000004      MOV      24(SP),R0      ;;GET MESSAGE ADDR.
2699 011524 062766 000002 000004      ADD      82,4(SP)      ;;BUMP RETURN ADDR.
2700 011532 005737 001174      15:     TST      $MSGTYPE    ;;SEE IF DONE W/ LAST XMISSION?
2701 011536 001375      BNE      15            ;;IF NOT: WAIT
2702 011540 010037 001210      MOV      R0,$MSGADR    ;;PUT ADDR IN MAILBOX
2703 011544 105720      25:     TSTB     (R0)+      ;;FIND END OF MESSAGE
2704 011546 001376      BNE      25
2705 011550 163700 001210      SUB      $MSGADR,R0    ;;SUB START OF MESSAGE
2706 011554 006200      ASR      R0            ;;GET MESSAGE LNTH IN WORDS
2707 011556 010037 001212      MOV      R0,$MSGLEN    ;;PUT LENGTH IN MAILBOX
2708 011562 012737 000004 001174      MOV      84,$MSGTYPE   ;;TELL APT TO TAKE MSG.
2709 011570 000413      BR       55
2710 011572 017637 000004 011616 35:     MOV      24(SP),4$     ;;PUT MSG ADDR IN JSR LINKAGE
2711 011600 062766 000002 000004      ADD      82,4(SP)      ;;BUMP RETURN ADDRESS
2712 011606 013746 177776      MOV      177776,-(SP)  ;;PUSH 177776 ON STACK
2713 011612 004737 011160      JSR     PC,$TYPE      ;;CALL TYPE MACRO
2714 011616 000000      45:     .WORD    0
2715 011620      55:
2716 011620 105737 011706 105:     TSTB     $FFLG         ;;SHOULD REPORT FATAL ERROR?
2717 011624 001416      BEQ     125           ;;IF NOT: BR
2718 011626 005737 001214      TST     $ENV          ;;RUNNING UNDER APT?
2719 011632 001413      BEQ     125           ;;IF NOT: BR
2720 011634 005737 001174 115:     TST     $MSGTYPE     ;;FINISHED LAST MESSAGE?
2721 011640 001375      BNE     115           ;;IF NOT: WAIT
2722 011642 017637 000004 001176      MOV      24(SP),$FATAL ;;GET ERROR #
2723 011650 062766 000002 000004      ADD      82,4(SP)      ;;BUMP RETURN ADDR.
2724 011656 005237 001174      INC     $MSGTYPE      ;;TELL APT TO TAKE ERROR
2725 011662 105037 011706 125:     CLRB    $FFLG         ;;CLEAR FATAL FLAG
2726 011666 105037 011705      CLRB    $LFLG        ;;CLEAR LOG FLAG
2727 011672 105037 011704      CLRB    $MFLG        ;;CLEAR MESSAGE FLAG
2728 011676 012601      MOV     (SP)+,R1      ;;POP STACK INTO R1
2729 011700 012600      MOV     (SP)+,R0      ;;POP STACK INTO R0
2730 011702 000207      RTS     PC            ;;RETURN
2731 011704      000      $MFLG: .BYTE    0      ;;MESSG. FLAG
2732 011705      000      $LFLG: .BYTE    0      ;;LOG FLAG
2733 011706      000      $FFLG: .BYTE    0      ;;FATAL FLAG
2734      011710      .EVEN
2735      000200      APTSIZE=200
2736      000001      APTENV=001
2737      000100      APTSPool=100
2738      000040      APTCSUP=040
2739      .SBTTL POWER DOWN AND UP ROUTINES
2740
2741      ;*****
2742      ;POWER DOWN ROUTINE
2743 011710 012737 012050 000024 $PWDRN: MOV      8$ILLUP,2$PWVVEC ;;SET FOR FAST UP
2744 011716 012737 000340 000026      MOV      8340,2$PWVVEC+2 ;;PRIO:7
2745 011724 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
2746 011726 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
2747 011730 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
2748 011732 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
2749 011734 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
2750 011736 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
2751 011740 017746 167174      MOV      2$SWR,-(SP)   ;;PUSH 2$SWR ON STACK

```

```

2752 011744 010637 012054      MOV      SP,$SAVR6      ;;SAVE SP
2753 011750 012737 011762 000024  MOV      $SPWRUP,$PWRVEC ;;SET UP VECTOR
2754 011756 000000      HALT
2755 011760 000776      BR      .-2            ;;HANG UP
2756
2757      ;*****
2758      ;POWER UP ROUTINE
2759 011762 012737 012050 000024 $PWRUP: MOV      $SILLUP,$PWRVEC ;;SET FOR FAST DOWN
2760 011770 013706 012054      MOV      $SAVR6,SP      ;;GET SP
2761 011774 005037 012054      CLR      $SAVR6         ;;WAIT LOOP FOR THE TTY
2762 012000 005237 012054      IS:    INC      $SAVR6   ;;WAIT FOR THE INC
2763 012004 001375      BNE     IS              ;;OF WORD
2764 012006 012677 167126      MOV      (SP)+,$SWR     ;;POP STACK INTO $SWR
2765 012012 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
2766 012014 012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
2767 012016 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
2768 012020 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
2769 012022 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
2770 012024 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
2771 012026 012737 011710 000024  MOV      $SPWRDN,$PWRVEC ;;SET UP THE POWER DOWN VECTOR
2772 012034 012737 000340 000026  MOV      $340,$PWRVEC+2 ;;PRIO:
2773 012042 104401      TYPE     $POWER        ;;REPORT THE POWER FAILURE
2774 012044 012056      SPWRMG: .WORD    $POWER ;;POWER FAIL MESSAGE POINTER
2775 012046 000002      RTI
2776 012050 000000      $SILLUP: HALT          ;;THE POWER UP SEQUENCE WAS STARTED
2777 012052 000776      BR      .-2            ;;BEFORE THE POWER DOWN WAS COMPLETE
2778 012054 000000      $SAVR6: 0              ;;PUT THE SP HERE
2779 012056 005015 047520 042527 $POWER: .ASCIZ  <15><12>"POWER"
2780 012064 000122      .EVEN
2781
2782
2783      ;*
2784      ;*THIS ROUTINE WILL PROTECT THE PROGRAM
2785      ;*FROM INTERRUPTS.
2786      ;*
2787      ;*THE TRAP CATCHER IS SET UP FOR
2788      ;*      .WORD    +2
2789      ;*      .WORD    JSR    PC,R0
2790      ;*
2791      ;*ILLEGAL INTERRUPTS OR INTERRUPTS TO THE WRONG VECTOR
2792      ;*GOTO THE VECTOR AND PICK UP THE ".+2" AS AN ADDRESS
2793      ;*AND "4700" AS NEW STATUS.
2794      ;*THE .+2 AS A PC WILL CAUSE EXECUTION OF THE "JSR PC,R0" (AN ILLEGAL INSTR).
2795      ;*AND TRAP TO LOCATION "4". IN LOCATION 10 WE HAVE A
2796      ;*POINTER HERE. IF THIS CONDITION CAUSES A TRAP TO LOC. 4
2797      ;*WE WILL REPORT IT IN THE SAME MANNER THAT WE WOULD
2798      ;*REPORT ANY OTHER ERROR.
2799
2800      ;*IF A BUSS ERROR TRAP DID OCCUR AND CAUSE A TRAP TO 4,
2801      ;*WE WILL HALT.
2802
2803 012066 011637 012132      IOTRD: MOV      (6),TRTO    ;GET WHERE WE CAME TO.
2804 012072 162737 000004 012132  SUB      #4,TRTO       ;FORM REAL ADDR.
2805

```

```

2806 012100 023727 012132 001000      CMP      TRTO,#1000      ;DID TRAP COME FROM LESS THAN ADDR. 1000?
2807 012106 003402                      BLE      2$
2808
2809 012110 000000                      1$:     HALT            ;NO! MUST BE A BUSS ILLEGAL ADDR. TIME OUT.
2810                                     ; ADDRESS CONTAINED IN TRTO.
2811
2812 012112 000776                      BR       1$            ;DON'T ALLOW A CONTINUE.
2813 012114                      2$:
2814
2815 012114 016637 000004 012134      MOV      4(6),TRFRO    ;GET TRAPPED FROM ADDR.
2816
2817 012122 062706 000004              ADD      #4,SP         ;/ADD #4 TO STACK POINTER.
2818
2819

```

;;SSSSSSSSSS>>> ERROR <<<SSSSSSSSSS

```

2823
2824 012126 104007                      ERROR   7             ;/MODULE FAULT DETECTED:
2825                                     ;ERROR! ILLEGAL INTERRUPT
2826                                     ;OR INTERRUPT TO WRONG
2827                                     ;VECTOR - IF TEST NUMBER
2828                                     ;IS LESS THAN 10, ITS LIKELY
2829                                     ;(BUT NOT EXCLUSIVELY) TO BE A
2830                                     ;DEVICE OTHER THAN THE IBV-11
2831                                     ;TO BLAME.
