

KPV11-A

DIAGNOSTICS
MD-11-DVKPA-A

EP DVKPA DL A
COPYRIGHT 1977
FICHE 1 OF 1

JUN 1979
digital
MADE IN USA

The image displays a grid of 50 diagnostic data sheets for the KPV11-A system, arranged in 10 rows and 5 columns. Each sheet contains technical information such as test procedures, error codes, and component specifications. The sheets are organized into sections, with some containing tables of data and others providing detailed instructions or diagrams. The text on the sheets is small and difficult to read, but the overall layout is structured and systematic.

IDENTIFICATION
.....

SEQ 0001

Product Code: MAINDEC-11-DVKPA-A-D
Product Name: KPV11-A Diagnostic
Date Created: January 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.

(5)

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
6.0	ERRORS
6.1	Error Printout
6.2	Non-Standard Error Halts
7.0	RESTRICTIONS
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
9.0	PROGRAM DESCRIPTION

1.0 ABSTRACT

This program allows the user to checkout or debug the KPV11-A, LSI-11 Power Fail/Line Time Generator. To check-out Power fail, Non-Volatile Memory (i.e., core) must be located in the first 4K memory. The user should check the jumpers on the option as well as jumpers on the LSI-11 module.

Even though the KPV-11A is a LSI-11 option, this program was designed to run on any PDP-11 Family Computer. 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 core memory (or more) and console I/O facilities (i.e., TTY). If CPU is an LSI-11, Power-up option #1 should be jumpered on CPU module.
2. KPV-11 under test.

The following jumpers must be installed or removed (Standard Factory Configuration):

INSTALLED -----	REMOVED -----
W1, W2, W3	W6, W9, W10
W4, W5, W7	W12
W8, W11, W13	
W14, W15	

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 KPV11-A with a standard address and vector. If any of these are different on the KPV11-A or KD11 you are testing, change the corresponding location in memory before starting this test.

TAG	ADDRESS	CURRENT CONTENTS	COMMENTS
---	-----	-----	-----
SBASE	001246	177846	;;BASE ADDRESS OF EQUIPMENT
SVECT1	001242	000100	;;INTERRUPT VECTOR #1
SSWREG	000176	000000	;;MANUAL SWR
noclk	001370	000000	;;indicates if clock to be tested, ;;=0 test clock;=1 no clock test.

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 "ODT".
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
08	000400	LOOP ON TEST IN SWR <710>

NOTE

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

5.2 Scope Loops

If an error occurs and the user wishes to scope the error, "SSWREG" 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 "ODT", "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-DVKPA-A
SWR=000000 NEWS"

2. Program waits for operator to enter a switch register setting. If the program is restarted, no timeout will occur to change the Switch Register setting, see section 8.6.
3. Program executes a pass of logic tests, subtest iterations inhibited.
4. Program reports any errors it detects as well as any power failures that occur.
5. Program reports "END PASS XX POWER FAILS YY" where XX is the total passes completed and YY is the total number of power fails detected while the program has run. Both numbers are octal.
6. Program continues execution of steps 3-5 only with subtest iterations (unless inhibited by operator).

NOTE

Power failures CAN NOT be generated by this program. The operator MUST generate them. They can be inflicted only after step 02 has been executed.

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

None.

8.0 MISCELLANEOUS
-----**8.1 Power Fail**

After a power failure occurs, the program execution will continue at the point where the power failure occurred. The program will type "PF-OK" if no messages were being typed out. If a message was being typed out at the time of the power fail, it will complete typeout of that message.

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
0.75 Minutes (50 sec) With Iterations - No Errors

0.4 LSI-11 "ODT" Commands

FORMAT	DESCRIPTION
<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).
RIG 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 KD11 (LSI-11 CPU) can be entered in four different ways (others are a subset of these) from the run state:

1. Execution of an 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 commands listed in section 8.4.

8.6 Use of Program Software SWR

The software switch register may be changed while the program is running by typing "G (control and letter G keys typed simultaneously). When "G is typed, the program responds by typing "SWRXXXXX" where XXXXX 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 "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,R0 (i.e., Location 300 would contain 302 and location 302 would contain 4700).

When a device unexpectedly 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 set, would prevent further interrupts from happening. When the CPU attempts to execute "4700" (JSR PC,R0), 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 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".

9.0 PROGRAM DESCRIPTION

-
- TST1** In this test we write all available memory from the end of the program to address 17470 with either a zero pattern or a ones pattern depending on whether we are on an even pass number or odd pass number. We restrict memory checking to the lower 4K of memory since other memory could be volatile. "MILIM" (address 1342) may be changed to reflect an end address greater than 17470 if more non-volatile memory is in your system.
- After writing memory, it is read back to assure that it could be written correctly. If not, an error is reported.
- During the memory write period, power failures could occur. Appropriate software flags are set while writing memory so that the power up routine can check memory after a power up without confusing the pattern written in memory.
- TST2** Here we check to see that the clock will respond with "B RPLY" when addressed. This test is in two parts: Part One addresses the clock with "B DIN"; Part Two addresses the clock with "B DOUT". If this test fails, no further tests will be performed. The operator must fix the error. He may loop on test two.
- TST3** Here we try setting, then clearing bit 6 (interrupt enable) of the Control and Status Register (CSR). If a power failure occurs during this test, the test will be performed again.
- TST4** In this test we wait to see if the clock CSR bit 7 will set, interrupt enable is also set. After CSR bit 7 sets, we expect an interrupt to occur.
- TST5** This test is designed to see if we can clear CSR bit 7. To do this, we first must allow the clock to interrupt, assuring that the flag just set. Then we can attempt to clear it by writing it to a zero.
- TST6** In this test we make sure that the clock doesn't interrupt when CSR bit 6 is clear.
- TST7** This test is the same as Part 2 of TST1. We verify that the pattern in memory is the same as we wrote there. Only this time we spend a lot of time reading memory, in hopes of catching a power failure while reading it.

PDOWN

Power down routine. This routine is entered when a Power down sequence has occurred. The first thing we check (if this is not the first power down) is to see if the last power down had had enough time to complete this routine. If not, we halt. Next we set a flag to indicate that we started this power down routine. Now we save all general registers, as well as error message locations in case we were reporting an error before the power failure occurred. A pointer to the power up routine is put in location 24 so that on power up, program control is transferred there. Next we clear the power down flag indicating that this power down was successfully completed. Last we wait for power to go away while doing a Branch-self. If "POWER OK" isn't asserted before the power goes away, we stand a good chance at changing memory locations since memory was cycling during a power loss.

PUP

Routine to handle power recoveries

The first thing we check for is to make sure the power down routine had had enough time to complete. Next we put a pointer to a routine that will take care of the case where we didn't have enough time to start our power up routine. Now we check to see that we didn't experience 5 power failures without completely finishing one. In that case the stack pointer is getting dangerously low and we couldn't put anymore information with it. With the preliminaries done, we put a pointer to our power down routine into location 24. Next we'll verify the contents of our writable memory, checking the flags to be sure that we weren't writing a new pattern into memory.

When the power failure occurred, Flag "BLKCK" was set when we changed pass numbers but not pattern. When the new pattern was written into at least one location, BLKCK is cleared. Flag "MEMWR" is set when we started writing a new pattern and cleared when we fully have written a new pattern. If MEMWR is set, we will get the last address to check from "LMAW", thus checking only memory that was written with the new pattern.

Now we will check to see that the clock CSR will initialize properly. First, however, we check to make sure that the clock passes "TST2" (the addressing test). If it didn't we won't touch the clock here since we can't afford any time-out traps if the clock should prove incapable of passing that test.

Next we restore all general registers and other information that was stored on the stack when the power failure occurred.

Last we check to see if a message was being typed out at the time the power failure occurred.

If no message was being typed, we will type: "PF-OK". If a message was being typed, we will back up the ASCII string pointer to pick up the character that was lost when the power failure occurred, then continue to type out the message. NOTE: If the power failure did not occur in the serial line unit controlling the console I/O device, a duplicate character will be typed of the last character typed before the power fail that caused the CPU to go away.

15	OPERATIONAL SWITCH SETTINGS
27	TRAP CATCHER
46	BASIC DEFINITIONS
162	ACT11 HOOKS
173	APT PARAMETER BLOCK
195	COMMON TAGS
239	APT MAILBOX-ETABLE
288	ERROR POINTER TABLE
374	INITIALIZE THE COMMON TAGS
431	TYPE PROGRAM NAME
436	GET VALUE FOR SOFTWARE SWITCH REGISTER
455	T1 *WRITE CORE WITH MEMORY PATTERN
519	T2 *TEST THAT THE CLOCK GIVES "B RPLY" ON "DIN" AND "DOUT"
556	T3 *TEST THAT BITS (INTR, ENBL) CAN BE SET AND CLEARED
615	T4 *TEST THAT THE CLOCK READY FLAG WILL SET AND THE CLOCK INTERRUPTS
676	T5 *TEST THAT WE CAN CLEAR READY FLAG
725	T6 *TEST THAT THE CLOCK DOESN'T INTR, WHEN INTR, ENABLE CLEAR
763	T7 *VERIFY MEMORY LOCATIONS WRITTEN IN TEST1
797	T10 *END OF TESTS
804	END OF PASS ROUTINE
929	MEMORY TEST ON POWER UP,
961	CLOCK CHECK AFTER POWER UP
1001	ERROR HANDLER ROUTINE
1051	ERROR MESSAGE TIMEOUT ROUTINE
1098	SCOPE HANDLER ROUTINE
1164	TTY INPUT ROUTINE
1303	BINARY TO OCTAL (ASCII) AND TYPE
1380	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
1447	TYPE ROUTINE
1526	APT COMMUNICATIONS ROUTINE
1658	TRAP DECODER
1681	TRAP TABLE
1701	ASCII MESSAGES

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

000001

000000

000004 000004 000200

000174 000000

000176 000000 000100 000200 000002

000200 000137 001374

001100

000011
000012
000015

```

;TITLE MAINDEC-11-DVKPA-A
;COPYRIGHT (C) 1977
;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.
;
STN=1
    
```

```

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

.SBTTL 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 BUS ERROR.
;=174
DISPREG: ;WORD 0 ;SOFTWARE DISPLAY REGISTER.
SWREG: ;WORD 0 ;SOFTWARE SWITCH REGISTER.
;=100
;WORD 104,200,2 ;IF "B EVENT" ON Q-BUS IS
;CONNECTED, WE NEED A WAY OF
;IGNORING ITS INTERRUPTS.
;=200
JMP START
    
```

.SBTTL BASIC DEFINITIONS

```

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

.SBTTL MISCELLANEOUS DEFINITIONS

```

HT= 11 ;CODE FOR HORIZONTAL TAB
LF= 12 ;CODE FOR LINE FEED
CR= 15 ;CODE FOR CARRIAGE RETURN
    
```

55	000200	CRLF= 200	;;CODE FOR CARRIAGE RETURN-LINE FEED
56	177776	PS= 177776	;;PROCESSOR STATUS WORD
57		,EQUIV PS,PSW	
58	177774	STKLMT= 177774	;;STACK LIMIT REGISTER
59	177772	PIRQ= 177772	;;PROGRAM INTERRUPT REQUEST REGISTER
60	177570	DSWR= 177570	;;HARDWARE SWITCH REGISTER
61	177570	DDISP= 177570	;;HARDWARE DISPLAY REGISTER
62			
63		; *GENERAL PURPOSE REGISTER DEFINITIONS	
64	000000	R0= 00	;;GENERAL REGISTER
65	000001	R1= 01	;;GENERAL REGISTER
66	000002	R2= 02	;;GENERAL REGISTER
67	000003	R3= 03	;;GENERAL REGISTER
68	000004	R4= 04	;;GENERAL REGISTER
69	000005	R5= 05	;;GENERAL REGISTER
70	000006	R6= 06	;;GENERAL REGISTER
71	000007	R7= 07	;;GENERAL REGISTER
72	000006	SP= 06	;;STACK POINTER
73	000007	PC= 07	;;PROGRAM COUNTER
74			
75		; *PRIORITY LEVEL DEFINITIONS	
76	000000	PR0= 0	;;PRIORITY LEVEL 0
77	000040	PR1= 40	;;PRIORITY LEVEL 1
78	000100	PR2= 100	;;PRIORITY LEVEL 2
79	000140	PR3= 140	;;PRIORITY LEVEL 3
80	000200	PR4= 200	;;PRIORITY LEVEL 4
81	000240	PR5= 240	;;PRIORITY LEVEL 5
82	000300	PR6= 300	;;PRIORITY LEVEL 6
83	000340	PR7= 340	;;PRIORITY LEVEL 7
84			
85		; *SWITCH REGISTER SWITCH DEFINITIONS	
86	100000	SW15= 100000	
87	040000	SW14= 40000	
88	020000	SW13= 20000	
89	010000	SW12= 10000	
90	004000	SW11= 4000	
91	002000	SW10= 2000	
92	001000	SW09= 1000	
93	000400	SW08= 400	
94	000200	SW07= 200	
95	000100	SW06= 100	
96	000040	SW05= 40	
97	000020	SW04= 20	
98	000010	SW03= 10	
99	000004	SW02= 4	
100	000002	SW01= 2	
101	000001	SW00= 1	
102		,EQUIV SW09,SW9	
103		,EQUIV SW08,SW8	
104		,EQUIV SW07,SW7	
105		,EQUIV SW06,SW6	
106		,EQUIV SW05,SW5	
107		,EQUIV SW04,SW4	
108		,EQUIV SW03,SW3	