2832                                     ;IF THE INTERRUPT OCCURRED
2833                                     ;DURING AN INTERRUPT TEST, I'D
2834                                     ;SUSPECT A PROBLEM WITH THE
2835                                     ;IBV-11.
2836                                     ;IF THE ADDRESS THE INTERRUPT
2837                                     ;VECTOR TO IS WITHIN THE RANGE
2838                                     ;OF VECTORS ASSIGNED TO THE IBV-11,
2839                                     ;THEN I'D SUSPECT THE IBV-11
2840                                     ;INTERRUPTED ILLEGALLY.
2841                                     ;IF THE ADDRESS THE INTERRUPT
2842                                     ;VECTORED TO IS OUTSIDE OF THE
2843                                     ;RANGE ASSIGNED TO THE IBV-11,
2844                                     ;I'D SUSPECT THAT THE
2845                                     ;IBV-11 PUT THE WRONG VECTOR ON
2846                                     ;THE BUSS DURING THE INTERRUPT
2847                                     ;PROCESS.
2848                                     ;FOR THIS ERROR - DON'T
2849                                     ;USE "LOOP ON ERROR" OPTION.
2850                                     ;ALSO EXPECT THE INTERRUPT TEST TO
2851                                     ;REPORT THAT THE IBV-11 DIDN'T
2852                                     ;INTERRUPT.
2853                                     ;FOLLOW RECOMMENDED PROCEDURE
2854                                     ;IN THE DOCUMENT (ON THIS DIAGNOSTIC)
2855                                     ;FOR LOOPING ON ERROR

```

;;SSSSSSSSSS↑↑↑ ERROR ↑↑↑SSSSSSSSSS
ATI

2858 012130 000002
2859

2860 012132 000000
2861 012134 000000

TRTO: .WORD 0 ; ADDR THAT WE INTERRUPTED TO
TRFR0: .WORD 0 ; ADDR THAT WE INTERRUPTED FROM.

2862
2863
2864
2865
2866
2867
2868
2869
2870

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

2871 012136 010046
2872 012140 016600 000002
2873 012144 005740
2874 012146 111000
2875 012150 006300
2876 012152 016000 012172
2877 012156 000200

\$TRAP: MOV RO, -(SP) ; SAVE RO
MOV 2(SP), RO ; GET TRAP ADDRESS
TST -(RO) ; BACKUP BY 2
MOVB (RO), RO ; GET RIGHT BYTE OF TRAP
ASL RO ; POSITION FOR INDEXING
MOV \$TRPAD(RO), RO ; INDEX TO TABLE
RTS RO ; GO TO ROUTINE

2878
2879
2880

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

2881
2882 012160 011646
2883 012162 016666 000004 000002
2884 012170 000002
2885

\$TRAP2: MOV (SP), -(SP) ; MOVE THE PC DOWN
MOV 4(SP), 2(SP) ; MOVE THE PSW DOWN
RTI ; RESTORE THE PSW

2886
2887
2888
2889
2890
2891
2892

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

2893 012172 012160
2894 012174 011160
2895 012176 010532
2896 012200 010506
2897 012202 010546
2898 012204 010734
2899

; 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)
\$GTSWR ; ; CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING
\$CKSWR ; ; CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
\$RDCHR ; ; CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
\$RDLIN ; ; CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE

2900 012206 010002
2901
2902 012210 007732
2903 012212 010214
2904 012214 010334
2905
2906

.SBTTL MESSAGES AND TABLES

2908 012216 005007 044415 051502 EM1: .ASCIZ<7><12><15>#IBS FUNCTION ERROR#
2909 012224 043040 047125 052103
2910 012232 047511 020116 051105
2911 012240 047522 000122
2912
2913 012244 005007 044415 042102 EM2: .ASCIZ<7><12><15>#IBD FUNCTION ERROR#

MAINDEC-11-DVIBA-A MACY11 27(663) 29-MAR-77 12:57 PAGE 57
DVIBA.P11 MESSAGES AND TABLES

SEG 0072

| | | | | | | | | |
|------|--------|--------|--------|--------|-------|-------|-----------------------------------|--|
| 2968 | | 012640 | | | | | .EVEN | |
| 2969 | | | | | | | | |
| 2970 | 012640 | 001200 | 001116 | 001346 | DT1: | .WORD | \$TESTN,\$ERRPC,IBS | |
| 2971 | | | | | | | | |
| 2972 | 012646 | 000000 | | | IBSA: | .WORD | 0 | |
| 2973 | | | | | | | | |
| 2974 | 012650 | 000000 | 000000 | | IBOA: | .WORD | 0,0 | |
| 2975 | | | | | | | | |
| 2976 | 012654 | 001200 | 001116 | 001124 | DT3: | .WORD | \$TESTN,\$ERRPC,\$GDDAT,\$BDDAT,0 | |
| 2977 | 012662 | 001126 | 000000 | | | | | |
| 2978 | | | | | | | | |
| 2979 | 012666 | 001200 | 001116 | 001346 | DT5: | .WORD | \$TESTN,\$ERRPC,IBS,0 | |
| 2980 | 012674 | 000000 | | | | | | |
| 2981 | | | | | | | | |
| 2982 | 012676 | 001200 | 001116 | 012132 | DT7: | .WORD | \$TESTN,\$ERRPC,TRTO,TRFRO,0 | |
| 2983 | 012704 | 012134 | 000000 | | | | | |
| 2984 | | | | | | | | |
| 2985 | 012710 | 000000 | 000000 | | DF0: | .WORD | 0,0 | |
| 2986 | | | | | | | | |
| 2987 | | 000001 | | | | | .END | |

| | | | | | | | | | | | | | | | | | | | | |
|-----------------|------|-------|-------|-------|-----|-----|------|------|------|------|------|------|------|--|--|--|--|--|--|--|
| ABASE = 160150 | 167# | 255 | 296 | 376 | 377 | 378 | 379 | 384 | 385 | | | | | | | | | | | |
| ACDM1 = 000000 | 255 | 298 | | | | | | | | | | | | | | | | | | |
| ACDM2 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ACPUOP = 000000 | 255 | 270 | | | | | | | | | | | | | | | | | | |
| ADDW0 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW1 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW10 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW11 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW12 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW13 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW14 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW15 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW2 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW3 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW4 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW5 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW6 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW7 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW8 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADDW9 = 000000 | 255 | | | | | | | | | | | | | | | | | | | |