```
109      ,EQUIV SW02,SW2
110      ,EQUIV SW01,SW1
111      ,EQUIV SW00,SW0
112
113      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
114      100000      BIT15= 100000
115      040000      BIT14= 40000
116      020000      BIT13= 20000
117      010000      BIT12= 10000
118      004000      BIT11= 4000
119      002000      BIT10= 2000
120      001000      BIT09= 1000
121      000400      BIT08= 400
122      000200      BIT07= 200
123      000100      BIT06= 100
124      000040      BIT05= 40
125      000020      BIT04= 20
126      000010      BIT03= 10
127      000004      BIT02= 4
128      000002      BIT01= 2
129      000001      BIT00= 1
130
131      ,EQUIV BIT09,BIT9
132      ,EQUIV BIT08,BIT8
133      ,EQUIV BIT07,BIT7
134      ,EQUIV BIT06,BIT6
135      ,EQUIV BIT05,BIT5
136      ,EQUIV BIT04,BIT4
137      ,EQUIV BIT03,BIT3
138      ,EQUIV BIT02,BIT2
139      ,EQUIV BIT01,BIT1
140      ,EQUIV BIT00,BIT0
141
142      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
143      000004      ERRVEC= 4           ;; TIME OUT AND OTHER ERRORS
144      000010      RESVEC= 10        ;; RESERVED AND ILLEGAL INSTRUCTIONS
145      000014      TBITVEC=14        ;; "T" BIT
146      000014      TRTVEC= 14        ;; TRACE TRAP
147      000014      BPTVEC= 14        ;; BREAKPOINT TRAP (BPT)
148      000020      IOTVEC= 20        ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
149      000024      PWRVEC= 24        ;; POWER FAIL
150      000030      ENTVEC= 30        ;; EMULATOR TRAP (ENT) **ERROR**
151      000034      TRAPVEC=34        ;; "TRAP" TRAP
152      000060      TRVEC= 60         ;; TTY KEYBOARD VECTOR
153      000064      TPVEC= 64         ;; TTY PRINTER VECTOR
154      000240      PIRGVEC=240       ;; PROGRAM INTERRUPT REQUEST VECTOR
155
156      177546      ABASE= 177546
157      000100      AVECT1= 100
158      000001      STN=1
159
160      ,SBTTL ACT11 HOOKS
161
162      ;;*****
```


193
194
195
196
197
198
199 001100
200 001100 001100
201 001100 000000
202 001102 000
203 001103 000
204 001104 000000
205 001106 000000
206 001110 000000
207 001112 000000
208 001114 000
209 001115 001
210 001116 000000
211 001120 000000
212 001122 000000
213 001124 000000
214 001126 000000
215 001130 000000
216 001132 000000
217 001134 000
218 001135 000
219 001136 000000
220 001140 177570
221 001142 177570
222 001144 177560
223 001146 177562
224 001150 177564
225 001152 177566
226 001154 000
227 001155 002
228 001156 012
229 001157 000
230 001160 000000
231 001162 000000
232 001164 177607 000377
233 001170 077
234 001171 015
235 001172 000012
236
237
238
239
240
241 001174
242 001174 000000
243 001176 000000
244 001200 000000
245 001202 000000
246 001204 000000

.SBTTL COMMON TAGS

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

SCNTAG: ,=1100

STSTNM: ,WORD 0
SERFLG: ,BYTE 0
SICNT: ,WORD 0
SLPADR: ,WORD 0
SLPERR: ,WORD 0
SERTTL: ,WORD 0
SITENB: ,BYTE 0
SERMAX: ,BYTE 1
SERRPC: ,WORD 0
SGDADR: ,WORD 0
SBDADR: ,WORD 0
SGDDAT: ,WORD 0
SBD DAT: ,WORD 0
SAUTOS: ,BYTE 0
SINTAG: ,BYTE 0
SNR: ,WORD DSWR
DISPLAY: ,WORD DDISP
STKS: 177560
STKB: 177562
STPS: 177564
STPB: 177566
SHULL: ,BYTE 0
SFILLS: ,BYTE 2
SFILLC: ,BYTE 12
STPFLG: ,BYTE 0
STINES: 0
SESCAPE: 0
SBELL: ,ASCIZ <207><377><377>
SQUES: ,ASCII /?/
SCRLF: ,ASCII <15>
SLF: ,ASCIZ <12>

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

.SBTTL APT MAILBOX-ETABLE

;;*****
;EVEN
SHAIL: ;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

247	001206	000000	UNIT: ,WORD	AUNIT	;; I/O UNIT NUMBER
248	001210	000000	MSGAD: ,WORD	AMSGAD	;; MESSAGE ADDRESS
249	001212	000000	MSGLG: ,WORD	AMSGLG	;; MESSAGE LENGTH
250	001214		ETABLE:		;; APT ENVIRONMENT TABLE
251	001214	000	ENV: ,BYTE	AENV	;; ENVIRONMENT BYTE
252	001215	000	ENVH: ,BYTE	AENVH	;; ENVIRONMENT MODE BITS
253	001216	000000	SWREG: ,WORD	ASWREG	;; APT SWITCH REGISTER
254	001220	000000	SUSR: ,WORD	AUSR	;; USER SWITCHES
255	001222	000000	CPUOP: ,WORD	ACPUOP	;; CPU TYPE, OPTIONS
256			;		BITS 15-11=CPU TYPE
257			;		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
258			;		11/70=06, PDQ=07, Q=10
259			;		BIT 10=REAL TIME CLOCK
260			;		BIT 9=FLOATING POINT PROCESSOR
261			;		BIT 8=MEMORY MANAGEMENT
262	001224	000	ANS1: ,BYTE	ANANS1	;; HIGH ADDRESS, N, S, BYTE
263	001225	000	NTYP1: ,BYTE	ANTYP1	;; MEN, TYPE, BLK01
264			;		MEN, TYPE BYTE -- (HIGH BYTE)
265			;		900 NSEC CORE=001
266			;		300 NSEC BIPOLAR=002
267			;		500 NSEC NOS=003
268	001226	000000	ADR1: ,WORD	ANADR1	;; HIGH ADDRESS, BLK01
269			;		MEN, LAST ADDR, 03 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
270	001230	000	ANS2: ,BYTE	ANANS2	;; HIGH ADDRESS, N, S, BYTE
271	001231	000	NTYP2: ,BYTE	ANTYP2	;; MEN, TYPE, BLK02
272	001232	000000	ADR2: ,WORD	ANADR2	;; MEN, LAST ADDRESS, BLK02
273	001234	000	ANS3: ,BYTE	ANANS3	;; HIGH ADDRESS, N, S, BYTE
274	001235	000	NTYP3: ,BYTE	ANTYP3	;; MEN, TYPE, BLK03
275	001236	000000	ADR3: ,WORD	ANADR3	;; MEN, LAST ADDRESS, BLK03
276	001240	000	ANS4: ,BYTE	ANANS4	;; HIGH ADDRESS, N, S, BYTE
277	001241	000	NTYP4: ,BYTE	ANTYP4	;; MEN, TYPE, BLK04
278	001242	000000	ADR4: ,WORD	ANADR4	;; MEN, LAST ADDRESS, BLK04
279	001244	000100	VECT1: ,WORD	AVECT1	;; INTERRUPT VECTOR01, SUB PRIORITY01
280	001246	000000	VECT2: ,WORD	AVECT2	;; INTERRUPT VECTOR02, SUB PRIORITY02
281	001250	177546	BASE: ,WORD	ABASE	;; BASE ADDRESS OF EQUIPMENT UNDER TEST
282	001252	000000	DEVH: ,WORD	ADEVH	;; DEVICE MAP
283	001254	000000	CDW1: ,WORD	ACDW1	;; CONTROLLER DESCRIPTION WORD01
284	001256		ETEND:		
285			,NEXT		

23

286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
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

001256

001256 006754
001260 007165
001262 007324
001264 007364

001266 007005
001270 007165
001272 007324
001274 007364

001276 007051
001300 007222
001302 007336
001304 007364

001306 007072
001310 007247
001312 007346
001314 007364

001316 007115
001320 007247
001322 007346
001324 007364

,\$BTTL 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 \$ITEMS, THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT,
;\$NOTE1: IF \$ITEMS 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

EN1 ;MEMORY READ/WRITE ERROR
DH1 ;ERRPC ADDR WAS S/B
DT1 ;\$ERRPC,\$RADDR,\$BDDAT,\$GDDAT
DF0 ;ALL NUMBERS ARE IN OCTAL FORM

;\$ITEM 2

EN2 ;MEMORY DATA ERROR AFTER POWER FAIL
DH1 ;ERRPC ADDR WAS S/B
DT1 ;\$ERRPC,\$RADDR,\$BDDAT,\$GDDAT
DF0 ;ALL NUMBERS ARE IN OCTAL FORM

;\$ITEM 3

EN3 ;CLOCK CSR ERROR
DH3 ;ERRPC WAS S/B
DT3 ;\$ERRPC,\$BDDAT,\$GDDAT
DF0 ;ALL NUMBERS ARE IN OCTAL FORM

;\$ITEM 4

EN4 ;CLOCK ADDR ERROR
DH4 ;ERRPC ADDR
DT4 ;\$ERRPC,\$LRS
DF0 ;ALL NUMBERS ARE IN OCTAL FORM

;\$ITEM 5

EN5 ;CLOCK INTERRUPT ERROR
DH4 ;ERRPC ADDR
DT4 ;\$ERRPC,\$LRS
DF0 ;ALL NUMBERS ARE IN OCTAL FORM

;\$ITEM 6

```

340 001326 007144           EM6           ;INTERRUPT ERROR
341 001330 007268           DH6           ;TEST  ERRPC  TO      FROM
342 001332 007354           DT6           ;TSTNM,ERRPC,TRTO,TRFRO
343 001334 007364           DFO           ;ALL NUMBERS ARE IN OCTAL FORM
344
345           ;REGISTER DEFINITION, CONSTANTS, STORAGE,
346
347 001336 177546           LKS:  ,WORD   ABASE           ;>NO    <;ADDRESS OF CLOCK CSR, IF DIFFERENT
348           ;>PATCH <;CHANGE CONTENTS OF "ABASE",
349 001340 000100           KVECT: ,WORD   AVECT1         ;>NO    <;VECTOR ADDRESS OF CLOCK,
350 001342 000102           KVECTP: ,WORD  AVECT1+2       ;>PATCH <;
351
352 001344 017470           HILIN: ,WORD   017470         ;HI ADDR, CORE FOR WRITE PATTERN
353 001346 007370           LOLIN: ,WORD   MENST          ;LOW ADDR CORE FOR MEN TEST,
354 001350 007372           LON2:  ,WORD   MENST+2       ;NEXT ADDRESS,
355
356 001352 000000           PCOUNT: ,WORD   0             ;COUNTS # OF POWER FAILS,
357 001354 000000           PFLAG: ,WORD   0             ;#1 INDICATES PFAIL, 0 - NO POWER FAIL
358 001356 000000           ADCK:  ,WORD   0             ;#1 WHEN WE HAVE TESTED CLOCK ADDRESS
359 001360 000001           MENWR: ,WORD   1             ;#1 WHEN MEMORY BEING WRITTEN,
360 001362 000000           RADDR: ,WORD   0             ;RADDR, DURING MEN CHECK
361 001364 000000           BAD:   ,WORD   0             ;#1 WHEN POWER DOWN BAD,
362 001366 000000           BLKCK: ,WORD   0
363 001370 000000           LNR:   ,WORD   0             ;LAST MEMORY ADDR,WRITTEN W/NEW PATTERN,
364 001372 000000           NOCLK: ,WORD   0             ;INDICATES WHETHER THE CLOCK SHOULD BE TESTED OR NOT,
365           ;#0 THEN YES,;#1 NO,USER MAY CHANGE THIS LOC,
366
367
368 001374 013703 001346           START: NOV    LOLIN,R3         ;SET FOR MEN CHECK ON POWER UP,
369 001400 012713 177777           NOV    0-1,(3)              ;PATTERN ALL ONES
370 001404 010337 001370           NOV    R3,LNR
371 001410 005000           CLR    R0
372           ;SBTTL INITIALIZE THE COMMON TAGS
373           ;;CLEAR THE COMMON TAGS (SCNTAG) AREA
374 001412 012706 001100           NOV    @SCNTAG,R6           ;;FIRST LOCATION TO BE CLEARED
375 001416 005026           CLR    (R6)+                ;;CLEAR MEMORY LOCATION
376 001420 022706 001140           CMP    @R6,R6 ;;DONE?
377 001424 001374           BNE    ,=6                  ;;LOOP BACK IF NO
378 001426 012706 001100           NOV    @STACK,SP           ;;SETUP THE STACK POINTER
379           ;;INITIALIZE A FEW VECTORS
380 001432 012737 004354 000020           NOV    @SCOPE,@IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
381 001440 012737 000340 000022           NOV    @340,@IOTVEC+2 ;;LEVEL 7
382 001446 012737 004032 000030           NOV    @ERROR,@ENTVEC ;;ENT VECTOR FOR ERROR ROUTINE
383 001454 012737 000340 000032           NOV    @340,@ENTVEC+2 ;;LEVEL 7
384 001462 012737 006674 000034           NOV    @TRAP,@TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
385 001470 012737 000340 000036           NOV    @340,@TRAPVEC+2 ;;LEVEL 7
386 001476 005037 001160           CLR    @STINES              ;;INITIALIZE NUMBER OF ITERATIONS
387 001502 005037 001162           CLR    @ESCAPE              ;;CLEAR THE ESCAPE ON ERROR ADDRESS
388 001506 112737 000001 001115           NOV    @1,@ERNAX            ;;ALLOW ONE ERROR PER TEST
389 001514 012737 001514 001106           NOV    @,@,@LPADR           ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
390 001522 012737 001522 001110           NOV    @,@,@LPERR          ;;SETUP THE ERROR LOOP ADDRESS
391           ;;SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS
392           ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER,
393 001530 013746 000004           NOV    @ERRVEC,-(SP) ;;SAVE ERROR VECTOR

```

```

394 001534 012737 001570 000004      MOV      0640,00ERRVEC      ;;SET UP ERROR VECTOR
395 001542 012737 177570 001140      MOV      0DSWR,SWR         ;;SETUP FOR A HARDWARE SWICH REGISTER
396 001550 012737 177570 001142      MOV      0DDISP,DISPLAY    ;;AND A HARDWARE DISPLAY REGISTER
397 001556 022777 177777 177354      CMP      0-1,0SWR         ;;TRY TO REFERENCE HARDWARE SWR
398 001564 001012                    BNE      660               ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
399                                ;;AND THE HARDWARE SWR IS NOT 0-1
400 001566 000403                    BR       650               ;;BRANCH IF NO TIMEOUT
401 001570 012716 001576                640:    MOV      0600,(SP)        ;;SET UP FOR TRAP RETURN
402 001574 000002                    RTI
403 001576 012737 000176 001140 650:    MOV      0SWREG,SWR        ;;POINT TO SOFTWARE SWR
404 001604 012737 000174 001142      MOV      0DISPREG,DISPLAY
405 001612 012637 000004                660:    MOV      (SP)+,00ERRVEC    ;;RESTORE ERROR VECTOR
406
407 001616 000037 001202                    CLR      0PASS            ;;CLEAR PASS COUNT
408 001622 132737 000200 001215      BITB    0APTSIZE,0ENVH    ;;TEST USER SIZE UNDER APT
409 001630 001403                    BEQ      670               ;;YES,USE NON-APT SWITCH
410 001632 012737 001216 001140      MOV      00SWREG,SWR      ;;NO,USE APT SWITCH REGISTER
411 001640                                670:
412
413 001640 012737 003344 000024      MOV      0PDOWN,PWRVEC    ;;INITIALIZE POWER FAIL VECTOR,
414 001646 012737 000340 000026      MOV      0340,PWRVEC+2    ;;NO INTERRUPTS ALLOWED ON POWER FAIL,
415 001654 000037 001202                    CLR      0PASS            ;;CLEAR PASS COUNT
416 001660 000037 001352                    CLR      PCOUNT           ;;CLEAR POWER FAIL COUNT
417 001664 000037 001356                    CLR      ADCR             ;;CLOCK ADDR, NOT TESTED,
418 001670 000037 001364                    CLR      BAD
419
420
421 001674 012746 000340      MOV      0340,-(6)        ;/-PR-
422 001700 012746 001706      MOV      0600,-(6)        ;/PUT NEW STATUS ON STACK
423 001704 000002      RTI                      ;/PUT RETURN ADDR. ON STACK.
424                                ;/DO AN RTI CPU WILL RETURN TO NEXT
425                                ;/ADDR. WITH NEW STATUS. IF STATUS
426                                ;/00 THEN INTERRUPTS ALLOWED,
426 001706                                680:
427
428
429                                ;SBTTL TYPE PROGRAM NAME
430                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
431 001706 000227 177777      INC      0-1              ;;FIRST TIME?
432 001712 001033                    BNE      690               ;;BRANCH IF NO
433 001714 104401 001762      TYPE    ,700              ;;TYPE ASCII STRING
434                                ;SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
435 001720 000737 000042      TST     0042              ;;ARE WE RUNNING UNDER XDDP/ACT?
436 001724 001012                    BNE      710               ;;BRANCH IF YES
437 001726 123727 001214 000001      CMPS   0ENV,01           ;;ARE WE RUNNING UNDER APT?
438 001734 001406                    BEQ      710               ;;BRANCH IF YES
439 001736 023727 001140 000176      CMP     SWR,0SWREG        ;;SOFTWARE SWITCH REG SELECTED?
440 001744 001008                    BNE      720               ;;BRANCH I/ NO
441 001746 104406      GTSWR                    ;;GET SOFT-SWR SETTINGS
442 001750 000403                    BR       720
443 001752 112737 000001 001134 710:    MOV     01,0AUTOS        ;;SET AUTO-MODE INDICATOR
444 001760                                720:
445 001760 000410                    BR       690               ;;GET OVER THE ASCII
446                                ;;700:
447 002002                                690:    ,ASCII <CRLF>#ND11-DVKPA-AB<CRLF>
    
```

```

448
449
450 002002                    LOOP:
451
452
453                    ;*****
454                    ;*TEST 1            *WRITE CORE WITH MEMORY PATTERN
455                    ;*
456                    ;*IN THIS TEST WE WILL WRITE AVAILABLE CORE WILL ALL ONES OR ALL
457                    ;*ZEROS DEPENDING ON ODD OR EVEN PASS (RESPECTIVELY).
458                    ;*AFTER CORE IS WRITTEN, WE WILL VERIFY RESULTS.
459                    ;*
460                    ;*IF WE ARE IN THE PROCESS OF CHANGING CORE PATTERN DURING A POWER
461                    ;*FAIL, ONLY THOSE LOCATIONS THAT WERE WRITTEN WILL BE CHECKED.
462                    ;*
463                    ;*****
464 002002    000240            TST1:    NOP
465 002004    012737    000001    001160    MOV        #1,STIMES            ;DO 1 ITERATION
466
467 002012    112737    000001    001102            MOVB       #1,STSTNM            ;SET TEST #1
468 002020    012737    000001    001200            MOV        #1,STESTN            ;ALSO IN MAIL BOX!
469 002026    012737    002042    001106            MOV        #16,SLPADR           ;LOOP FOR ERROR
470 002034    012737    002042    001110            MOV        #16,SLPERR
471 002042    012737    000001    001360    16:        MOV        #1,MEMWR            ;#1 WHEN WE ARE CHANGING MEMORY
472 002050    013737    001346    001370            MOV        LOLIN,LMWR           ;PATTERN -- PICK UP LOW ADDRESS
473 002056    008001                                    CLR        R1                    ;PATTERN = 0 ON EVEN PASS
474 002060    132737    000001    001202            BITB       #1,SPASS            ;IS THIS AN EVEN PASS?
475 002066    001402                                    BEQ        20                    ;YES = CONTINUE
476 002070    012701    177777                                    MOV        #-1,R1               ;NO = ODD PASS, CHANGE TO -1
477
478 002074    010177    177270    20:        MOV        R1,OLMWR            ;WRITE PATTERN
479 002100    005037    001366                                    CLR        BLKCK                ;INDICATE
480                    ;PATTERN OK FOR POWER UP CLOCK,
481                    ;IF WAS # 1 THEN REVERSE PATTERN WOULD
482                    ;HAVE BEEN USED.
483
484 002104    023737    001370    001344    30:        CMP        LMWR,HILIM           ;ARE WE AT HIGH LIMIT?
485 002112    001411                                    BEQ        50
486 002114    103402                                    BLO        40
487 002116    000000                                    HALT
488                    ;SOFT ERROR! ADDRESS IN LOLIM
489                    ;GREATER THAN ADDRESS IN HILIM
490                    ;FATAL CONDITION - DID YOU PATCH THESE
491 002120    000771                                    BR        30                    ;LOCATIONS? I HAD THEM RIGHT,
492 002122    010177    177242    40:        MOV        R1,OLMWR            ;WRITE PATTERN
493 002126    062737    000002    001370            ADD        #2,LMWR
494 002134    000763                                    BR        30                    ;LOOP BACK.
495 002136    005037    001360    50:        CLR        MEMWR               ;INDICATE WE'RE THROUGH WRITING
496                    ;MEMORY
497 002142    013703    001346                                    MOV        LOLIN,R3            ;PUT LOWER LIMIT IN R3
498 002146    010302                                    MOV        R3,R2               ;AND IN R2
499
500                    ;NOW LETS CHECK MEM
501

```

```

502 002150 020237 001344      68:  CMP      R2,HILIM      ;ARE WE AT HIGH ADDR?
503 002154 001415              BEQ      TST2          ;?
504
505 002156 022201              CMP      (2)+,R1      ;PATTERN OK?
506 002160 001773              BEQ      68           ;YES = LOOP,
507
508 002162 010237 001362      MOV      R2,RADDR     ;NO = RECORD ADDR,
509 002166 162737 000002 001362  SUB      02,RADDR     ;=2 = REAL ADDR,
510 002174 017737 177162 001126  MOV      0RADDR,0BDDAT ;RECORD BAD DATA,
511 002202 010137 001124      MOV      R1,0GDDAT   ;RECORD GOOD DATA,
512
513 002206 104001              ERROR   1            ;PROBLEM WITH MEMORY BEFORE
514                          ;EVEN A POWER FAIL,
515
516
517                          ;*****
518                          ;*TEST 2          *TEST THAT THE CLOCK GIVES "B RPLY" ON "DIN" AND "DOUT"
519                          ;*****
519 002210 000004      TST2:  SCOPE
520
521 002212 005737 001372      TST     NOCLK         ;SHOULD WE TEST CLOCK (=0 YES)
522 002216 001402              BEQ     48            ;
523 002220 000137 003020      JMP     NCLK         ;=1 NO TEST OF CLOCK,
524
525 002224      48:
526 002224 013746 000004      MOV     ERRVEC,=(6)  ;SAVE CONTENTS OF LOC. 4,
527 002230 012737 002270 000004  MOV     010,ERRVEC   ;SET FOR BUS ERROR ON "DIN"
528
529 002236 017737 177074 001126  MOV     0LKS,0BDDAT  ;ISSUE "B DIN L" AND CLOCK ADDRESS,
530                          ;WILL TRAP FROM HERE IF CLOCK DOES
531                          ;NOT RESPOND WITH "B RPLY L"
532
533 002244 012737 002274 000004  MOV     020,ERRVEC   ;SET FOR BUS ERROR ON "DOUT"
534 002252 012777 000001 177056  MOV     01,0LKS ;ISSUE "B DOUT" AND CLOCK ADDRESS,
535                          ;WILL TRAP FROM HERE IF CLOCK DOES
536 002260 012737 000001 001356  MOV     01,ADCK      ;NOT RESPOND WITH "B RPLY L"
537 002266 000403              BR     30
538
539 002270 104004      10:  ERROR   4            ;CLOCK DID NOT RESPOND WITH "B RPLY L"
540                          ;WHEN ISSUED ADDRESS AND "B DIN L"
541                          ;CHECK ADDRESS SELECTION JUMPERS
542                          ;AND ADDRESS ENTERED IN LOCATION "0BASE".
543 002272 000401              BR     30
544
545 002274 104004      20:  ERROR   4            ;CLOCK DID NOT RESPOND WITH "B RPLY L"
546                          ;WHEN ISSUED ADDRESS AND "B DOUT L"
547                          ;WE KNOW ADDRESS AND "B DIN" DID
548                          ;WORK SO ADDRESS MUX SHOULD BE GOOD,
549
550 002276 012637 000004      30:  MOV     (6)+,ERRVEC ;RESTORE CONTENTS OF LOC 4,
551
552
553                          ;*****
554                          ;*TEST 3          *TEST THAT BIT6 (INTR, ENBL) CAN BE SET AND CLEARED
555                          ;*****
555 002302 000004      TST3:  SCOPE

```

```

556
557 002304 005737 001356      18:  TST  ADCK      ;DID CLOCK ADDRESS TEST PASS?
558 002310 001002              BNE  28      ;YES - DO THIS TEST.
559 002312 000000              HALT              ;NO - HALT! WE CANNOT PROCEED IF
560 002314 000773              BR   18      ;CLOCK CAN NOT BE ADDRESSED, PLEASE
561                                     ;LOOP ON LAST TEST OR FIX PROBLEM,
562
563 002316 005037 001354      28:  CLR  PFLAG    ;CLEAR POWER FAIL FLAG
564
565                                     ;/-PR-
566 002322 012746 000340      NOV  8340,-(6) ;/PUT NEW STATUS ON STACK
567 002326 012746 002334      NOV  8648,-(6) ;/PUT RETURN ADDR, ON STACK.
568 002332 000002              RTI              ;/DO AN RTI CPU WILL RETURN TO NEXT
569                                     ;/ADDR, WITH NEW STATUS, IF STATUS
570                                     ;/NO THEN INTERRUPTS ALLOWED,
571 002334                      648:
572
573
574 002334 052777 000100 176774  BIS  8BIT6,8LKS ;SET INTR, ENABLE
575 002342 017737 176770 001126  NOV  8LKS,8BDDAT ;READ CSR,
576 002350 032777 000100 176760  BIT  8BIT6,8LKS ;DID IT SET?
577 002356 001010              BNE  38      ;YES CONTINUE
578 002360 005737 001354      TST  PFLAG    ;DID A POWER FAIL OCCUR?
579 002364 001354              BNE  28      ;YES - REDO TEST,
580
581 002366 012737 000100 001124  NOV  8100,8GDDAT ;RECORD EXPECTED,
582
583 002374 104003              ERROR 3      ;INTERRUPT ENABLE BIT (BIT06)
584                                     ;FAILED TO SET,
585                                     ; NOTE
586                                     ;A DISCREPANCY COULD OCCUR IN ERROR
587                                     ;TYPEOUT SINCE S/B = 100 BUT IF
588                                     ;READY FLAG WAS SET BAD DATA WOULD
589                                     ;EQUAL 200 DISPLAYING READY
590 002376 000420              BR   TST4
591
592 002400 005037 001124      38:  CLR  8GDDAT    ;CLEAR S/B,
593 002404 005077 176726      CLR  8LKS      ;CLEAR CLOCK CSR,
594 002410 017737 176722 001126  NOV  8LKS,8BDDAT ;READ CSR
595 002416 032737 000100 001126  BIT  8BIT6,8BDDAT ;DID IT CLEAR,
596 002424 001004              BNE  48      ;NO - REPORT AN ERROR,
597 002426 005737 001354      TST  PFLAG    ;YES - MAKE SURE NO POWER FAIL OCCURRED,
598 002432 001331              BNE  28
599 002434 000401              BR   TST4
600
601 002436 104003      48:  ERROR 3      ;INTERRUPT ENABLE BIT FAILED TO
602                                     ;CLEAR (BIT06),
603                                     ; NOTE:
604                                     ;IN ERROR TYPEOUT BIT7 MAY ALSO
605                                     ;BE SET, THIS IS NOT A BIT7 ERROR,
606                                     ;WE CAN NOT PREDICT WHEN BIT7
607                                     ;WILL SET OR CLEAR AND DO NOT
608                                     ;REFLECT IT IN S/B,
609

```