| ADEVCT = 000000 | 255 | 261 | | | | | | | | | | | | | | | | | | |
| ADEVN = 000000 | 255 | 297 | | | | | | | | | | | | | | | | | | |
| AENV = 000000 | 255 | 266 | | | | | | | | | | | | | | | | | | |
| AENVN = 000000 | 255 | 267 | | | | | | | | | | | | | | | | | | |
| AFATAL = 000000 | 255 | 258 | | | | | | | | | | | | | | | | | | |
| AMADR1 = 000000 | 255 | 283 | | | | | | | | | | | | | | | | | | |
| AMADR2 = 000000 | 255 | 287 | | | | | | | | | | | | | | | | | | |
| AMADR3 = 000000 | 255 | 290 | | | | | | | | | | | | | | | | | | |
| AMADR4 = 000000 | 255 | 293 | | | | | | | | | | | | | | | | | | |
| AMAMS1 = 000000 | 255 | 277 | | | | | | | | | | | | | | | | | | |
| AMAMS2 = 000000 | 255 | 285 | | | | | | | | | | | | | | | | | | |
| AMAMS3 = 000000 | 255 | 288 | | | | | | | | | | | | | | | | | | |
| AMAMS4 = 000000 | 255 | 291 | | | | | | | | | | | | | | | | | | |
| AMSGAD = 000000 | 255 | 263 | | | | | | | | | | | | | | | | | | |
| AMSGLG = 000000 | 255 | 264 | | | | | | | | | | | | | | | | | | |
| AMSGTY = 000000 | 255 | 257 | | | | | | | | | | | | | | | | | | |
| AMTYP1 = 000000 | 255 | 278 | | | | | | | | | | | | | | | | | | |
| AMTYP2 = 000000 | 255 | 286 | | | | | | | | | | | | | | | | | | |
| AMTYP3 = 000000 | 255 | 289 | | | | | | | | | | | | | | | | | | |
| AMTYP4 = 000000 | 255 | 292 | | | | | | | | | | | | | | | | | | |
| APASS = 000000 | 255 | 260 | | | | | | | | | | | | | | | | | | |
| APRIOR = 000200 | 169# | 255 | | | | | | | | | | | | | | | | | | |
| APTCSU = 000040 | 2633 | 2738# | | | | | | | | | | | | | | | | | | |
| APTEMV = 000001 | 2189 | 2626 | 2694 | 2736# | | | | | | | | | | | | | | | | |
| APTSIZ = 000200 | 446 | 2735# | | | | | | | | | | | | | | | | | | |
| APTSPO = 000100 | 2628 | 2696 | 2737# | | | | | | | | | | | | | | | | | |
| ASWREG = 000000 | 255 | 268 | | | | | | | | | | | | | | | | | | |
| ATESTN = 000000 | 255 | 259 | | | | | | | | | | | | | | | | | | |
| AUNIT = 000000 | 255 | 262 | | | | | | | | | | | | | | | | | | |
| AUSWR = 000000 | 255 | 269 | | | | | | | | | | | | | | | | | | |
| AVECT1 = 000640 | 168# | 255 | 294 | 380 | 381 | 382 | 383 | 386 | 387 | 388 | 389 | 393 | 394 | | | | | | | |
| | 395 | 396 | 398 | 399 | 400 | 401 | | | | | | | | | | | | | | |
| AVECT2 = 000000 | 255 | 295 | | | | | | | | | | | | | | | | | | |
| BITO = 000001 | 151# | 722 | 723 | 738 | 853 | 991 | 1009 | 1472 | 1475 | 1490 | 1583 | 1677 | 2070 | | | | | | | |

| | | | | | | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| ERRVEC= 000004 | 154# | 431 | 432* | 443* | 450* | 451* | 540 | 541* | 547* | 575* | 594 | 595* | 601* |
| | 633* | 2277 | 2278* | 2280* | 2283* | | | | | | | | |
| GNS = ##### U | 525 | 2894 | 2895 | 2896 | 2897 | 2898 | 2900 | 2902 | 2903 | 2904 | | | |
| GTSWR = 104406 | 520 | 2900# | | | | | | | | | | | |
| HT = 000011 | 64# | 2641 | 2682 | | | | | | | | | | |
| IBCA 001354 | 379# | 458* | 459* | 603 | 703 | | | | | | | | |
| IBD 001350 | 377# | 454* | 455* | 456 | 463 | 549 | 665 | 992* | 994 | 1009* | 1010 | 1031* | 1033 |
| | 1048# | 1049 | 1070* | 1072 | 1087* | 1088 | 1109* | 1111 | 1126* | 1127 | 1148* | 1150 | 1165* |
| | 1166 | 1187* | 1189 | 1204* | 1205 | 1226* | 1228 | 1243* | 1244 | 1265* | 1267 | 1282* | 1283 |
| | 1302* | 1304 | 1325* | 1326 | 1340* | 1341 | 1356 | 1357 | 1369 | 1390 | 1395 | 1408 | 1430 |
| | 1435 | 1451 | 1473 | 1478 | 1491 | 1513 | 1518 | 1531 | 1550 | 1564 | 1585* | 1650* | 1652 |
| | 1744* | 1780* | 2072* | | | | | | | | | | |
| IBDA 012650 | 463* | 2974# | | | | | | | | | | | |
| IBD2 001370 | 385# | 464* | 465* | 2073 | | | | | | | | | |
| IBS 001346 | 376# | 453* | 454 | 462 | 545 | 644 | 721* | 722* | 724 | 738* | 740 | 757* | 758* |
| | 760 | 774* | 776 | 793* | 794* | 796 | 810* | 812 | 829* | 830* | 832 | 835* | 836 |
| | 854 | 871* | 872* | 874 | 889* | 891 | 914* | 916* | 918 | 933* | 934 | 951* | 952* |
| | 954 | 968* | 970 | 990* | 1029* | 1068* | 1107* | 1146* | 1185* | 1224* | 1263* | 1301* | 1324* |
| | 1339* | 1355* | 1388* | 1389* | 1392* | 1407* | 1428* | 1429* | 1432* | 1448 | 1471* | 1472* | 1475* |
| | 1490* | 1511* | 1512* | 1515* | 1530* | 1549* | 1563* | 1582* | 1583* | 1587 | 1605* | 1607 | 1626* |
| | 1627* | 1639 | 1630 | 1649* | 1674* | 1677* | 1697* | 1708* | 1709* | 1726* | 1761* | 1771* | 1779* |
| | 1802* | 1830* | 1873* | 1883* | 1908* | 1929 | 1953* | 1955 | 1959 | 1986* | 1990 | 2016* | 2021* |
| | 2048* | 203* | 2070* | 2091* | 2970 | 2979 | | | | | | | |
| IBSA 012646 | 462* | 2972# | | | | | | | | | | | |
| IBS2 001366 | 384# | 464 | 1831* | 1839* | 1856* | 1874* | 1884* | 1909* | 1924* | 1925* | 1954* | 1958* | 1987* |
| | 1988* | 2015* | 2027* | 2047* | 2069* | 2071* | 2092* | | | | | | |
| IBWC 001352 | 378# | 456* | 457* | 458 | 598 | 689 | | | | | | | |
| IOTRD 012066 | 46 | 450 | 2803# | | | | | | | | | | |
| IOTVEC= 000020 | 159# | 416* | 417* | | | | | | | | | | |
| LF = 000012 | 65# | 2676 | 2682 | | | | | | | | | | |
| PC =%000007 | 85# | 2117* | 2120* | 2130* | 2135 | 2186* | 2192* | 2245* | 2374* | 2631* | 2650* | 2657* | 2664* |
| | 2678* | 2680* | 2713* | 2730* | | | | | | | | | |
| PIRQ = 177772 | 71# | | | | | | | | | | | | |
| PIRQVE= 000240 | 165# | | | | | | | | | | | | |
| PRA 001402 | 393# | 479* | 480* | 481 | 497 | 498* | 1772* | 1804 | 1805* | 2018* | 2045 | 2046* | |
| PRA2 001412 | 398# | 487* | 488* | 489 | | | | | | | | | |
| PR8 001404 | 394# | 481* | 482* | 483 | 500 | 501* | 1876* | 1906 | 1907* | | | | |