```

610 ;*****
611 ;*TEST 4 *TEST THAT THE CLOCK READY FLAG WILL SET AND THE CLOCK INTERRUPTS
612 ;*****
613 002440 000004 TST4: SCOPE
614
615 002442 005037 001354 18: CLR PFLAG ;CLEAR POWER FAIL FLAG,
616 002446 28:
617
618 ;/-PR-
619 002446 012746 000340 MOV 8340,-(6) ;/PUT NEW STATUS ON STACK
620 002452 012746 002460 MOV 8648,-(6) ;/PUT RETURN ADDR, ON STACK,
621 002456 000002 RTI ;/DO AN RTI CPU WILL RETURN TO NEXT
622 ;/ADDR, WITH NEW STATUS, IF STATUS
623 ;/0 THEN INTERRUPTS ALLOWED,
624 002460 648:
625
626 002460 012777 000100 176650 MOV 8BIT6,8LKS ;SET INTERRUPT ENABLE ON CLOCK,
627 002466 005002 CLR R2 ;KEEP TIME-OUT COUNTER,
628
629 002470 105777 176642 48: TSTB 8LKS ;DID READY FLAG SET?
630 002474 100415 BMI 58 ;YES = NEXT CHECK,
631 002476 005202 INC R2 ;NO = DID WE ALLOW ENOUGH TIME?
632 002500 001373 BNE 48 ;NO = LOOP,
633 002502 005737 001354 TST PFLAG ;YES = DID A POWER FAIL OCCUR?
634 002506 001355 BNE 18 ;YES = DO TEST OVER, POWER FAIL
635 ;INITIALIZED CLOCK
636 002510 017737 176622 001126 MOV 8LKS,8DDAT ;RECORD CLOCK CSR,
637 002516 012737 000300 001124 MOV 8300,8GDDAT ;RECORD S/B
638
639 002524 104003 ERROR 3 ;CLOCK READY FLAG FAILED TO SET,
640
641 002526 000430 BR TST5 ;;
642
643 002530 012777 002564 176602 58: MOV 868,8KVECT ;SET UP FOR CLOCK INTERRUPT,
644
645 ;/-PR-
646 002536 012746 000000 MOV 80,-(6) ;/PUT NEW STATUS ON STACK
647 002542 012746 002550 MOV 8658,-(6) ;/PUT RETURN ADDR, ON STACK,
648 002546 000002 RTI ;/DO AN RTI CPU WILL RETURN TO NEXT
649 ;/ADDR, WITH NEW STATUS, IF STATUS
650 ;/0 THEN INTERRUPTS ALLOWED,
651 002550 658:
652
653
654 002550 000240 NOP ;INTERRUPTS FROM HERE
655 002552 005737 001354 TST PFLAG ;NO INTERRUPT, DID A POWER FAIL OCCUR?
656 002556 001331 BNE 18 ;YES = DO TEST OVER, POWER FAIL
657 ;INITIALIZED CLOCK,
658
659 002560 104005 ERROR 5 ;CLOCK FAILED TO INTERRUPT
660 ;READY FLAG DID SET, COULD YOU HAVE
661 ;A JUMPER WRONG?
662 002562 000402 BR 78
663

```

```

664 002564 062706 000004      68:  ADD      04,R6      ;UPDATE STACK POINTER
665
666 002570 013777 001340 176542  78:  MOV      KVECT,0KVECT ;RESTORE VECTOR TO LOOK FOR
667 002576 062777 000002 176534      ADD      02,0KVECT    ;ILLEGAL INTERRUPTS.
668 002604 005077 176526      CLR      0LKS         ;CLEAR CSR.
669
670      ;*****
671      ;*TEST 5      *TEST THAT WE CAN CLEAR READY FLAG
672
673      ;*
674      ;*IN THIS TEST WE'RE GONNA TRY TO CLEAR THE READY BIT (BIT07)
675      ;*TO DO THIS WE'RE GONNA CLEAR IT, STALL TIME ALLOWING
676      ;*IT TO SET AGAIN, THEN CLEAR IT TESTING THAT IT ACTUALLY CLEARED.
677      ;*
678      ;*****
679 002610 000004      TSTS:  SCOPE
680
681 002612 005037 001354      18:  CLR      PFLAG      ;CLEAR POWER FAIL FLAG.
682
683 002616 005077 176514      CLR      0LKS         ;CLEAR CLOCK
684 002622 005002      CLR      R2          ;CLEAR TIME OUT FLAG.
685
686 002624 105777 176506      28:  TSTB    0LKS         ;DID READY FLAG SET?
687 002630 100415      BMI      38
688 002632 005302      DEC      R2          ;TIME OUT (ALL ALLOWED TIME EXPIRE?)
689 002634 001373      BNE      28          ;NO - LOOP
690 002636 005737 001354      TST     PFLAG       ;YES - DID A POWER FAIL OCCUR?
691 002642 001363      BNE      18          ;YES - DO TEST OVER.
692
693 002644 017737 176466 001126      MOV     0LKS,0BDDAT  ;RECORD CSR.
694 002652 012737 000200 001124      MOV     0BIT7,0GDDAT ;RECORD S/B.
695
696 002660 104003      ERROR   3           ;CLOCK READY FLAG FAILED TO SET.
697
698 002662 000420      BR      TST6        ;;
699
700 002664 005077 176446      38:  CLR      0LKS         ;THE FLAG JUST SET NOW WE WILL
701 002670 005037 001124      CLR     0GDDAT      ;TRY TO CLEAR IT.
702 002674 017737 176436 001126      MOV     0LKS,0BDDAT ;READ THE CSR IF ZERO - OK.
703 002702 001004      BNE     48           ;IF NOT WE'LL REPORT AN ERROR.
704 002704 005737 001354      TST     PFLAG       ;OF COURSE WE'VE GOT TO CHECK
705      ;THE POWER FAIL FLAG, SINCE A
706      ;POWER FAIL COULD HAVE CLEARED
707      ;THE FLAG!
708      ;IF SO REDO TEST
709 002710 001340      BNE     18           ;IF SO REDO TEST
710 002712 000404      BR      TST6        ;;
711
712 002714 005737 001354      48:  TST     PFLAG       ;CHECK POWER FAIL FLAG-IF POWER FAIL OCCURED
713 002720 001334      BNE     18           ;DURING TEST AN ERROR COULD RESULT.
714      ;IF NO POWER FAIL OCCURED REPORT ERROR.
715
716 002722 104003      ERROR   3           ;COULD NOT CLEAR CLOCK
717      ;READY FLAG (BIT07)

```

```
718 ;;;;;;;;;;;;;;
719 ;*TEST 6 *TEST THAT THE CLOCK DOESN'T INTR, WHEN INTR, ENABLE CLEAR
720 ;;;;;;;;;;;;;;
721 002724 000004 TST6: SCOPE
722 002726 012737 000020 001160 MOV #20,8TIMES ;DO 20 ITERATIONS
723
724 002734 005037 001354 18: CLR PFLAG ;CLEAR POWER FAIL FLAG,
725
726 002740 012777 002776 176372 MOV #38,8KVECT ;SET FOR ILLEGAL INTERRUPT,
727
728
729 ;/-PR-
730 002746 012746 000000 MOV #0,-(6) ;/PUT NEW STATUS ON STACK
731 002752 012746 002760 MOV #648,-(6) ;/PUT RETURN ADDR, ON STACK,
732 002756 000002 RTI ;/DO AN RTI CPU WILL RETURN TO NEXT
733 ;/ADDR, WITH NEW STATUS, IF STATUS
734 ;/0 THEN INTERRUPTS ALLOWED,
735 002760 648:
736
737
738 002760 005002 CLR R2
739 002762 105202 28: INCB R2 ;STALL
740 002764 001376 BNE 28
741 002766 005737 001354 TST PFLAG ;NO INTERRUPT - DID POWER FAIL
742 002772 001360 BNE 18 ;YES - REDO TEST,
743 002774 000403 BR 48
744
745 002776 062706 000004 38: ADD #4,R6 ;UPDATE STACK POINTER,
746
747 003002 104005 ERROR 5 ;CLOCK INTERRUPTED WHEN
748 ;BITS (INTR ENBL) CLEAR,
749 003004 013777 001340 176326 48: MOV KVECT,8KVECT ;RESTORE VECTOR
750 003012 062777 000002 176320 ADD #2,8KVECT
751
752 003020 012737 000006 001102 NCLK: MOV #6,8STNN ;ENTRY POINT IF CLOCK NOT TESTED,
753
754
755 ;;;;;;;;;;;;;;
756 ;*TEST 7 *VERIFY MEMORY LOCATIONS WRITTEN IN TEST1
757 ;*
758 ;*IN THIS MACRO WE ARE GOING TO VERIFY THE CONTENTS OF
759 ;*MEMORY AGAIN, WE ARE GOING TO DEVOTE MOST OF OUR TIME
760 ;*OF THE TOTAL TEST TIME HERE HOPING WE CAN CATCH A POWER
761 ;*FAIL WHILE WE'RE READING MEMORY,
762 ;*
763 ;;;;;;;;;;;;;;
764 003026 000004 TST7: SCOPE
765 003030 012737 002000 001160 MOV #002000,8TIMES ;DO 002000 ITERATIONS
766
767
768 003036 005037 001354 CLR PFLAG
769 003042 005001 CLR R1 ;ZERO PATTERN IS EVEN PASS,
770 003044 132737 000001 001202 BITS #BIT0,8PASS ;EVEN PASS?
771 003052 001402 BEQ 18 ;YES - CONTINUE
```

```

772 003054 012701 177777      MOV      0-1,R1      ;NO = PATTERN ALL ONES,
773 003060 013702 001346      MOV      LOLIM,R2   ;PICK-UP LOW ADDRESS
774                                     ;
775 003064 020237 001344      CMP      R2,HILIM   ;ARE WE AT HIGH ADDRESS?
776 003070 001415      BEQ      TST10      ;;
777                                     ;
778 003072 022201      CMP      (2)+,R1    ;PATTERN OK?
779 003074 001773      BEQ      28         ;YES = LOOP,
780 003076 010237 001362      MOV      R2,RADDR   ;NO = RECORD ADDR,
781 003102 162737 000002 001362  SUB      02,RADDR    ;-2 = READ ADDR,
782 003110 017737 176246 001126  MOV      @RADDR,@BDDAT ;GET CONTENTS
783 003116 010137 001124      MOV      R1,@GDDAT  ;GOOD DATA,
784                                     ;
785 003122 104001      ERROR   1          ;PROBLEM WITH MEMORY, DATA
786                                     ;READ NOT DATA DEPOSITED,
787                                     ;
788                                     ;*****
789                                     ;*TEST 10      *END OF TESTS
790                                     ;*****
791 003124 000004      TST10:  SCOPE
792                                     ;
793                                     ;
794                                     ;SBTTL  END OF PASS ROUTINE
795                                     ;
796                                     ;*****
797                                     ;*INCREMENT THE PASS NUMBER (@PASS)
798                                     ;*IF THERES A MONITOR GO TO IT
799                                     ;*IF THERE ISN'T JUMP TO LOOP
800                                     ;
801 003126      SEOP:
802 003126 005237 001366      INC      BLKCK
803 003132 005037 001102      CLR      $TSTN     ;;ZERO THE TEST NUMBER
804 003136 005037 001160      CLR      $TIMES    ;;ZERO THE NUMBER OF ITERATIONS
805 003142 005237 001202      INC      $PASS     ;;INCREMENT THE PASS NUMBER
806 003146 042737 100000 001202  BIC      $100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
807 003154 005327      DEC      (PC)+     ;;LOOP?
808 003156 000001      SEOPCT: ,WORD    1
809 003160 003065      BGT      $DOAGN    ;;YES
810 003162 012737      MOV      (PC)+,@(PC)+ ;;RESTORE COUNTER
811 003164 000001      SENDCT: ,WORD    1
812 003166 003156      SEOPCT
813                                     ;
814 003170 104401 003176      TYPE    ,658      ;;TYPE ASCII STRING
815 003174 000406      BR      648      ;;GET OVER THE ASCII
816                                     ;
817 003212      ;;658: ,ASCII <200>END PASS 0
818 003212 013746 001202      648:      MOV      $PASS,-($P) ;;SAVE $PASS FOR TYPEOUT
819 003216 104402      TYP0C    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
820 003220 104401 003226      TYPE    ,678      ;;TYPE ASCII STRING
821 003224 000410      BR      668      ;;GET OVER THE ASCII
822                                     ;
823 003246      ;;678: ,ASCII 0 POWER FAILS 0
824 003246 013746 001352      668:      MOV      PCOUNT,-($P) ;;SAVE PCOUNT FOR TYPEOUT
825 003252 104402      TYP0C    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)

```

```

826 003254 005737 001372      TST      NOCLK      ;WAS CLOCK TESTED?
827 003260 001415      BEQ      10      ;YES SKIP TYPEOUT.
828                               ;NO INDICATE CLOCK NOT TESTED.
829 003262 104401 003270      TYPE      ,698      ;TYPE ASCIZ STRING
830 003266 000412      BR      688      ;GET OVER THE ASCIZ
831      ;698:      ,ASCIZ      0      CLOCK NOT TESTED
832 003314      688:
833 003314      101
834 003314 013700 000042      SGET42: NOV      0042,R0      ;GET MONITOR ADDRESS
835 003320 001405      BEQ      SDOAGN      ;BRANCH IF NO MONITOR
836 003322 000005      RESET      ;CLEAR THE WORLD
837 003324 004710      SENDAD: JSR      PC,(R0)      ;GO TO MONITOR
838 003326 000240      NOP      ;SAVE ROOM
839 003330 000240      NOP      ;FOR
840 003332 000240      NOP      ;ACT11
841 003334      SDOAGN:
842 003334 000137      JMP      0(PC)+      ;RETURN
843 003336 002002      BRTHAD: ,WORD      LOOP
844 003340      377      377      000      SNULL: ,BYTE      -1,-1,0      ;NULL CHARACTER STRING
845      003344      ,EVEN
846
847      ;
848      ;THIS ROUTINE HANDLES POWER DOWNS
849      ;
850
851 003344 005737 001364      PDOWN: TST      BAD      ;HAVE WE HAD ENOUGH TIME TO
852 003350 001402      BEQ      PD2      ;DOWN ERROR ON PREVIOUS POWER FAIL?
853 003352 000000      PDBAD: HALT      ;YES - ERROR POWER DOWN ROUTINE
854 003354 000773      BR      PDOWN      ;DID NOT HAVE TIME TO COMPLETE
855                          ;SEQUENCE BEFORE CPU STOPPED.
856
857 003356 005237 001364      PD2:      INC      BAD      ;SET POWER DOWN "NOT ENOUGH TIME" FLAG
858
859 003362 005737 001366      TST      BLKCK      ;IN PROCESS OF CHANGING PATTERN?
860 003366 001410      BEQ      10      ;NO GET OUT.
861
862 003370 027777 175752 175752      CMP      0L0LIN,0L0W2      ;PATTERN =?
863 003376 001404      BEQ      10
864 003400 005037 001366      CLR      BLKCK      ;NO-HAD WRITTEN NEW PATTERN-BUT NOT
865 003404 005237 001360      INC      MEMWR      ;HAD TIME ENOUGH TIME TO CLR BLKCK
866 003410      101
867 003410 005077 175722      CLR      0LKS      ;STOP CLOCK.
868 003414 010046      NOV      R0,-(6)      ;SAVE GENERAL REGISTERS
869 003416 010146      NOV      R1,-(6)
870 003420 010246      NOV      R2,-(6)
871 003422 010346      NOV      R3,-(6)
872 003424 010446      NOV      R4,-(6)
873 003426 010546      NOV      R5,-(6)
874 003430 013746 001124      NOV      0GDDAT,-(6)
875 003434 013746 001126      NOV      0BDDAT,-(6)
876 003440 013746 001362      NOV      RADDR,-(6)
877 003444 010637 003504      NOV      R6,R6
878 003450 012700 000024      NOV      020,,R0      ;NOW WE WILL SIMULATE DOING
879 003454 013737 001124 001124 201      NOV      0GDDAT,0GDDAT      ;160. INSTRUCTIONS, THE POWER DOWN
    
```

```

880 003462 005300                    DEC    R0                    ;SEQUENCE SHOULD ALLOW US AT LEAST
881 003464 001373                    BNE    28                    ;2 MILLISECONDS OF TIME TO DO A POWER
882                                                                            ;FAIL ROUTINE, HOWEVER, ONE MUST
883                                                                            ;BE WATCHFUL THE THE SYSTEM DOESN'T
884                                                                            ;DO MEMORY REFRESH CYCLES ON OTHER
885                                                                            ;MEMORIES DURING THIS TIME, THUS STEALING
886                                                                            ;TIME FROM THE POWER DOWN ROUTINE,
887 003466 012737 003506 000024      MOV    @PUP, PWRVEC        ;POINT TO POWER UP ROUTINE,
888 003474 005037 001364                    CLR    BAD                    ;INDICATE WE HAD ENOUGH TIME
889                                                                            ;
890 003500 000000                    38:    HALT                    ;HALT AT END OF POWER DOWN ROUTINE,
891 003502 000776                    BR     38                    ;ONE MUST DO THIS OR THERE IS A CHANCE
892                                                                            ;THAT THE INSTRUCTION OR MEMORY LOCATION
893                                                                            ;REFERENCED BY THAT INSTRUCTION
894                                                                            ;MAY BE DESTROYED WHEN POWER GOES AWAY.
895                                                                            ;
896                                                                            ;
897 003504 000000                    SR6:   0                    ;SAVE STACK
898                                                                            ;
899                                                                            ;*
900                                                                            ;*THIS ROUTINE HANDLES POWER UPS,
901                                                                            ;*
902 003506 013706 003504                    PUP:   MOV    SR6, R6                ;RESTORE STACK
903 003512 005737 001364                    TST    BAD                    ;POWER DOWN OK?
904 003516 001318                    BNE    PDBAD                ;NO - GO THERE,
905 003520 012737 004026 000024      MOV    @PUPBD, PWRVEC      ;SET UP FOR BAD POWER DOWN
906                                                                            ;BEFORE WE CAN POWER UP,
907 003526 005237 001354                    INC    PFLAG                ;INDICATE POWER FAIL,
908                                                                            ;
909                                                                            ;
910 003532 023727 001354 000005      PU2:   CMP    PFLAG, #5            ;HAVE WE DONE TOO MANY POWER
911 003540 001005                    BNE    PD3                    ;UPS WITHOUT DOING COMPLETE ONES?
912 003542 012737 003356 000024      MOV    @PD2, PWRVEC        ;STAY HERE FROM NOW ON ON POWER SEQUENCES
913                                                                            ;FORCE OPERATOR TO RESTART PROGRAM,
914 003550 000000                    HALT                        ;TOO MANY POWER FAILS OCCURRED BEFORE
915 003552 000767                    BR     PU2                    ;ONE COULD BE COMPLETED 5 * MAX,
916                                                                            ;
917 003554 012737 003344 000024      PD3:   MOV    @PDOWN, PWRVEC      ;SET UP FOR POWER DOWN,
918                                                                            ;
919                                                                            ;*BTTL MEMORY TEST ON POWER UP,
920                                                                            ;
921 003562 005037 001124                    CLR    @GDDAT                ;PATTERN ZERO ON EVEN PASS
922 003566 132737 000001 001202      BITS   @1, @PASS            ;EVEN PASS
923 003574 001403                    BEQ    18                    ;YES - CONTINUE
924 003576 012737 177777 001124      MOV    @-1, @GDDAT        ;NO - LOOK FOR ALL ONES,
925 003604 005737 001366                    18:    TST    BLKCK                ;WERE WE STARTING TO WRITE MEM?
926 003610 001407                    BEQ    28                    ;NO - CONTINUE
927 003612 005137 001124                    COM    @GDDAT                ;YES - USE OPPOSITE PATTERN,
928 003616 013706 001344                    MOV    N1LIN, R5            ;PICK UP HIGH ADDR,
929 003622 013704 001346                    MOV    L0LIN, R4            ;PICK UP LO ADDR
930 003626 000411                    BR     CK1                    ;
931 003630 013704 001346                    28:    MOV    L0LIN, R4            ;PICK UP LOW ADDR,
932 003634 013706 001344                    MOV    N1LIN, R5            ;PICK UP HIGH ADDR,
933 003640 005737 001360                    TST    MEMWR                ;WERE WE CHANGING PATTERN?
  
```

934	003644	001402			BEG	CK1		;NO = OK
935	003646	013705	001370		MOV	LMHR,R5		;YES = R3 WILL HAVE LAST ADDR.
936	003652	020405		CK1:	CMP	R4,R5		;DONE CHECK?
937	003654	001415			BEG	CKEND		;YES END
938								
939	003656	022437	001124		CMP	(4)+,8GDDAT		;MEMORY OK
940	003662	001773			BEG	CK1		;YES = CHECK NEXT ADDR.
941	003664	016437	177776	001126	MOV	-2(4),8BDDAT		;NO = GET FAILING PATTERN
942	003672	010437	001362		MOV	R4,RADDR		;GET ADDR
943	003676	162737	000002	001362	SUB	#2,RADDR		;READ = -2
944								
945	003704	104002			ERROR	2		;READ PATTERN ERROR MEMORY
946								;CONTAINED OTHER THAN WRITTEN.
947	003706	000761			BR	CK1		
948								
949	003710	005237	001352		CKEND:	INC	PCOUNT	;UPDATE POWER FAIL COUNT.
950								
951						.SBTTL	CLOCK CHECK AFTER POWER UP	
952								
953	003714	005737	001356		TST	ADCK		;HAS CLOCK BEEN TESTED?
954	003720	001412			BEG	PUEND		;NO = LETS NOT TEST IT HERE!
955								
956	003722	017737	175410	001126	MOV	8LKS,8BDDAT		;READ CSR
957	003730	005037	001124		CLR	8GDDAT		;EXPECT ZERO ALTHOUGH BIT7
958								;MAY BE SET, WE WON'T LOOK
959								;AT IT.
960	003734	032737	000100	001126	BIT	8BIT6,8BDDAT		;IS BIT6 SET?
961	003742	001401			BEG	PUEND		;NO = GOOD.
962	003744	104003			ERROR	3		;BIT6 OF CLOCK CSR SET AFTER
963								;POWER UP - INIT FAILED TO
964								;CLEAR IT.
965	003746				PUEND:			
966	003746	012637	001362		MOV	(6)+,RADDR		;RESTORE EVERYTHING THAT WAS SAVED BEFORE POWER FAIL.
967	003752	012637	001126		MOV	(6)+,8BDDAT		
968	003756	012637	001124		MOV	(6)+,8GDDAT		
969	003762	012605			MOV	(6)+,R5		
970	003764	012604			MOV	(6)+,R4		
971	003766	012603			MOV	(6)+,R3		
972	003770	012602			MOV	(6)+,R2		
973	003772	012601			MOV	(6)+,R1		
974	003774	012600			MOV	(6)+,R0		
975	003776	001401			BEG	28		;IF R0=0 THEN NO MESSAGE BEING TYPED.
976	004000	105300			DECB	R0		;IF MESSAGE BEING TYPED, BACK UP FOR LAST CHARACTER.
977	004002							
978	004002	005700			TST	R0		;IF NO MESSAGE BEING TYPED OUT WE
979								;WILL TYPE OUT "PF-OK".
980	004004	001007			BNE	18		
981	004006	104401	004014		TYPE	,658		;TYPE ASCII STRING
982	004012	000404			BR	648		;GET OVER THE ASCII
983					;1658:	,ASCII	<200>PF-OK:	
984	004024				648:			
985	004024	000002			18:	RTI		
986								
987	004026	000000			PUPBD:	HALT		;POWER DOWN OCCURRED BEFORE

```
988 004030 000776 BR PUPBD ;WE COULD SUCCESSFULLY START
989 ;A POWER UP
990
991 ,SBTTL ERROR HANDLER ROUTINE
992
993 ;;*****
994 ;;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
995 ;;SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
996 ;;AND GO TO SERRTYP ON ERROR
997 ;;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
998 ;;SW15=1 HALT ON ERROR
999 ;;SW13=1 INHIBIT ERROR TYPEOUTS
1000 ;;SW10=1 BELL ON ERROR
1001 ;;SW09=1 LOOP ON ERROR
1002 ;;CALL
1003 ;; ERROR N ;;ERROR=ENT AND N=ERROR ITEM NUMBER
1004
1005 004032 SERROR:
1006 004032 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
1007 004034 108237 001103 70: INCB SERFLG ;;SET THE ERROR FLAG
1008 004040 001778 BEQ 70 ;;DON'T LET THE FLAG GO TO ZERO
1009 004042 013777 001102 175072 NOV STSTNN,ODISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
1010 004050 032777 002000 175062 BIT SBIT10,OSWR ;;BELL ON ERROR?
1011 004056 001402 BEQ 10 ;;NO - SKIP
1012 004060 104401 001164 TYPE ,SBELL ;;RING BELL
1013 004064 008237 001112 10: INC SERTTL ;;COUNT THE NUMBER OF ERRORS
1014 004070 011637 001116 NOV (SP),SERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
1015 004074 162737 000002 001116 SUB 02,SERRPC
1016 004102 117737 175010 001114 NOV SERRPC,SITENS ;;STRIP AND SAVE THE ERROR ITEM CODE
1017 004110 032777 020000 175022 BIT SBIT13,OSWR ;;SKIP TYPEOUT IF SET
1018 004116 001004 BNE 300 ;;SKIP TYPEOUTS
1019 004120 004737 004220 JSR PC,SERRTYP ;;GO TO USER ERROR ROUTINE
1020 004124 104401 001171 TYPE ,SCLRF
1021 004130 200:
1022 004130 122737 000001 001214 CNPB SAPTENV,SENV ;;RUNNING IN APT MODE
1023 004136 001007 BNE 20 ;;NO,SKIP APT ERROR REPORT
1024 004140 113737 001114 004152 NOV SITEMB,216 ;;SET ITEM NUMBER AS ERROR NUMBER
1025 004146 004737 006364 JSR PC,SATY4 ;;REPORT FATAL ERROR TO APT
1026 004152 000 210: ,BYTE 0
1027 004153 000 ,BYTE 0
1028 004154 000777 220: BR 220 ;;APT ERROR LOOP
1029 004156 005777 174756 20: TST OSWR ;;HALT ON ERROR
1030 004162 100002 BPL 30 ;;SKIP IF CONTINUE
1031 004164 000000 HALT ;;HALT ON ERROR!
1032 004166 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
1033 004170 032777 001000 174742 30: BIT SBIT09,OSWR ;;LOOP ON ERROR SWITCH SET?
1034 004176 001402 BEQ 40 ;;BR IF NO
1035 004200 013716 001110 NOV SLPERR,(SP) ;;FUDGE RETURN FOR LOOPING
1036 004204 005737 001162 40: TST SESCPE ;;CHECK FOR AN ESCAPE ADDRESS
1037 004210 001402 BEQ 50 ;;BR IF NONE
1038 004212 013716 001162 NOV SESCPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
1039 004216 50:
1040 004216 000002 RTI ;;RETURN
1041 ,SBTTL ERROR MESSAGE TYPEOUT ROUTINE
```

1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095

004220
004220 104401 001171
004224 010046
004226 005000
004230 153700 001114
004234 001004
004236 013746 001116
004242 104402
004244 000426
004246 005300
004250 006300
004252 006300
004254 006300
004256 062700 001256
004262 012037 004272
004266 001404
104401
000000
004274 104401 001171
004300 012037 004310
004304 001404
004306 104401
004310 000000
004312 104401 001171
004316 011000
004320 001004
004322 012600
004324 104401 001171
004330 000267
004332
013046
004334 104402
004336 005710
004340 001770
004342 104401 004350
004346 000771
004350 020040 000
004354

;;*****
;THIS ROUTINE USES THE "ITEM CONTROL BYTE" (SITEMB) TO DETERMINE WHICH
;ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR TABLE" (SERRTB),
;AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR,

SERRTYP:

TYPE ,SCLRF ;; "CARRIAGE RETURN" & "LINE FEED"
NOV R0,-(SP) ;; SAVE R0
CLR R0 ;; PICKUP THE ITEM INDEX
BISB @SITEMB,R0
BNE 10 ;; IF ITEM NUMBER IS ZERO, JUST
 ;; TYPE THE PC OF THE ERROR
NOV SERRPC,-(SP) ;; SAVE SERRPC FOR TYPEOUT
 ;; ERROR ADDRESS
TYPCC ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
BR 60 ;; GET OUT
10: DEC R0 ;; ADJUST THE INDEX SO THAT IT WILL
 ;; WORK FOR THE ERROR TABLE
ASL R0
ASL R0
ASL R0
ADD @SERRTB,R0 ;; FORM TABLE POINTER
NOV (R0)+,26 ;; PICKUP "ERROR MESSAGE" POINTER
BEQ 30 ;; SKIP TYPEOUT IF NO POINTER
TYPE ;; TYPE THE "ERROR MESSAGE"
20: ,WORD 0 ;; "ERROR MESSAGE" POINTER GOES HERE
TYPE ,SCLRF ;; "CARRIAGE RETURN" & "LINE FEED"
30: NOV (R0)+,46 ;; PICKUP "DATA HEADER" POINTER
BEQ 50 ;; SKIP TYPEOUT IF 0
TYPE ;; TYPE THE "DATA HEADER"
40: ,WORD 0 ;; "DATA HEADER" POINTER GOES HERE
TYPE ,SCLRF ;; "CARRIAGE RETURN" & "LINE FEED"
50: NOV (R0),R0 ;; PICKUP "DATA TABLE" POINTER
BNE 70 ;; GO TYPE THE DATA
60: NOV (SP)+,R0 ;; RESTORE R0
TYPE ,SCLRF ;; "CARRIAGE RETURN" & "LINE FEED"
RTS PC ;; RETURN
70: NOV 0(R0)+,-(SP) ;; SAVE 0(R0)+ FOR TYPEOUT
TYPCC ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
TST (R0) ;; IS THERE ANOTHER NUMBER?
BEQ 60 ;; BR IF NO
TYPE ,00 ;; TYPE TWO(2) SPACES
BR 70 ;; LOOP
80: ,ASCII / / ;; TWO(2) SPACES
 ,EVEN

,SBITL SCOPE HANDLER ROUTINE

;;*****
;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS, IT WILL INCREMENT
;AND LOAD THE TEST NUMBER(SSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
;AND LOAD THE ERROR FLAG (SERFLG) INTO DISPLAY<15:00>
;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;@SW1401 LOOP ON TEST