| PR82 001414 | 399# | 489* | 490* | 491 | | | | | | | | | |
| PRC 001406 | 395# | 483* | 484* | 485 | 503 | 504* | 1675* | 1699 | 1700* | 1706* | 1728 | 1729* | 1735 |
| | 1736* | | | | | | | | | | | | |
| PRC2 001416 | 400# | 491* | 492* | 493 | 1832* | 1858 | 1859* | | | | | | |
| PRO 001410 | 396# | 485* | 486* | 506 | 507* | 1737* | 1763 | 1764* | | | | | |
| PRO2 001420 | 401# | 493* | 494* | | | | | | | | | | |
| PRO = 000000 | 88# | | | | | | | | | | | | |
| PR1 = 000040 | 89# | | | | | | | | | | | | |
| PR2 = 000100 | 90# | | | | | | | | | | | | |
| PR3 = 000140 | 91# | | | | | | | | | | | | |
| PR4 = 000200 | 92# | | | | | | | | | | | | |
| PR5 = 000240 | 93# | | | | | | | | | | | | |
| PR6 = 000300 | 94# | | | | | | | | | | | | |
| PR7 = 000340 | 95# | | | | | | | | | | | | |
| PS = 177776 | 68# | 69 | | | | | | | | | | | |
| PSH = 177776 | 69# | | | | | | | | | | | | |
| PMRVEC= 000024 | 160# | 422* | 423* | 2743* | 2744* | 2753* | 2759* | 2771* | 2772* | | | | |

| | | | | | | | |
|--------|---|--------|-------|-------|-------|-------|-------|
| SE14 | = | 040000 | 99# | | | | |
| SE15 | = | 100000 | 98# | | | | |
| SE2 | = | 000004 | 121# | | | | |
| SE3 | = | 000010 | 120# | | | | |
| SE4 | = | 000020 | 119# | | | | |
| SE5 | = | 000040 | 118# | | | | |
| SE6 | = | 000100 | 117# | | | | |
| SE7 | = | 000200 | 116# | | | | |
| SE8 | = | 000400 | 115# | | | | |
| SE9 | = | 001000 | 114# | | | | |
| TBITVE | = | 000014 | 156# | | | | |
| TKVEC | = | 000060 | 163# | | | | |
| TPVEC | = | 000064 | 164# | | | | |
| TRAPVE | = | 000034 | 162# | 420* | 421* | | |
| TRFR0 | = | 012134 | 2815# | 2861# | 2982 | | |
| TRT0 | = | 012132 | 2803# | 2804* | 2806 | 2860# | 2982 |
| TRTVEC | = | 000014 | 157# | | | | |
| TST1 | = | 002444 | 532# | | | | |
| TST10 | = | 003156 | 772 | 777 | 791# | | |
| TST11 | = | 003246 | 808 | 813 | 827# | | |
| TST12 | = | 003372 | 848 | 856 | 869# | | |
| TST13 | = | 003462 | 887 | 892 | 906# | | |
| TST14 | = | 003564 | 930 | 935 | 949# | | |
| TST15 | = | 003654 | 966 | 971 | 987# | | |
| TST16 | = | 003746 | 1007 | 1011 | 1026# | | |
| TST17 | = | 004040 | 1046 | 1050 | 1065# | | |
| TST2 | = | 002570 | 591# | | | | |
| TST20 | = | 004132 | 1085 | 1089 | 1104# | | |
| TST21 | = | 004224 | 1124 | 1128 | 1143# | | |
| TST22 | = | 004316 | 1163 | 1167 | 1182# | | |
| TST23 | = | 004410 | 1202 | 1206 | 1221# | | |
| TST24 | = | 004502 | 1241 | 1245 | 1260# | | |
| TST25 | = | 004574 | 1280 | 1284 | 1299# | | |
| TST26 | = | 004626 | 1305 | 1322# | | | |
| TST27 | = | 004752 | 1337 | 1353 | 1368 | 1370 | 1386# |
| TST3 | = | 002654 | 638# | | | | |
| TST30 | = | 005040 | 1406 | 1409 | 1426# | | |
| TST31 | = | 005130 | 1446 | 1452 | 1469# | | |
| TST32 | = | 005216 | 1489 | 1492 | 1509# | | |
| TST33 | = | 005304 | 1529 | 1532 | 1547# | | |
| TST34 | = | 005346 | 1561 | 1565 | 1580# | | |
| TST35 | = | 005400 | 1588 | 1602# | | | |
| TST36 | = | 005437 | 1608 | 1623# | | | |
| TST37 | = | 005476 | 1631 | 1646# | | | |
| TST4 | = | 002704 | 645 | 659# | | | |
| TST40 | = | 005534 | 1653 | 1672# | | | |
| TST41 | = | 005632 | 1705# | | | | |
| TST42 | = | 006034 | 1730 | 1769# | | | |
| TST43 | = | 006152 | 1814# | | | | |
| TST44 | = | 006266 | 1864# | | | | |
| TST45 | = | 006402 | 1914# | | | | |
| TST46 | = | 006426 | 1930 | 1944# | | | |
| TST47 | = | 006500 | 1957 | 1960 | 1976# | | |
| TST5 | = | 002734 | 666 | 685# | | | |

| | | | | | | | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| ADD | 455 | 457 | 459 | 465 | 467 | 469 | 471 | 473 | 475 | 477 | 480 | 482 | 484 | 486 | 488 |
| | 490 | 492 | 494 | 553 | 565 | 607 | 621 | 1696 | 1733 | 1760 | 1801 | 1855 | 1902 | 2042 | 2230 |
| ASL | 2359 | 2368 | 2487 | 2497 | 2568 | 2639 | 2699 | 2711 | 2723 | 2817 | | | | | |
| ASLB | 2227 | 2228 | 2229 | 2382 | 2383 | 2384 | 2875 | | | | | | | | |
| ASR | 2573 | | | | | | | | | | | | | | |
| BCC | 2706 | | | | | | | | | | | | | | |
| BEQ | 2574 | | | | | | | | | | | | | | |
| | 447 | 517 | 645 | 666 | 690 | 704 | 741 | 762 | 777 | 798 | 813 | 834 | 838 | 856 | 876 |
| | 892 | 920 | 935 | 956 | 971 | 996 | 1011 | 1035 | 1050 | 1074 | 1089 | 1113 | 1128 | 1152 | 1167 |
| | 1191 | 1206 | 1230 | 1245 | 1269 | 1284 | 1305 | 1358 | 1370 | 1409 | 1452 | 1492 | 1532 | 1565 | 1608 |
| | 1631 | 1653 | 2075 | 2128 | 2175 | 2178 | 2201 | 2204 | 2232 | 2237 | 2250 | 2287 | 2289 | 2291 | 2295 |
| | 2304 | 2339 | 2366 | 2381 | 2514 | 2629 | 2642 | 2677 | 2693 | 2697 | 2717 | 2719 | | | |
| BGE | 2307 | | | | | | | | | | | | | | |
| BGT | 2119 | 2378 | 2419 | 2521 | 2582 | | | | | | | | | | |
| BHI | 2293 | | | | | | | | | | | | | | |
| BIC | 461 | 738 | 774 | 810 | 889 | 933 | 968 | 1009 | 1048 | 1087 | 1126 | 1165 | 1204 | 1243 | 1282 |
| | 1407 | 1490 | 1530 | 2116 | 2335 | 2352 | 2379 | 2406 | 2412 | 2420 | 2511 | | | | |
| BIS | 722 | 723 | 758 | 794 | 830 | 872 | 916 | 952 | 1339 | 1389 | 1392 | 1429 | 1432 | 1472 | 1475 |
| | 1512 | 1515 | 1563 | 1583 | 1627 | 1677 | 1709 | 1779 | 1884 | 1988 | 2021 | 2027 | 2070 | 2071 | 2386 |
| | 2516 | 2517 | 2576 | 2577 | | | | | | | | | | | |
| BISB | 2219 | | | | | | | | | | | | | | |
| BIT | 1326 | 1342 | 1357 | 1369 | 1390 | 1395 | 1408 | 1430 | 1435 | 1448 | 1451 | 1473 | 1478 | 1491 | 1513 |
| | 1518 | 1531 | 1550 | 1564 | 1587 | 1628 | 1630 | 1929 | 1955 | 1959 | 1990 | 2177 | 2184 | 2200 | 2272 |
| | 2286 | 2294 | 2301 | | | | | | | | | | | | |
| BITB | 446 | 2628 | 2633 | 2665 | 2696 | | | | | | | | | | |
| BLE | 2807 | | | | | | | | | | | | | | |
| BLOS | 2432 | | | | | | | | | | | | | | |
| BLT | 2376 | 2417 | 2522 | 2565 | 2581 | 2656 | | | | | | | | | |
| BMI | 2572 | | | | | | | | | | | | | | |
| BNE | 413 | 436 | 511 | 515 | 519 | 1327 | 1343 | 1391 | 1396 | 1431 | 1436 | 1449 | 1474 | 1479 | 1514 |