```

1096                                     ;%SW11=1      INHIBIT ITERATIONS
1097                                     ;%SW09=1      LOOP ON ERROR
1098                                     ;%SW08=1      LOOP ON TEST IN SWR<7:0>
1099                                     ;%CALL
1100                                     ;%          SCOPE          ;;SCOPE=IOT
1101
1102 004354                                %SCOPE:
1103 004354 104407                        CKSWR          ;;TEST FOR CHANGE IN SOFT-SWM
1104 004356 104407                        CKSWR
1105 004360 032777 040000 174552 18:   BIT          %BIT14,%SWR      ;;LOOP ON PRESENT TEST?
1106 004366 001114                        BNE          %OVER          ;;YES IF SW14=1
1107                                     ;%START OF CODE FOR THE XOR TESTER%
1108 004370 000416                        %XTSTR: BR      %          ;;IF RUNNING ON THE "XOR" TESTER CHANGE
1109                                     ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
1110 004372 013746 000004                        NOV          %ERRVEC,-(%SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
1111 004376 012737 004416 000004                        NOV          %5,%ERRVEC      ;;SET FOR TIMEOUT
1112 004404 005737 177060                        TST          %177060        ;;TIME OUT ON XOR?
1113 004410 012637 000004                        NOV          (%SP)+,%ERRVEC ;;RESTORE THE ERROR VECTOR
1114 004414 000463                        BR          %SVLAD          ;;GO TO THE NEXT TEST
1115 004416 022626                        %:   CMP          (%SP)+,(%SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
1116 004420 012637 000004                        NOV          (%SP)+,%ERRVEC ;;RESTORE THE ERROR VECTOR
1117 004424 000423                        BR          %          ;;LOOP ON THE PRESENT TEST
1118 004426                                     %:;%END OF CODE FOR THE XOR TESTER%
1119 004426 032777 000400 174504                        BIT          %BIT08,%SWR      ;;LOOP ON SPEC. TEST?
1120 004434 001404                        BEQ          %          ;;BR IF NO
1121 004436 127737 174476 001102                        CNPD         %SWR,%STSTNM     ;;ON THE RIGHT TEST? SWR<7:0>
1122 004444 001465                        BEQ          %          ;;BR IF YES
1123 004446 105737 001103                        %:   TSTB         %ERFLG      ;;HAS AN ERROR OCCURRED?
1124 004452 001421                        BEQ          %          ;;BR IF NO
1125 004454 123737 001115 001103                        CNPD         %ERNAX,%ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
1126 004462 101015                        BHI          %          ;;BR IF NO
1127 004464 032777 001000 174446                        BIT          %BIT09,%SWR      ;;LOOP ON ERROR?
1128 004472 001404                        BEQ          %          ;;BR IF NO
1129 004474 013737 001110 001106 %:   NOV          %LPERR,%LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
1130 004502 000446                        BR          %OVER
1131 004504 108037 001103                        %:   CLRB         %ERFLG      ;;ZERO THE ERROR FLAG
1132 004510 005037 001160                        CLR          %STINES        ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
1133 004514 000415                        BR          %          ;;ESCAPE TO THE NEXT TEST
1134 004516 032777 004000 174414 %:   BIT          %BIT11,%SWR     ;;INHIBIT ITERATIONS?
1135 004524 001011                        BNE          %          ;;BR IF YES
1136 004526 005737 001202                        TST         %PASS          ;;IF FIRST PASS OF PROGRAM
1137 004532 001406                        BEQ          %          ;; INHIBIT ITERATIONS
1138 004534 005237 001104                        INC          %ICNT          ;;INCREMENT ITERATION COUNT
1139 004540 023737 001160 001104                        CNPD         %STINES,%ICNT   ;;CHECK THE NUMBER OF ITERATIONS MADE
1140 004546 002024                        BGE          %OVER         ;;BR IF MORE ITERATION REQUIRED
1141 004550 012737 000001 001104 %:   NOV          %1,%ICNT      ;;REINITIALIZE THE ITERATION COUNTER
1142 004556 013737 004634 001160                        NOV          %NXCNT,%STINES ;;SET NUMBER OF ITERATIONS TO DO
1143 004564 105237 001102                        %SVLAD: INCB         %STSTNM ;;COUNT TEST NUMBERS
1144 004570 113737 001162 001200                        NOV         %STSTNM,%STSTNM ;;SET TEST NUMBER IN APT MAILBOX
1145 004576 011637 001106                        NOV          (%SP),%LPADR     ;;SAVE SCOPE LOOP ADDRESS
1146 004602 011637 001110                        NOV          (%SP),%LPERR    ;;SAVE ERROR LOOP ADDRESS
1147 004606 005037 001162                        CLR          %ESCAPE        ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
1148 004612 112737 000001 001115                        NOV         %1,%ERNAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
1149 004620 013777 001102 174314 %OVER: NOV          %STSTNM,%DISPLAY ;;DISPLAY TEST NUMBER

```

```

1150 004626 013716 001106          MOV      8LPADR,(SP)      ;;FUDGE RETURN ADDRESS
1151 004632 000002          RTI                    ;;FIXES PS
1152 004634 000020          SMAXCNT: 20           ;;MAX, NUMBER OF ITERATIONS
1153          ,SBTTL  TTY INPUT ROUTINE
1154
1155          ;;*****
1156          ,ENABL  LSB
1157
1158          ;;*****
1159          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE,
1160          ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
1161          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
1162          ;*WHEN OPERATING IN TTY FLAG MODE,
1163 004636 022737 000176 001140  SCKSWR: CMP      8SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
1164 004644 001074          BNE      158           ;;BRANCH IF NO
1165 004646 105777 174272          TSTB    88TK8         ;;CHAR THERE?
1166 004652 100071          BPL      158           ;;IF NO, DON'T WAIT AROUND
1167 004654 117746 174266          MOVB    88TK8,-(SP)    ;;SAVE THE CHAR
1168 004660 042716 177600          BIC     8"C177,(SP)   ;;STRIP-OFF THE ASCII
1169 004664 022726 000007          CMP     87,(SP)+      ;;IS IT A CONTROL G?
1170 004670 001062          BNE     158           ;;NO, RETURN TO USER
1171 004672 123727 001134 000001  CMPB    8AUTOB,81     ;;ARE WE RUNNING IN AUTO-MODE?
1172 004700 001456          BEQ     158           ;;BRANCH IF YES
1173
1174 004702 104401 005363          TYPE    ,8CNTLG      ;;ECHO THE CONTROL-G (^G)
1175 004706 104401 005370  SGTSWR: TYPE    ,8MSWR      ;;TYPE CURRENT CONTENTS
1176 004712 013746 000176          MOV     SWREG,-(SP)   ;;SAVE SWREG FOR TYPEOUT
1177 004716 104402          TYPOC   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1178 004720 104401 005401          TYPE    ,8MNEW      ;;PROMPT FOR NEW SWR
1179 004724 025046 1981          CLR     -(SP)        ;;CLEAR COUNTER
1180 004726 005046          CLR     -(SP)        ;;THE NEW SWR
1181 004730 105777 174210  761          TSTB    88TK8         ;;CHAR THERE?
1182 004734 100375          BPL     78           ;;IF NOT TRY AGAIN
1183
1184 004736 117746 174204          MOVB    88TK8,-(SP)   ;;PICK UP CHAR
1185 004742 042716 177600          BIC     8"C177,(SP)   ;;MAKE IT 7-BIT ASCII
1186
1187
1188
1189 004746 021627 000025  981          CMP     (SP),825     ;;IS IT A CONTROL-U?
1190 004752 001005          BNE     108           ;;BRANCK IF NOT
1191 004754 104401 005356          TYPE    ,8CNTLU     ;;YES, ECHO CONTROL-U (^U)
1192 004760 062706 000006  2081         ADD     86,SP        ;;IGNORE PREVIOUS INPUT
1193 004764 000757          BR      198         ;;LET'S TRY IT AGAIN
1194
1195
1196 004766 021627 000015  1081        CMP     (SP),815     ;;IS IT A <CR>?
1197 004772 001022          BNE     168           ;;BRANCH IF NO
1198 004774 005766 000004          TST     4(SP)        ;;YES, IS IT THE FIRST CHAR?
1199 005000 001403          BEQ     118           ;;BRANCH IF YES
1200 005002 016677 000002 174130      MOV     2(SP),8SWR    ;;SAVE NEW SWR
1201 005010 062706 000006  1181        ADD     86,SP        ;;CLEAR UP STACK
1202 005014 104401 001171  1481        TYPE    ,8CRLF      ;;ECHO <CR> AND <LF>
1203 005020 123727 001135 000001  CMPB    8INTAG,81    ;;RE-ENABLE TTY KBD INTERRUPTS?

```

```

1204 005026 001003      BNE      158      ;;BRANCH IF NOT
1205 005030 012777 000100 174106      MOV      0100,08TK8 ;;RE-ENABLE TTY KBD INTERRUPTS
1206 005036 000002      RTI      ;;RETURN
1207 005040 004737 006276      JSR      PC,8TYPEC ;;ECHO CHAR
1208 005044 021627 000060      CMP      (SP),060   ;;CHAR < 0?
1209 005050 002420      BLT      188      ;;BRANCH IF YES
1210 005052 021627 000067      CMP      (SP),067   ;;CHAR > 7?
1211 005056 003018      BGT      188      ;;BRANCH IF YES
1212 005060 042726 000060      BIC      060,(SP)+  ;;STRIP-OFF ASCII
1213 005064 008766 000002      TST      2(SP)      ;;IS THIS THE FIRST CHAR
1214 005070 001403      BEQ      178      ;;BRANCH IF YES
1215 005072 006316      ASL      (SP)      ;;NO, SHIFT PRESENT
1216 005074 006316      ASL      (SP)      ;; CHAR OVER TO MAKE
1217 005076 006316      ASL      (SP)      ;; ROOM FOR NEW ONE.
1218 005100 005266 000002      INC      2(SP)      ;;KEEP COUNT OF CHAR
1219 005104 056616 177776      BIS      -2(SP),(SP) ;;SET IN NEW CHAR
1220 005110 000707      BR       78        ;;GET THE NEXT ONE
1221 005112 104401 001170      TYPE    ,8QUES    ;;TYPE ?<CR><LF>
1222 005116 000720      BR       208      ;;SIMULATE CONTROL-U
1223      ,DSABL L8B
1224
1225
1226      ;;=====
1227      ;;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
1228      ;;CALL:
1229      ;;      RDCHR      ;;INPUT A SINGLE CHARACTER FROM THE TTY
1230      ;;      RETURN HERE ;;CHARACTER IS ON THE STACK
1231      ;;      ;;WITH PARITY BIT STRIPPED OFF
1232      ;;
1233
1234 005120 011646      @RDCHR: MOV      (SP),-(SP) ;;PUSH DOWN THE PC
1235 005122 016666 000004 000002      MOV      4(SP),2(SP) ;;SAVE THE PC
1236 005130 105777 174010      TSTB    08TK8     ;;WAIT FOR
1237 005134 100378      BPL      18      ;;A CHARACTER
1238 005136 117766 174004 000004      MOVB    08TKB,4(SP) ;;READ THE TTY
1239 005144 042766 177600 000004      BIC      0"C<177>,4(SP) ;;GET RID OF JUNK IF ANY
1240 005152 026627 000004 000023      CMP      4(SP),023 ;;IS IT A CONTROL-0?
1241 005160 001013      BPL      28      ;;BRANCH IF NO
1242 005162 105777 173756      TSTB    08TK8     ;;WAIT FOR A CHARACTER
1243 005166 100378      BPL      28      ;;LOOP UNTIL ITS THERE
1244 005170 117746 173752      MOVB    08TKB,-(SP) ;;GET CHARACTER
1245 005174 042716 177600      BIC      0"C177,(SP) ;;MAKE IT 7-BIT ASCII
1246 005200 022627 000021      CMP      (SP)+,021 ;;IS IT A CONTROL-0?
1247 005204 001366      BNE      28      ;;IF NOT DISCARD IT
1248 005206 000750      BR       18      ;;YES, RESUME
1249 005210 026627 000004 000140      CMP      4(SP),0140 ;;IS IT UPPER CASE?
1250 005216 002407      BLT      46      ;;BRANCH IF YES
1251 005220 026627 000004 000175      CMP      4(SP),0175 ;;IS IT A SPECIAL CHAR?
1252 005226 003003      BGT      46      ;;BRANCH IF YES
1253 005230 042766 000040 000004      BIC      040,4(SP) ;;MAKE IT UPPER CASE
1254 005236 000002      RTI      ;;GO BACK TO USER
1255      ;;=====
1256      ;;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
1257      ;;CALL:
    
```

```

1258 ;* RDLIN ; INPUT A STRING FROM THE TTY
1259 ;* RETURN HERE ; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
1260 ;* ; TERMINATOR WILL BE A BYTE OF ALL 0'S
1261
1262 005240 010346 8RDLIN: MOV R3,-(SP) ; SAVE R3
1263 005242 012703 005346 18: MOV 8TTYIN,R3 ; GET ADDRESS
1264 005246 022703 005356 28: CMP 8TTYIN+8.,R3 ; BUFFER FULL?
1265 005252 101405 ; BR IF YES
1266 005254 104410 ; GO READ ONE CHARACTER FROM THE TTY
1267 005256 112613 ; GET CHARACTER
1268 005260 122713 000177 108: CMPB 8177,(R3) ; IS IT A RUBOUT
1269 005264 001003 ; BNE 38 ; SKIP IF NOT
1270 005266 104401 001170 48: TYPE ,8QUES ; TYPE A '?'
1271 005272 000763 ; BR 18 ; CLEAR THE BUFFER AND LOOP
1272 005274 111337 005344 38: MOVB (R3),98 ; ECHO THE CHARACTER
1273 005300 104401 005344 ; TYPE ,98
1274 005304 122723 000015 ; CMPB 815,(R3)+ ; CHECK FOR RETURN
1275 005310 001356 ; BNE 28 ; LOOP IF NOT RETURN
1276 005312 105063 177777 ; CLRB -1(R3) ; CLEAR RETURN (THE 15)
1277 005316 104401 001172 ; TYPE ,8LF ; TYPE A LINE FEED
1278 005322 012603 ; MOV (SP)+,R3 ; RESTORE R3
1279 005324 011646 ; MOV (SP),-(SP) ; ADJUST THE STACK AND PUT ADDRESS OF THE
1280 005326 016666 000004 000002 ; MOV 4(SP),2(SP) ; FIRST ASCII CHARACTER ON IT
1281 005334 012766 005346 000004 ; MOV 8TTYIN,4(SP)
1282 005342 000002 ; RTI ; RETURN
1283 005344 000 98: ,BYTE 0 ; STORAGE FOR ASCII CHAR, TO TYPE
1284 005345 000 ; ,BYTE 0 ; TERMINATOR
1285 005346 000010 ; ,BLKB 8 ; RESERVE 8 BYTES FOR TTY INPUT
1286 005356 052536 005015 000 8TTYIN: ,ASCIZ /"U/<15><12> ; CONTROL "U"
1287 005363 136 006807 000012 8CNTLU: ,ASCIZ /"G/<15><12> ; CONTROL "G"
1288 005370 005015 053523 020122 8RNR: ,ASCIZ <15><12>/RNR = /
1289 005376 020075 000 ;
1290 005401 040 047040 053505 8RNEW: ,ASCIZ / NEW = /
1291 005406 036440 000040 ;
1292 ; ,SBTTL BINARY TO OCTAL (ASCII) AND TYPE
1293
1294 ; ;
1295 ; *****
1296 ; THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
1297 ; OCTAL (ASCII) NUMBER AND TYPE IT.
1298 ; STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
1299 ; CALL:
1300 ; MOV NUM,-(SP) ; NUMBER TO BE TYPED
1301 ; TYPOS ; CALL FOR TYPEOUT
1302 ; ,BYTE N ; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
1303 ; ,BYTE N ; N=1 OR 0
1304 ; ;
1305 ; ;
1306 ; ;
1307 ; ;
1308 ; ;
1309 ; ;
1310 ; ;
1311 ; ;
1312 ; ;
1313 ; ;
1314 ; ;
1315 ; ;
1316 ; ;
1317 ; ;
1318 ; ;
1319 ; ;
1320 ; ;
1321 ; ;
1322 ; ;
1323 ; ;
1324 ; ;
1325 ; ;
1326 ; ;
1327 ; ;
1328 ; ;
1329 ; ;
1330 ; ;
1331 ; ;
1332 ; ;
1333 ; ;
1334 ; ;
1335 ; ;
1336 ; ;
1337 ; ;
1338 ; ;
1339 ; ;
1340 ; ;
1341 ; ;
1342 ; ;
1343 ; ;
1344 ; ;
1345 ; ;
1346 ; ;
1347 ; ;
1348 ; ;
1349 ; ;
1350 ; ;
1351 ; ;
1352 ; ;
1353 ; ;
1354 ; ;
1355 ; ;
1356 ; ;
1357 ; ;
1358 ; ;
1359 ; ;
1360 ; ;
1361 ; ;
1362 ; ;
1363 ; ;
1364 ; ;
1365 ; ;
1366 ; ;
1367 ; ;
1368 ; ;
1369 ; ;
1370 ; ;
1371 ; ;
1372 ; ;
1373 ; ;
1374 ; ;
1375 ; ;
1376 ; ;
1377 ; ;
1378 ; ;
1379 ; ;
1380 ; ;
1381 ; ;
1382 ; ;
1383 ; ;
1384 ; ;
1385 ; ;
1386 ; ;
1387 ; ;
1388 ; ;
1389 ; ;
1390 ; ;
1391 ; ;
1392 ; ;
1393 ; ;
1394 ; ;
1395 ; ;
1396 ; ;
1397 ; ;
1398 ; ;
1399 ; ;
1400 ; ;
1401 ; ;
1402 ; ;
1403 ; ;
1404 ; ;
1405 ; ;
1406 ; ;
1407 ; ;
1408 ; ;
1409 ; ;
1410 ; ;
1411 ; ;
1412 ; ;
1413 ; ;
1414 ; ;
1415 ; ;
1416 ; ;
1417 ; ;
1418 ; ;
1419 ; ;
1420 ; ;
1421 ; ;
1422 ; ;
1423 ; ;
1424 ; ;
1425 ; ;
1426 ; ;
1427 ; ;
1428 ; ;
1429 ; ;
1430 ; ;
1431 ; ;
1432 ; ;
1433 ; ;
1434 ; ;
1435 ; ;
1436 ; ;
1437 ; ;
1438 ; ;
1439 ; ;
1440 ; ;
1441 ; ;
1442 ; ;
1443 ; ;
1444 ; ;
1445 ; ;
1446 ; ;
1447 ; ;
1448 ; ;
1449 ; ;
1450 ; ;
1451 ; ;
1452 ; ;
1453 ; ;
1454 ; ;
1455 ; ;
1456 ; ;
1457 ; ;
1458 ; ;
1459 ; ;
1460 ; ;
1461 ; ;
1462 ; ;
1463 ; ;
1464 ; ;
1465 ; ;
1466 ; ;
1467 ; ;
1468 ; ;
1469 ; ;
1470 ; ;
1471 ; ;
1472 ; ;
1473 ; ;
1474 ; ;
1475 ; ;
1476 ; ;
1477 ; ;
1478 ; ;
1479 ; ;
1480 ; ;
1481 ; ;
1482 ; ;
1483 ; ;
1484 ; ;
1485 ; ;
1486 ; ;
1487 ; ;
1488 ; ;
1489 ; ;
1490 ; ;
1491 ; ;
1492 ; ;
1493 ; ;
1494 ; ;
1495 ; ;
1496 ; ;
1497 ; ;
1498 ; ;
1499 ; ;
1500 ; ;
1501 ; ;
1502 ; ;
1503 ; ;
1504 ; ;
1505 ; ;
1506 ; ;
1507 ; ;
1508 ; ;
1509 ; ;
1510 ; ;
1511 ; ;
1512 ; ;
1513 ; ;
1514 ; ;
1515 ; ;
1516 ; ;
1517 ; ;
1518 ; ;
1519 ; ;
1520 ; ;
1521 ; ;
1522 ; ;
1523 ; ;
1524 ; ;
1525 ; ;
1526 ; ;
1527 ; ;
1528 ; ;
1529 ; ;
1530 ; ;
1531 ; ;
1532 ; ;
1533 ; ;
1534 ; ;
1535 ; ;
1536 ; ;
1537 ; ;
1538 ; ;
1539 ; ;
1540 ; ;
1541 ; ;
1542 ; ;
1543 ; ;
1544 ; ;
1545 ; ;
1546 ; ;
1547 ; ;
1548 ; ;
1549 ; ;
1550 ; ;
1551 ; ;
1552 ; ;
1553 ; ;
1554 ; ;
1555 ; ;
1556 ; ;
1557 ; ;
1558 ; ;
1559 ; ;
1560 ; ;
1561 ; ;
1562 ; ;
1563 ; ;
1564 ; ;
1565 ; ;
1566 ; ;
1567 ; ;
1568 ; ;
1569 ; ;
1570 ; ;
1571 ; ;
1572 ; ;
1573 ; ;
1574 ; ;
1575 ; ;
1576 ; ;
1577 ; ;
1578 ; ;
1579 ; ;
1580 ; ;
1581 ; ;
1582 ; ;
1583 ; ;
1584 ; ;
1585 ; ;
1586 ; ;
1587 ; ;
1588 ; ;
1589 ; ;
1590 ; ;
1591 ; ;
1592 ; ;
1593 ; ;
1594 ; ;
1595 ; ;
1596 ; ;
1597 ; ;
1598 ; ;
1599 ; ;
1600 ; ;
1601 ; ;
1602 ; ;
1603 ; ;
1604 ; ;
1605 ; ;
1606 ; ;
1607 ; ;
1608 ; ;
1609 ; ;
1610 ; ;
1611 ; ;
1612 ; ;
1613 ; ;
1614 ; ;
1615 ; ;
1616 ; ;
1617 ; ;
1618 ; ;
1619 ; ;
1620 ; ;
1621 ; ;
1622 ; ;
1623 ; ;
1624 ; ;
1625 ; ;
1626 ; ;
1627 ; ;
1628 ; ;
1629 ; ;
1630 ; ;
1631 ; ;
1632 ; ;
1633 ; ;
1634 ; ;
1635 ; ;
1636 ; ;
1637 ; ;
1638 ; ;
1639 ; ;
1640 ; ;
1641 ; ;
1642 ; ;
1643 ; ;
1644 ; ;
1645 ; ;
1646 ; ;
1647 ; ;
1648 ; ;
1649 ; ;
1650 ; ;
1651 ; ;
1652 ; ;
1653 ; ;
1654 ; ;
1655 ; ;
1656 ; ;
1657 ; ;
1658 ; ;
1659 ; ;
1660 ; ;
1661 ; ;
1662 ; ;
1663 ; ;
1664 ; ;
1665 ; ;
1666 ; ;
1667 ; ;
1668 ; ;
1669 ; ;
1670 ; ;
1671 ; ;
1672 ; ;
1673 ; ;
1674 ; ;
1675 ; ;
1676 ; ;
1677 ; ;
1678 ; ;
1679 ; ;
1680 ; ;
1681 ; ;
1682 ; ;
1683 ; ;
1684 ; ;
1685 ; ;
1686 ; ;
1687 ; ;
1688 ; ;
1689 ; ;
1690 ; ;
1691 ; ;
1692 ; ;
1693 ; ;
1694 ; ;
1695 ; ;
1696 ; ;
1697 ; ;
1698 ; ;
1699 ; ;
1700 ; ;
1701 ; ;
1702 ; ;
1703 ; ;
1704 ; ;
1705 ; ;
1706 ; ;
1707 ; ;
1708 ; ;
1709 ; ;
1710 ; ;
1711 ; ;
1712 ; ;
1713 ; ;
1714 ; ;
1715 ; ;
1716 ; ;
1717 ; ;
1718 ; ;
1719 ; ;
1720 ; ;
1721 ; ;
1722 ; ;
1723 ; ;
1724 ; ;
1725 ; ;
1726 ; ;
1727 ; ;
1728 ; ;
1729 ; ;
1730 ; ;
1731 ; ;
1732 ; ;
1733 ; ;
1734 ; ;
1735 ; ;
1736 ; ;
1737 ; ;
1738 ; ;
1739 ; ;
1740 ; ;
1741 ; ;
1742 ; ;
1743 ; ;
1744 ; ;
1745 ; ;
1746 ; ;
1747 ; ;
1748 ; ;
1749 ; ;
1750 ; ;
1751 ; ;
1752 ; ;
1753 ; ;
1754 ; ;
1755 ; ;
1756 ; ;
1757 ; ;
1758 ; ;
1759 ; ;
1760 ; ;
1761 ; ;
1762 ; ;
1763 ; ;
1764 ; ;
1765 ; ;
1766 ; ;
1767 ; ;
1768 ; ;
1769 ; ;
1770 ; ;
1771 ; ;
1772 ; ;
1773 ; ;
1774 ; ;
1775 ; ;
1776 ; ;
1777 ; ;
1778 ; ;
1779 ; ;
1780 ; ;
1781 ; ;
1782 ; ;
1783 ; ;
1784 ; ;
1785 ; ;
1786 ; ;
1787 ; ;
1788 ; ;
1789 ; ;
1790 ; ;
1791 ; ;
1792 ; ;
1793 ; ;
1794 ; ;
1795 ; ;
1796 ; ;
1797 ; ;
1798 ; ;
1799 ; ;
1800 ; ;
1801 ; ;
1802 ; ;
1803 ; ;
1804 ; ;
1805 ; ;
1806 ; ;
1807 ; ;
1808 ; ;
1809 ; ;
1810 ; ;
1811 ; ;
1812 ; ;
1813 ; ;
1814 ; ;
1815 ; ;
1816 ; ;
1817 ; ;
1818 ; ;
1819 ; ;
1820 ; ;
1821 ; ;
1822 ; ;
1823 ; ;
1824 ; ;
1825 ; ;
1826 ; ;
1827 ; ;
1828 ; ;
1829 ; ;
1830 ; ;
1831 ; ;
1832 ; ;
1833 ; ;
1834 ; ;
1835 ; ;
1836 ; ;
1837 ; ;
1838 ; ;
1839 ; ;
1840 ; ;
1841 ; ;
1842 ; ;
1843 ; ;
1844 ; ;
1845 ; ;
1846 ; ;
1847 ; ;
1848 ; ;
1849 ; ;
1850 ; ;
1851 ; ;
1852 ; ;
1853 ; ;
1854 ; ;
1855 ; ;
1856 ; ;
1857 ; ;
1858 ; ;
1859 ; ;
1860 ; ;
1861 ; ;
1862 ; ;
1863 ; ;
1864 ; ;
1865 ; ;
1866 ; ;
1867 ; ;
1868 ; ;
1869 ; ;
1870 ; ;
1871 ; ;
1872 ; ;
1873 ; ;
1874 ; ;
1875 ; ;
1876 ; ;
1877 ; ;
1878 ; ;
1879 ; ;
1880 ; ;
1881 ; ;
1882 ; ;
1883 ; ;
1884 ; ;
1885 ; ;
1886 ; ;
1887 ; ;
1888 ; ;
1889 ; ;
1890 ; ;
1891 ; ;
1892 ; ;
1893 ; ;
1894 ; ;
1895 ; ;
1896 ; ;
1897 ; ;
1898 ; ;
1899 ; ;
1900 ; ;
1901 ; ;
1902 ; ;
1903 ; ;
1904 ; ;
1905 ; ;
1906 ; ;
1907 ; ;
1908 ; ;
1909 ; ;
1910 ; ;
1911 ; ;
1912 ; ;
1913 ; ;
1914 ; ;
1915 ; ;
1916 ; ;
1917 ; ;
1918 ; ;
1919 ; ;
1920 ; ;
1921 ; ;
1922 ; ;
1923 ; ;
1924 ; ;
1925 ; ;
1926 ; ;
1927 ; ;
1928 ; ;
1929 ; ;
1930 ; ;
1931 ; ;
1932 ; ;
1933 ; ;
1934 ; ;
1935 ; ;
1936 ; ;
1937 ; ;
1938 ; ;
1939 ; ;
1940 ; ;
1941 ; ;
1942 ; ;
1943 ; ;
1944 ; ;
1945 ; ;
1946 ; ;
1947 ; ;
1948 ; ;
1949 ; ;
1950 ; ;
1951 ; ;
1952 ; ;
1953 ; ;
1954 ; ;
1955 ; ;
1956 ; ;
1957 ; ;
1958 ; ;
1959 ; ;
1960 ; ;
1961 ; ;
1962 ; ;
1963 ; ;
1964 ; ;
1965 ; ;
1966 ; ;
1967 ; ;
1968 ; ;
1969 ; ;
1970 ; ;
1971 ; ;
1972 ; ;
1973 ; ;
1974 ; ;
1975 ; ;
1976 ; ;
1977 ; ;
1978 ; ;
1979 ; ;
1980 ; ;
1981 ; ;
1982 ; ;
1983 ; ;
1984 ; ;
1985 ; ;
1986 ; ;
1987 ; ;
1988 ; ;
1989 ; ;
1990 ; ;
1991 ; ;
1992 ; ;
1993 ; ;
1994 ; ;
1995 ; ;
1996 ; ;
1997 ; ;
1998 ; ;
1999 ; ;
2000 ; ;

```