| | 1519 | 1551 | 1588 | 1629 | 1826 | 1930 | 1957 | 1960 | 1991 | 2089 | 2153 | 2155 | 2185 | 2190 | 2220 |
| | 2242 | 2273 | 2302 | 2331 | 2337 | 2357 | 2364 | 2371 | 2408 | 2414 | 2436 | 2442 | 2512 | 2570 | 2627 |
| | 2634 | 2636 | 2644 | 2652 | 2666 | 2673 | 2695 | 2701 | 2704 | 2721 | 2763 | | | | |
| BPL | 2197 | 2333 | 2349 | 2404 | 2410 | 2510 | 2556 | 2586 | 2621 | 2670 | | | | | |
| BR | 438 | 521 | 524 | 551 | 604 | 618 | 701 | 726 | 736 | 772 | 808 | 848 | 887 | 930 | 966 |
| | 1007 | 1046 | 1085 | 1124 | 1163 | 1202 | 1241 | 1280 | 1337 | 1353 | 1368 | 1406 | 1446 | 1489 | 1529 |
| | 1561 | 1694 | 1730 | 1757 | 1799 | 1853 | 1899 | 2039 | 2086 | 2195 | 2225 | 2252 | 2275 | 2281 | 2284 |
| | 2297 | 2300 | 2360 | 2387 | 2389 | 2415 | 2438 | 2488 | 2503 | 2524 | 2567 | 2584 | 2623 | 2649 | 2659 |
| | 2668 | 2675 | 2687 | 2709 | 2755 | 2777 | 2812 | | | | | | | | |
| CLR | 411 | 424 | 425 | 445 | 643 | 664 | 688 | 721 | 739 | 757 | 775 | 793 | 811 | 829 | 871 |
| | 890 | 914 | 932 | 951 | 969 | 1008 | 1047 | 1086 | 1125 | 1164 | 1203 | 1242 | 1281 | 1301 | 1303 |
| | 1324 | 1325 | 1388 | 1428 | 1471 | 1511 | 1549 | 1582 | 1674 | 1697 | 1726 | 1761 | 1771 | 1802 | 1830 |
| | 1831 | 1856 | 1873 | 1874 | 1908 | 1909 | 1924 | 1953 | 1986 | 1987 | 2015 | 2016 | 2047 | 2048 | 2066 |
| | 2068 | 2069 | 2091 | 2092 | 2113 | 2114 | 2218 | 2299 | 2314 | 2346 | 2347 | 2501 | 2559 | 2562 | 2761 |
| CLRB | 1585 | 1780 | 2298 | 2443 | 2588 | 2648 | 2674 | 2725 | 2726 | 2727 | | | | | |
| CMP | 412 | 435 | 518 | 725 | 761 | 797 | 833 | 837 | 855 | 875 | 919 | 955 | 2282 | 2306 | 2330 |
| | 2336 | 2356 | 2363 | 2375 | 2377 | 2407 | 2413 | 2416 | 2418 | 2431 | 2580 | 2806 | 2292 | 2338 | 2370 |
| CMPB | 516 | 995 | 1034 | 1073 | 1112 | 1151 | 1190 | 1229 | 1268 | 2074 | 2189 | 2288 | | | |
| | 2435 | 2441 | 2626 | 2641 | 2643 | 2651 | 2672 | 2676 | 2694 | | | | | | |
| DEC | 2117 | 2152 | 2154 | 2226 | | | | | | | | | | | |
| DECB | 2509 | 2520 | 2655 | 2658 | | | | | | | | | | | |
| EMT | 60 | | | | | | | | | | | | | | |
| HALT | 2198 | 2622 | 2754 | 2776 | 2809 | | | | | | | | | | |
| INC | 510 | 1925 | 2115 | 2180 | 2305 | 2385 | 2515 | 2523 | 2566 | 2724 | 2762 | | | | |

| | | | | | | | | | | | | | | | | | | | |
|---------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|--|--|--|--|
| INCB | 1744 | 2088 | 2174 | 2310 | 2678 | | | | | | | | | | | | | | |
| IOT | 61 | | | | | | | | | | | | | | | | | | |
| JMP | 54 | 1827 | 2135 | | | | | | | | | | | | | | | | |
| JSR | 850 | 2130 | 2186 | 2192 | 2374 | 2631 | 2650 | 2657 | 2664 | 2713 | | | | | | | | | |
| MOV | 410 | 414 | 416 | 417 | 418 | 419 | 420 | 421 | 422 | 423 | 427 | 428 | 431 | 432 | 433 | | | | |
| | 434 | 439 | 441 | 442 | 443 | 448 | 450 | 451 | 453 | 454 | 456 | 458 | 460 | 462 | 463 | | | | |
| | 464 | 466 | 468 | 470 | 472 | 474 | 476 | 479 | 481 | 483 | 485 | 487 | 489 | 491 | 493 | | | | |
| | 497 | 498 | 500 | 501 | 503 | 504 | 506 | 507 | 533 | 534 | 536 | 537 | 538 | 540 | 541 | | | | |
| | 547 | 575 | 577 | 578 | 592 | 594 | 595 | 601 | 633 | 639 | 644 | 660 | 686 | 689 | 703 | | | | |
| | 724 | 740 | 759 | 760 | 776 | 795 | 796 | 812 | 831 | 832 | 835 | 836 | 853 | 854 | 873 | | | | |
| | 874 | 891 | 909 | 910 | 917 | 918 | 934 | 953 | 954 | 970 | 990 | 991 | 992 | 1029 | 1030 | | | | |
| | 1031 | 1068 | 1069 | 1070 | 1107 | 1108 | 1109 | 1146 | 1147 | 1148 | 1185 | 1186 | 1187 | 1224 | 1225 | | | | |
| | 1226 | 1263 | 1264 | 1265 | 1340 | 1341 | 1355 | 1603 | 1605 | 1624 | 1626 | 1647 | 1649 | 1650 | 1675 | | | | |
| | 1676 | 1679 | 1680 | 1699 | 1700 | 1706 | 1707 | 1708 | 1711 | 1712 | 1728 | 1729 | 1735 | 1736 | 1737 | | | | |
| | 1738 | 1740 | 1741 | 1763 | 1764 | 1772 | 1773 | 1775 | 1776 | 1784 | 1785 | 1804 | 1805 | 1832 | 1833 | | | | |
| | 1835 | 1836 | 1839 | 1858 | 1859 | 1876 | 1877 | 1879 | 1880 | 1883 | 1906 | 1907 | 1945 | 1954 | 1958 | | | | |
| | 2018 | 2019 | 2023 | 2024 | 2045 | 2046 | 2064 | 2065 | 2072 | 2120 | 2124 | 2127 | 2150 | 2151 | 2176 | | | | |
| | 2181 | 2202 | 2205 | 2217 | 2222 | 2231 | 2236 | 2241 | 2243 | 2247 | 2277 | 2278 | 2280 | 2283 | 2296 | | | | |
| | 2308 | 2309 | 2312 | 2313 | 2316 | 2317 | 2343 | 2367 | 2372 | 2401 | 2402 | 2429 | 2430 | 2445 | 2446 | | | | |
| | 2447 | 2448 | 2484 | 2492 | 2493 | 2494 | 2500 | 2507 | 2525 | 2526 | 2527 | 2528 | 2529 | 2549 | 2550 | | | | |
| | 2551 | 2552 | 2553 | 2554 | 2555 | 2560 | 2563 | 2583 | 2589 | 2590 | 2591 | 2592 | 2593 | 2595 | 2596 | | | | |
| | 2624 | 2625 | 2630 | 2638 | 2653 | 2690 | 2691 | 2698 | 2702 | 2707 | 2708 | 2710 | 2712 | 2722 | 2728 | | | | |
| | 2729 | 2743 | 2744 | 2745 | 2746 | 2747 | 2748 | 2749 | 2750 | 2751 | 2752 | 2753 | 2759 | 2760 | 2764 | | | | |
| | 2765 | 2766 | 2767 | 2768 | 2769 | 2770 | 2771 | 2772 | 2803 | 2815 | 2871 | 2872 | 2876 | 2882 | 2883 | | | | |
| MOV8 | 426 | 522 | 665 | 994 | 1010 | 1033 | 1049 | 1072 | 1088 | 1111 | 1127 | 1150 | 1166 | 1189 | 1205 | | | | |
| | 1228 | 1244 | 1267 | 1283 | 1302 | 1304 | 2073 | 2183 | 2191 | 2311 | 2315 | 2334 | 2351 | 2405 | 2411 | | | | |
| | 2434 | 2439 | 2485 | 2486 | 2489 | 2490 | 2491 | 2495 | 2498 | 2499 | 2518 | 2558 | 2561 | 2575 | 2578 | | | | |
| | 2587 | 2635 | 2663 | 2671 | 2685 | 2686 | 2688 | 2874 | | | | | | | | | | | |
| NEG | 2496 | 2557 | | | | | | | | | | | | | | | | | |
| NOP | 532 | 1683 | 1684 | 1715 | 1716 | 1746 | 1747 | 1781 | 1782 | 1788 | 1840 | 1841 | 1887 | 1888 | 2028 | | | | |
| | 2029 | 2131 | 2132 | 2133 | | | | | | | | | | | | | | | |
| RESET | 527 | 641 | 662 | 1606 | 1651 | 2129 | | | | | | | | | | | | | |
| ROL | 2502 | 2504 | 2505 | 2506 | 2508 | | | | | | | | | | | | | | |
| RTI | 440 | 579 | 911 | 1681 | 1713 | 1742 | 1777 | 1786 | 1837 | 1881 | 2025 | 2207 | 2318 | 2373 | 2421 | | | | |
| | 2449 | 2530 | 2597 | 2640 | 2775 | 2858 | 2884 | | | | | | | | | | | | |
| RTS | 2156 | 2245 | 2680 | 2730 | 2877 | | | | | | | | | | | | | | |
| SUB | 2182 | 2564 | 2705 | 2804 | | | | | | | | | | | | | | | |
| TRAP | 2886 | 2895 | 2896 | 2897 | 2898 | 2900 | 2902 | 2903 | 2904 | | | | | | | | | | |
| TST | 514 | 545 | 549 | 598 | 603 | 1825 | 2196 | 2203 | 2249 | 2279 | 2303 | 2365 | 2380 | 2513 | 2569 | | | | |
| | 2579 | 2637 | 2645 | 2667 | 2700 | 2718 | 2720 | 2873 | | | | | | | | | | | |
| TSTB | 1356 | 1607 | 1652 | 2290 | 2332 | 2348 | 2403 | 2409 | 2571 | 2585 | 2620 | 2669 | 2692 | 2703 | 2716 | | | | |
| .ASCII | 248 | 249 | | | | | | | | | | | | | | | | | |
| .ASCIIZ | 247 | 250 | 526 | 2138 | 2253 | 2453 | 2454 | 2455 | 2457 | 2779 | 2908 | 2913 | 2918 | 2922 | 2926 | | | | |
| | 2930 | 2935 | 2940 | 2948 | 2954 | 2960 | | | | | | | | | | | | | |
| .BLKB | 2452 | | | | | | | | | | | | | | | | | | |
| .BLKW | 2602 | | | | | | | | | | | | | | | | | | |
| .BYTE | 217 | 218 | 223 | 224 | 232 | 233 | 241 | 242 | 243 | 244 | 266 | 267 | 277 | 278 | 285 | | | | |
| | 286 | 288 | 289 | 291 | 292 | 2137 | 2193 | 2194 | 2450 | 2451 | 2531 | 2532 | 2533 | 2534 | 2731 | | | | |
| | 2732 | 2733 | | | | | | | | | | | | | | | | | |
| .DSABL | 2390 | | | | | | | | | | | | | | | | | | |
| .ENABL | 3 | 4 | 2323 | | | | | | | | | | | | | | | | |
| .END | 2987 | | | | | | | | | | | | | | | | | | |
| .ENDC | 17 | 32 | 34 | 35 | 36 | 60 | 152 | 166 | 175 | 179 | 181 | 188 | 190 | 197 | 211 | | | | |
| | 215 | 217 | 245 | 246 | 247 | 248 | 252 | 255 | 277 | 285 | 288 | 291 | 294 | 295 | 296 | | | | |

| | | | | | | | | | | | | | | | |
|--------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| | 297 | 298 | 301 | 406 | 414 | 415 | 418 | 420 | 422 | 424 | 425 | 427 | 429 | 450 | 512 |
| | 518 | 524 | 526 | 530 | 531 | 532 | 533 | 534 | 535 | 584 | 585 | 590 | 591 | 592 | 593 |
| | 636 | 637 | 638 | 639 | 640 | 646 | 657 | 658 | 659 | 660 | 661 | 667 | 679 | 680 | 684 |
| | 685 | 686 | 687 | 702 | 705 | 717 | 718 | 719 | 720 | 737 | 742 | 753 | 754 | 755 | 756 |
| | 773 | 778 | 789 | 790 | 791 | 792 | 809 | 814 | 825 | 826 | 827 | 828 | 849 | 857 | 867 |
| | 868 | 869 | 870 | 888 | 893 | 904 | 905 | 906 | 907 | 931 | 936 | 947 | 948 | 949 | 950 |
| | 967 | 972 | 985 | 986 | 987 | 988 | 1008 | 1012 | 1024 | 1025 | 1026 | 1027 | 1047 | 1051 | 1063 |
| | 1064 | 1065 | 1066 | 1086 | 1090 | 1102 | 1103 | 1104 | 1105 | 1125 | 1129 | 1141 | 1142 | 1143 | 1144 |
| | 1164 | 1168 | 1180 | 1181 | 1182 | 1183 | 1203 | 1207 | 1219 | 1220 | 1221 | 1222 | 1242 | 1246 | 1258 |
| | 1259 | 1260 | 1261 | 1281 | 1285 | 1297 | 1298 | 1299 | 1300 | 1306 | 1320 | 1321 | 1322 | 1323 | 1338 |
| | 1354 | 1369 | 1371 | 1384 | 1385 | 1386 | 1387 | 1397 | 1407 | 1410 | 1420 | 1424 | 1425 | 1426 | 1427 |
| | 1437 | 1447 | 1453 | 1463 | 1467 | 1468 | 1469 | 1470 | 1480 | 1490 | 1493 | 1503 | 1507 | 1508 | 1509 |
| | 1510 | 1520 | 1530 | 1533 | 1543 | 1545 | 1546 | 1547 | 1548 | 1562 | 1566 | 1578 | 1579 | 1580 | 1581 |
| | 1589 | 1600 | 1601 | 1602 | 1603 | 1604 | 1609 | 1621 | 1622 | 1623 | 1624 | 1625 | 1632 | 1644 | 1645 |
| | 1646 | 1647 | 1648 | 1654 | 1670 | 1671 | 1672 | 1673 | 1703 | 1704 | 1705 | 1706 | 1731 | 1767 | 1768 |
| | 1769 | 1770 | 1812 | 1813 | 1814 | 1815 | 1862 | 1863 | 1864 | 1865 | 1912 | 1913 | 1914 | 1915 | 1931 |
| | 1942 | 1943 | 1944 | 1945 | 1946 | 1958 | 1961 | 1974 | 1975 | 1976 | 1977 | 1992 | 2003 | 2004 | 2005 |
| | 2006 | 2052 | 2053 | 2061 | 2062 | 2063 | 2087 | 2095 | 2096 | 2097 | 2098 | 2106 | 2107 | 2108 | 2110 |
| | 2113 | 2119 | 2122 | 2123 | 2127 | 2129 | 2135 | 2137 | 2138 | 2141 | 2161 | 2164 | 2174 | 2181 | 2186 |
| | 2187 | 2188 | 2196 | 2207 | 2208 | 2211 | 2226 | 2255 | 2258 | 2261 | 2266 | 2272 | 2274 | 2295 | 2288 |
| | 2289 | 2290 | 2292 | 2294 | 2301 | 2305 | 2310 | 2312 | 2316 | 2319 | 2320 | 2323 | 2324 | 2326 | 2354 |
| | 2390 | 2394 | 2422 | 2423 | 2430 | 2432 | 2435 | 2437 | 2453 | 2459 | 2462 | 2539 | 2606 | 2635 | 2685 |
| | 2686 | 2689 | 2716 | 2731 | 2742 | 2751 | 2752 | 2758 | 2764 | 2765 | 2775 | 2782 | 2866 | 2872 | 2875 |
| | 2894 | 2895 | 2896 | 2897 | 2898 | 2899 | 2900 | 2901 | 2902 | 2903 | 2904 | 2905 | | | |
| .EQUIV | 60 | 61 | 69 | 114 | 115 | 116 | 117 | 118 | 119 | 120 | 121 | 122 | 123 | 142 | 143 |
| .EVEN | 144 | 145 | 146 | 147 | 148 | 149 | 150 | 151 | | | | | | | |
| .IF | 255 | 526 | 2254 | 2734 | 2781 | 2968 | | | | | | | | | |
| | 13 | 32 | 33 | 34 | 35 | 36 | 58 | 124 | 152 | 174 | 177 | 179 | 187 | 189 | 196 |
| | 210 | 214 | 216 | 245 | 246 | 247 | 251 | 252 | 254 | 277 | 285 | 288 | 291 | 294 | 295 |
| | 296 | 297 | 298 | 299 | 301 | 406 | 409 | 414 | 416 | 418 | 420 | 422 | 424 | 425 | 427 |
| | 445 | 511 | 512 | 513 | 516 | 525 | 529 | 531 | 533 | 534 | 535 | 583 | 585 | 590 | 592 |
| | 593 | 635 | 637 | 639 | 640 | 645 | 656 | 658 | 660 | 661 | 666 | 678 | 680 | 684 | 686 |
| | 687 | 701 | 704 | 716 | 718 | 720 | 736 | 741 | 752 | 754 | 756 | 772 | 777 | 788 | 790 |
| | 792 | 808 | 813 | 824 | 826 | 828 | 848 | 856 | 866 | 868 | 870 | 887 | 892 | 903 | 905 |
| | 907 | 930 | 935 | 946 | 948 | 950 | 966 | 971 | 984 | 986 | 988 | 1007 | 1011 | 1023 | 1025 |
| | 1027 | 1046 | 1050 | 1062 | 1064 | 1066 | 1085 | 1089 | 1101 | 1103 | 1105 | 1124 | 1128 | 1140 | 1142 |
| | 1144 | 1163 | 1167 | 1179 | 1181 | 1183 | 1202 | 1206 | 1218 | 1220 | 1222 | 1241 | 1245 | 1257 | 1259 |
| | 1261 | 1280 | 1284 | 1296 | 1298 | 1300 | 1305 | 1319 | 1321 | 1323 | 1337 | 1353 | 1368 | 1370 | 1383 |
| | 1385 | 1387 | 1392 | 1406 | 1407 | 1409 | 1423 | 1425 | 1427 | 1432 | 1446 | 1447 | 1452 | 1466 | 1468 |
| | 1470 | 1475 | 1489 | 1490 | 1492 | 1506 | 1508 | 1510 | 1515 | 1529 | 1530 | 1532 | 1544 | 1546 | 1548 |
| | 1561 | 1565 | 1577 | 1579 | 1581 | 1588 | 1599 | 1601 | 1603 | 1604 | 1608 | 1620 | 1622 | 1624 | 1625 |
| | 1631 | 1643 | 1645 | 1647 | 1648 | 1653 | 1669 | 1671 | 1673 | 1702 | 1704 | 1706 | 1730 | 1766 | 1768 |
| | 1770 | 1811 | 1813 | 1815 | 1861 | 1863 | 1865 | 1911 | 1913 | 1915 | 1930 | 1941 | 1943 | 1945 | 1946 |
| | 1957 | 1960 | 1973 | 1975 | 1977 | 1991 | 2002 | 2004 | 2006 | 2051 | 2053 | 2061 | 2063 | 2086 | 2094 |
| | 2096 | 2098 | 2105 | 2106 | 2107 | 2108 | 2109 | 2110 | 2112 | 2118 | 2121 | 2123 | 2127 | 2129 | 2135 |
| | 2137 | 2138 | 2160 | 2163 | 2174 | 2177 | 2184 | 2186 | 2187 | 2189 | 2196 | 2200 | 2207 | 2208 | 2210 |
| | 2225 | 2241 | 2257 | 2260 | 2265 | 2271 | 2272 | 2284 | 2286 | 2287 | 2288 | 2290 | 2291 | 2292 | 2301 |
| | 2303 | 2311 | 2313 | 2318 | 2319 | 2320 | 2322 | 2324 | 2325 | 2326 | 2354 | 2393 | 2394 | 2422 | 2430 |
| | 2431 | 2435 | 2436 | 2452 | 2453 | 2459 | 2461 | 2538 | 2605 | 2626 | 2684 | 2686 | 2689 | 2716 | 2731 |
| | 2741 | 2751 | 2752 | 2757 | 2764 | 2765 | 2773 | 2775 | 2779 | 2865 | 2871 | 2875 | 2886 | 2895 | 2896 |
| | 2897 | 2898 | 2899 | 2900 | 2902 | 2903 | 2904 | 2905 | | | | | | | |
| .IFF | 32 | 34 | 35 | 36 | 58 | 175 | 179 | 181 | 188 | 190 | 197 | 211 | 214 | 217 | 245 |
| | 252 | 255 | 414 | 511 | 512 | 529 | 530 | 531 | 532 | 533 | 584 | 585 | 591 | 592 | 636 |
| | 637 | 638 | 639 | 640 | 646 | 657 | 658 | 659 | 660 | 661 | 667 | 679 | 680 | 685 | 686 |

| | | | | | | | | | | | | | | | |
|-------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| | 702 | 705 | 717 | 718 | 719 | 720 | 737 | 742 | 753 | 754 | 755 | 756 | 773 | 778 | 789 |
| | 790 | 791 | 792 | 809 | 814 | 825 | 826 | 827 | 828 | 849 | 857 | 867 | 868 | 869 | 870 |
| | 888 | 893 | 904 | 905 | 906 | 907 | 931 | 936 | 947 | 948 | 949 | 950 | 967 | 972 | 985 |
| | 986 | 987 | 988 | 1008 | 1012 | 1024 | 1025 | 1026 | 1027 | 1047 | 1051 | 1063 | 1064 | 1065 | 1066 |
| | 1086 | 1090 | 1102 | 1103 | 1104 | 1105 | 1125 | 1129 | 1141 | 1142 | 1143 | 1144 | 1164 | 1168 | 1180 |
| | 1181 | 1182 | 1183 | 1203 | 1207 | 1219 | 1220 | 1221 | 1222 | 1242 | 1246 | 1258 | 1259 | 1260 | 1261 |
| | 1281 | 1285 | 1297 | 1298 | 1299 | 1300 | 1306 | 1320 | 1321 | 1322 | 1323 | 1338 | 1354 | 1369 | 1371 |
| | 1384 | 1385 | 1386 | 1387 | 1407 | 1410 | 1420 | 1424 | 1425 | 1426 | 1427 | 1447 | 1453 | 1467 | 1468 |
| | 1469 | 1470 | 1490 | 1493 | 1503 | 1507 | 1508 | 1509 | 1510 | 1530 | 1533 | 1543 | 1545 | 1546 | 1547 |
| | 1548 | 1562 | 1566 | 1578 | 1579 | 1580 | 1581 | 1589 | 1600 | 1601 | 1602 | 1603 | 1609 | 1621 | 1622 |
| | 1623 | 1624 | 1632 | 1644 | 1645 | 1646 | 1647 | 1654 | 1670 | 1671 | 1672 | 1673 | 1703 | 1704 | 1705 |
| | 1706 | 1731 | 1767 | 1768 | 1769 | 1770 | 1812 | 1813 | 1814 | 1815 | 1862 | 1863 | 1864 | 1865 | 1912 |
| | 1913 | 1914 | 1915 | 1931 | 1942 | 1943 | 1944 | 1945 | 1958 | 1961 | 1974 | 1975 | 1976 | 1977 | 1992 |
| | 2003 | 2004 | 2005 | 2006 | 2052 | 2053 | 2062 | 2063 | 2087 | 2095 | 2096 | 2097 | 2098 | 2106 | 2109 |
| | 2113 | 2119 | 2122 | 2137 | 2161 | 2163 | 2177 | 2207 | 2208 | 2211 | 2226 | 2255 | 2258 | 2285 | 2288 |
| | 2289 | 2292 | 2319 | 2320 | 2323 | 2326 | 2394 | 2396 | 2401 | 2422 | 2423 | 2432 | 2436 | 2453 | 2462 |
| | 2539 | 2606 | 2685 | 2742 | 2758 | 2775 | 2866 | 2872 | | | | | | | |
| .IFT | 526 | 2187 | 2300 | 2396 | 2401 | | | | | | | | | | |
| .IFTF | 526 | 2186 | 2298 | 2341 | 2394 | 2397 | | | | | | | | | |
| .IIF | 12 | 17 | 22 | 23 | 29 | 30 | 31 | 32 | 35 | 36 | 251 | 255 | 415 | 418 | 424 |
| | 425 | 427 | 428 | 512 | 2107 | 2113 | 2114 | 2125 | 2137 | 2141 | 2164 | 2165 | 2166 | 2167 | 2168 |
| | 2173 | 2199 | 2207 | 2208 | 2223 | 2248 | 2261 | 2262 | 2263 | 2264 | 2265 | 2266 | 2270 | 2299 | 2300 |
| | 2316 | 2319 | 2320 | 2323 | 2344 | 2445 | 2453 | 2459 | 2682 | 2894 | 2895 | 2896 | 2897 | 2898 | 2900 |
| | 2902 | 2903 | 2904 | | | | | | | | | | | | |
| .IRP | 406 | 529 | 583 | 635 | 656 | 678 | 716 | 752 | 788 | 824 | 866 | 903 | 946 | 984 | 1023 |
| | 1062 | 1101 | 1140 | 1179 | 1218 | 1257 | 1296 | 1319 | 1383 | 1423 | 1466 | 1506 | 1544 | 1577 | 1599 |
| | 1620 | 1643 | 1669 | 1702 | 1766 | 1811 | 1861 | 1911 | 1941 | 1973 | 2002 | 2051 | 2094 | 2271 | 2549 |
| | 2589 | 2690 | 2691 | 2712 | 2728 | 2729 | 2745 | 2751 | 2764 | 2765 | | | | | |
| .LIST | 2 | 35 | 45 | 166 | 245 | 252 | 255 | 406 | 429 | 512 | 513 | 526 | 529 | 533 | 555 |
| | 558 | 562 | 564 | 567 | 570 | 573 | 575 | 583 | 592 | 609 | 612 | 616 | 618 | 623 | 626 |
| | 630 | 632 | 635 | 639 | 647 | 650 | 653 | 655 | 656 | 660 | 668 | 671 | 674 | 676 | 678 |
| | 686 | 692 | 695 | 699 | 701 | 706 | 709 | 713 | 715 | 716 | 720 | 728 | 731 | 734 | 736 |
| | 743 | 746 | 749 | 751 | 752 | 756 | 764 | 767 | 770 | 772 | 779 | 782 | 785 | 787 | 788 |
| | 792 | 800 | 803 | 806 | 808 | 815 | 818 | 821 | 823 | 824 | 828 | 840 | 843 | 846 | 848 |
| | 858 | 861 | 864 | 866 | 870 | 879 | 882 | 885 | 887 | 894 | 897 | 900 | 902 | 903 | 907 |
| | 922 | 925 | 928 | 930 | 937 | 940 | 943 | 945 | 946 | 950 | 958 | 961 | 964 | 966 | 973 |
| | 976 | 979 | 981 | 984 | 988 | 999 | 1002 | 1005 | 1007 | 1013 | 1016 | 1019 | 1021 | 1023 | 1027 |
| | 1038 | 1041 | 1044 | 1046 | 1052 | 1055 | 1058 | 1060 | 1062 | 1066 | 1077 | 1080 | 1083 | 1085 | 1091 |
| | 1094 | 1097 | 1099 | 1101 | 1105 | 1116 | 1119 | 1122 | 1124 | 1130 | 1133 | 1136 | 1138 | 1140 | 1144 |
| | 1155 | 1158 | 1161 | 1163 | 1169 | 1172 | 1175 | 1177 | 1179 | 1183 | 1194 | 1197 | 1200 | 1202 | 1208 |
| | 1211 | 1214 | 1216 | 1218 | 1222 | 1233 | 1236 | 1239 | 1241 | 1247 | 1250 | 1253 | 1255 | 1257 | 1261 |
| | 1272 | 1275 | 1278 | 1280 | 1286 | 1289 | 1292 | 1294 | 1296 | 1300 | 1307 | 1310 | 1316 | 1318 | 1319 |
| | 1323 | 1329 | 1332 | 1335 | 1337 | 1345 | 1348 | 1351 | 1353 | 1360 | 1363 | 1366 | 1368 | 1372 | 1375 |
| | 1378 | 1380 | 1383 | 1387 | 1398 | 1401 | 1404 | 1406 | 1411 | 1414 | 1417 | 1419 | 1423 | 1427 | 1438 |
| | 1441 | 1444 | 1446 | 1454 | 1457 | 1460 | 1462 | 1466 | 1470 | 1481 | 1484 | 1487 | 1489 | 1494 | 1497 |
| | 1500 | 1502 | 1506 | 1510 | 1521 | 1524 | 1527 | 1529 | 1534 | 1537 | 1540 | 1542 | 1544 | 1548 | 1553 |
| | 1556 | 1559 | 1561 | 1567 | 1570 | 1573 | 1575 | 1577 | 1581 | 1590 | 1593 | 1596 | 1598 | 1599 | 1603 |
| | 1610 | 1613 | 1617 | 1619 | 1620 | 1624 | 1633 | 1636 | 1640 | 1642 | 1643 | 1647 | 1655 | 1658 | 1662 |
| | 1664 | 1669 | 1673 | 1686 | 1689 | 1692 | 1694 | 1702 | 1706 | 1718 | 1721 | 1724 | 1726 | 1749 | 1752 |
| | 1755 | 1757 | 1766 | 1770 | 1791 | 1794 | 1797 | 1799 | 1811 | 1815 | 1843 | 1846 | 1851 | 1853 | 1861 |
| | 1865 | 1890 | 1893 | 1897 | 1899 | 1911 | 1915 | 1932 | 1935 | 1938 | 1940 | 1941 | 1945 | 1962 | 1965 |
| | 1970 | 1972 | 1973 | 1977 | 1993 | 1996 | 1999 | 2001 | 2002 | 2006 | 2031 | 2034 | 2037 | 2039 | 2051 |
| | 2063 | 2077 | 2080 | 2084 | 2086 | 2094 | 2098 | 2113 | 2129 | 2207 | 2265 | 2422 | 2820 | 2823 | 2856 |
| | 2858 | 2886 | 2894 | 2895 | 2896 | 2897 | 2898 | 2899 | 2900 | 2901 | 2902 | 2903 | 2904 | 2905 | |

| | | | | | | | | | | | | | | | |
|--------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| .MACRO | 36 | 208 | 317 | 366 | 367 | 368 | 369 | 370 | 371 | 445 | 582 | 677 | 982 | 1380 | 1816 |
| | 2050 | 2886 | | | | | | | | | | | | | |
| .MCALL | 5 | 6 | 7 | 8 | 9 | 166 | 252 | 429 | 513 | | | | | | |
| .MEXIT | 300 | | | | | | | | | | | | | | |
| .MLIST | 1 | 35 | 45 | 166 | 245 | 252 | 255 | 406 | 429 | 512 | 513 | 526 | 529 | 533 | 555 |
| | 558 | 562 | 564 | 567 | 570 | 573 | 575 | 583 | 592 | 609 | 612 | 616 | 618 | 623 | 626 |
| | 630 | 632 | 635 | 639 | 647 | 650 | 653 | 655 | 656 | 660 | 668 | 671 | 674 | 676 | 678 |
| | 686 | 692 | 695 | 699 | 701 | 706 | 709 | 713 | 715 | 716 | 720 | 728 | 731 | 734 | 736 |
| | 743 | 746 | 749 | 751 | 752 | 756 | 764 | 767 | 770 | 772 | 779 | 782 | 785 | 787 | 788 |
| | 792 | 800 | 803 | 806 | 808 | 815 | 818 | 821 | 823 | 824 | 828 | 840 | 843 | 846 | 848 |
| | 858 | 861 | 864 | 866 | 870 | 879 | 882 | 885 | 887 | 894 | 897 | 900 | 902 | 903 | 907 |
| | 922 | 925 | 928 | 930 | 937 | 940 | 943 | 945 | 946 | 950 | 958 | 961 | 964 | 966 | 973 |
| | 976 | 979 | 981 | 984 | 988 | 999 | 1002 | 1005 | 1007 | 1013 | 1016 | 1019 | 1021 | 1023 | 1027 |
| | 1038 | 1041 | 1044 | 1046 | 1052 | 1055 | 1058 | 1060 | 1062 | 1066 | 1077 | 1080 | 1083 | 1085 | 1091 |
| | 1094 | 1097 | 1099 | 1101 | 1105 | 1116 | 1119 | 1122 | 1124 | 1130 | 1133 | 1136 | 1138 | 1140 | 1144 |
| | 1155 | 1158 | 1161 | 1163 | 1169 | 1172 | 1175 | 1177 | 1179 | 1183 | 1194 | 1197 | 1200 | 1202 | 1208 |
| | 1211 | 1214 | 1216 | 1218 | 1222 | 1233 | 1236 | 1239 | 1241 | 1247 | 1250 | 1253 | 1255 | 1257 | 1261 |
| | 1272 | 1275 | 1278 | 1280 | 1286 | 1289 | 1292 | 1294 | 1296 | 1300 | 1307 | 1310 | 1316 | 1318 | 1319 |
| | 1323 | 1329 | 1332 | 1335 | 1337 | 1345 | 1348 | 1351 | 1353 | 1360 | 1363 | 1366 | 1368 | 1372 | 1375 |
| | 1378 | 1380 | 1383 | 1387 | 1398 | 1401 | 1404 | 1406 | 1411 | 1414 | 1417 | 1419 | 1423 | 1427 | 1438 |
| | 1441 | 1444 | 1446 | 1454 | 1457 | 1460 | 1462 | 1466 | 1470 | 1481 | 1484 | 1487 | 1489 | 1494 | 1497 |
| | 1500 | 1502 | 1506 | 1510 | 1521 | 1524 | 1527 | 1529 | 1534 | 1537 | 1540 | 1542 | 1544 | 1548 | 1553 |
| | 1556 | 1559 | 1561 | 1567 | 1570 | 1573 | 1575 | 1577 | 1581 | 1590 | 1593 | 1596 | 1598 | 1599 | 1603 |
| | 1610 | 1613 | 1617 | 1619 | 1620 | 1624 | 1633 | 1636 | 1640 | 1642 | 1643 | 1647 | 1655 | 1658 | 1662 |
| | 1664 | 1669 | 1673 | 1686 | 1689 | 1692 | 1694 | 1702 | 1706 | 1718 | 1721 | 1724 | 1726 | 1749 | 1752 |
| | 1755 | 1757 | 1766 | 1770 | 1791 | 1794 | 1797 | 1799 | 1811 | 1815 | 1843 | 1846 | 1851 | 1853 | 1861 |
| | 1865 | 1890 | 1893 | 1897 | 1899 | 1911 | 1915 | 1932 | 1935 | 1938 | 1940 | 1941 | 1945 | 1962 | 1965 |
| | 1970 | 1972 | 1973 | 1977 | 1993 | 1996 | 1999 | 2001 | 2002 | 2006 | 2031 | 2034 | 2037 | 2039 | 2051 |
| | 2063 | 2077 | 2080 | 2084 | 2086 | 2094 | 2098 | 2113 | 2129 | 2207 | 2265 | 2422 | 2820 | 2823 | 2856 |
| | 2858 | 2886 | 2894 | 2895 | 2896 | 2897 | 2898 | 2899 | 2900 | 2901 | 2902 | 2903 | 2904 | 2905 | |
| .PAGE | 38 | 301 | | | | | | | | | | | | | |
| .REPT | 45 | | | | | | | | | | | | | | |
| .SBTTL | 25 | 37 | 56 | 172 | 185 | 208 | 252 | 301 | 372 | 404 | 408 | 508 | 513 | 529 | 583 |
| | 635 | 656 | 678 | 716 | 752 | 788 | 824 | 866 | 903 | 946 | 984 | 1023 | 1062 | 1101 | 1140 |
| | 1179 | 1218 | 1257 | 1296 | 1319 | 1383 | 1423 | 1466 | 1506 | 1544 | 1577 | 1599 | 1620 | 1643 | 1665 |
| | 1666 | 1667 | 1669 | 1702 | 1766 | 1807 | 1808 | 1809 | 1811 | 1861 | 1911 | 1941 | 1973 | 2002 | 2051 |
| | 2094 | 2101 | 2103 | 2158 | 2208 | 2255 | 2320 | 2459 | 2536 | 2603 | 2682 | 2739 | 2863 | 2886 | 2906 |
| .TITLE | 12 | | | | | | | | | | | | | | |
| .WORD | 45 | 46 | 48 | 49 | 51 | 180 | 201 | 202 | 203 | 204 | 205 | 206 | 216 | 219 | 220 |
| | 221 | 222 | 225 | 226 | 227 | 228 | 229 | 230 | 231 | 234 | 235 | 236 | 257 | 258 | 259 |
| | 260 | 261 | 262 | 263 | 264 | 268 | 269 | 270 | 283 | 287 | 290 | 293 | 294 | 295 | 296 |
| | 297 | 298 | 376 | 377 | 378 | 379 | 380 | 381 | 382 | 383 | 384 | 385 | 386 | 387 | 388 |
| | 389 | 393 | 394 | 395 | 396 | 398 | 399 | 400 | 401 | 851 | 2118 | 2121 | 2136 | 2234 | 2239 |
| | 2535 | 2632 | 2679 | 2714 | 2774 | 2860 | 2861 | 2893 | 2970 | 2972 | 2974 | 2976 | 2979 | 2982 | 2985 |

ERRORS DETECTED: 0

*DVIBA, DVIBA/SOL/CRF=DVIBA
 RUN-TIME: 95 39 7 SECONDS
 CORE USED: 25K

L07