```

1312 ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
1313 ;*CALL:
1314 ;*      MOV      NUM,=(SP)      ;;NUMBER TO BE TYPED
1315 ;*      TYPOC    ;;CALL FOR TYPEOUT
1316
1317 005412 017646 000000      0TYPOB: MOV      0(SP),=(SP)      ;;PICKUP THE MODE
1318 005416 116637 000001 005635      MOVB     1(SP),80FILL      ;;LOAD ZERO FILL SWITCH
1319 005424 112637 005637      MOVB     (SP)+,80MODE+1    ;;NUMBER OF DIGITS TO TYPE
1320 005430 062716 000002      ADD      02,(SP)          ;;ADJUST RETURN ADDRESS
1321 005434 000406      BR       0TYPON
1322 005436 112737 000001 005635      0TYPOC: MOVB     01,80FILL      ;;SET THE ZERO FILL SWITCH
1323 005444 112737 000006 005637      MOVB     06,80MODE+1    ;;SET FOR SIX(6) DIGITS
1324 005452 112737 000005 005634      0TYPON: MOVB     05,80CNT      ;;SET THE ITERATION COUNT
1325 005460 010346      MOV      R3,=(SP)          ;;SAVE R3
1326 005462 010446      MOV      R4,=(SP)          ;;SAVE R4
1327 005464 010546      MOV      R5,=(SP)          ;;SAVE R5
1328 005466 113704 005637      MOVB     80MODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
1329 005472 005404      NEG      R4
1330 005474 062704 000006      ADD      06,R4            ;;SUBTRACT IT FOR MAX, ALLOWED
1331 005500 110437 005636      MOVB     R4,80MODE        ;;SAVE IT FOR USE
1332 005504 113704 005635      MOVB     80FILL,R4        ;;GET THE ZERO FILL SWITCH
1333 005510 016608 000012      MOV      12(SP),R5        ;;PICKUP THE INPUT NUMBER
1334 005514 005003      CLR      R3                ;;CLEAR THE OUTPUT WORD
1335 005516 006108      10:     ROL      R5          ;;ROTATE MSB INTO "C"
1336 005520 000404      BR       30                ;;GO DO MSB
1337 005522 006108      20:     ROL      R5          ;;FORM THIS DIGIT
1338 005524 006108      ROL      R5
1339 005526 006108      ROL      R5
1340 005530 010503      MOV      R5,R3
1341 005532 006103      30:     ROL      R3          ;;GET LSB OF THIS DIGIT
1342 005534 105337 005636      DECB     80MODE            ;;TYPE THIS DIGIT?
1343 005540 100016      BPL      70                ;;BR IF NO
1344 005542 042703 177770      BIC      0177770,R3        ;;GET RID OF JUNK
1345 005546 001002      BNE      40                ;;TEST FOR 0
1346 005550 005704      TST      R4                ;;SUPPRESS THIS 0?
1347 005552 001403      BEQ      50                ;;BR IF YES
1348 005554 005204      40:     INC      R4          ;;DON'T SUPPRESS ANYMORE 0'S
1349 005556 052703 000060      BIS      0'0,R3           ;;MAKE THIS DIGIT ASCII
1350 005562 052703 000040      50:     BIS      0' ,R3     ;;MAKE ASCII IF NOT ALREADY
1351 005566 110337 005632      MOVB     R3,80            ;;SAVE FOR TYPING
1352 005572 104401 005632      TYPE     ,80              ;;GO TYPE THIS DIGIT
1353 005576 105337 005634      70:     DECB     80CNT      ;;COUNT BY 1
1354 005602 003347      BGT      20                ;;BR IF MORE TO DO
1355 005604 002402      BLT      60                ;;BR IF DONE
1356 005606 005204      INC      R4                ;;INSURE LAST DIGIT ISN'T A BLANK
1357 005610 000744      BR       20                ;;GO DO THE LAST DIGIT
1358 005612 012605      60:     MOV      (SP)+,R5    ;;RESTORE R5
1359 005614 012604      MOV      (SP)+,R4        ;;RESTORE R4
1360 005616 012603      MOV      (SP)+,R3        ;;RESTORE R3
1361 005620 016666 000002 000004      MOV      2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
1362 005626 012616      MOV      (SP)+,(SP)
1363 005630 000002      RTI
1364 005632      000      80:     ,BYTE  0          ;;STORAGE FOR ASCII DIGIT
1365 005633      000      ,BYTE  0          ;;TERMINATOR FOR TYPE ROUTINE

```

```

1366 005634 000          8OCNT: ,BYTE 0          ;;OCTAL DIGIT COUNTER
1367 005635 000          8OFILL: ,BYTE 0         ;;ZERO FILL SWITCH
1368 005636 000000      8OMODE: ,WORD 0          ;;NUMBER OF DIGITS TO TYPE
1369                               ,SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
1370
1371                               ;;*****
1372                               ;;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
1373                               ;;SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT, DEPENDING ON WHETHER THE
1374                               ;;NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
1375                               ;;BEFORE THE FIRST DIGIT OF THE NUMBER, LEADING ZEROS WILL ALWAYS BE
1376                               ;;REPLACED WITH SPACES.
1377                               ;;CALL:
1378                               ;;      MOV      NUM,-(SP)          ;;PUT THE BINARY NUMBER ON THE STACK
1379                               ;;      TYPDS          ;;GO TO THE ROUTINE
1380
1381 005640          8TYPDS:
1382 005640 010046      MOV      R0,-(SP)          ;;PUSH R0 ON STACK
1383 005642 010146      MOV      R1,-(SP)          ;;PUSH R1 ON STACK
1384 005644 010246      MOV      R2,-(SP)          ;;PUSH R2 ON STACK
1385 005646 010346      MOV      R3,-(SP)          ;;PUSH R3 ON STACK
1386 005650 010546      MOV      R5,-(SP)          ;;PUSH R5 ON STACK
1387 005652 012746 020200  MOV      020200,-(SP)          ;;SET BLANK SWITCH AND SIGN
1388 005656 016605 000020  MOV      20(SP),R5          ;;GET THE INPUT NUMBER
1389 005662 100004      BPL      10          ;;BR IF INPUT IS POS,
1390 005664 005405      NEG      R5          ;;MAKE THE BINARY NUMBER POS.
1391 005666 112766 000055 000001  MOVB   0'-' ,1(SP)          ;;MAKE THE ASCII NUMBER NEG.
1392 005674 005000      CLR      R0          ;;ZERO THE CONSTANTS INDEX
1393 005676 012703 006054      MOV      00DBLK,R3          ;;SETUP THE OUTPUT POINTER
1394 005702 112723 000040      MOVB   0' ,(R3)+          ;;SET THE FIRST CHARACTER TO A BLANK
1395 005706 005002      CLR      R2          ;;CLEAR THE BCD NUMBER
1396 005710 016001 006044      MOV      0DTBL(R0),R1          ;;GET THE CONSTANT
1397 005714 160105      SUB      R1,R5          ;;FORM THIS BCD DIGIT
1398 005716 002402      BLT     40          ;;BR IF DONE
1399 005720 005202      INC      R2          ;;INCREASE THE BCD DIGIT BY 1
1400 005722 000774      BR      30
1401 005724 060105      ADD      R1,R5          ;;ADD BACK THE CONSTANT
1402 005726 005702      TST     R2          ;;CHECK IF BCD DIGIT=0
1403 005730 001002      BNE     50          ;;FALL THROUGH IF 0
1404 005732 105716      TSTB   (SP)          ;;STILL DOING LEADING 0'S?
1405 005734 100407      BMI     70          ;;BR IF YES
1406 005736 106316      ASLB   (SP)          ;;MSD?
1407 005740 103003      BCC     60          ;;BR IF NO
1408 005742 116663 000001 177777  MOVB   1(SP),-1(R3)          ;;YES--SET THE SIGN
1409 005750 052702 000060      BIS     0'0,R2          ;;MAKE THE BCD DIGIT ASCII
1410 005754 052702 000040      BIS     0' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
1411 005760 110223      MOVB   R2,(R3)+          ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
1412 005762 005720      TST     (R0)+          ;;JUST INCREMENTING
1413 005764 020027 000010      CMP     R0,010          ;;CHECK THE TABLE INDEX
1414 005770 002746      BLT     20          ;;GO DO THE NEXT DIGIT
1415 005772 003002      BGT     00          ;;GO TO EXIT
1416 005774 010502      MOV     R5,R2          ;;GET THE LSD
1417 005776 000764      BR      60          ;;GO CHANGE TO ASCII
1418 006000 105726      001    TSTB   (SP)+          ;;WAS THE LSD THE FIRST NON-ZERO?
1419 006002 100003      BPL     90          ;;BR IF NO

```

```
1420 006004 116663 177777 177776          MOVB   -1(SP),-2(R3)    ;;YES--SET THE SIGN FOR TYPING
1421 006012 105013          CLRB   (R3)             ;;SET THE TERMINATOR
1422 006014 012609          MOV    (SP)+,R5        ;;POP STACK INTO R5
1423 006016 012603          MOV    (SP)+,R3        ;;POP STACK INTO R3
1424 006020 012602          MOV    (SP)+,R2        ;;POP STACK INTO R2
1425 006022 012601          MOV    (SP)+,R1        ;;POP STACK INTO R1
1426 006024 012600          MOV    (SP)+,R0        ;;POP STACK INTO R0
1427 006026 104401 006054          TYPE   ,SDBLK          ;;NOW TYPE THE NUMBER
1428 006032 016666 000002 000004          MOV    2(SP),4(SP)    ;;ADJUST THE STACK
1429 006040 012616          MOV    (SP)+,(SP)
1430 006042 000002          RTI                    ;;RETURN TO USER
1431 006044 023420          SDBLK: 10000,
1432 006046 001750          1000,
1433 006050 000144          100,
1434 006052 000012          10,
1435 006054 000004          SDBLK: ,BLKW 4
1436                                     ,SBTTL TYPE ROUTINE
1437
1438                                     ;;*****
1439                                     ;;ROUTINE TO TYPE ASCIZ MESSAGE, MESSAGE MUST TERMINATE WITH A 0 BYTE,
1440                                     ;;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED,
1441                                     ;;NOTE1:          SNULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER,
1442                                     ;;NOTE2:          SFILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED,
1443                                     ;;NOTE3:          SFILLC CONTAINS THE CHARACTER TO FILL AFTER,
1444                                     ;;
1445                                     ;;CALL:
1446                                     ;;1) USING A TRAP INSTRUCTION
1447                                     ;;      TYPE      ,NESADR          ;;NESADR IS FIRST ADDRESS OF AN ASCIZ STRING
1448                                     ;;OR
1449                                     ;;      TYPE
1450                                     ;;      NESADR
1451                                     ;;
1452
1453 006064 105737 001157          STYPE: TSTB   STPFLG    ;;IS THERE A TERMINAL?
1454 006070 100002          BPL    16              ;;BR IF YES
1455 006072 000000          HALT   0              ;;HALT HERE IF NO TERMINAL
1456 006074 000430          BR     36             ;;LEAVE
1457 006076 010046          18:   MOV    R0,-(SP)   ;;SAVE R0
1458 006100 017600 000002          MOV    02(SP),R0      ;;GET ADDRESS OF ASCIZ STRING
1459 006104 122737 000001 001214          CMPEQ  SPTENV,SENV    ;;RUNNING IN APT MODE
1460 006112 001011          BNE    628            ;;NO,GO CHECK FOR APT CONSOLE
1461 006114 132737 000100 001215          BITB   SPTSPool,SENVM ;;SPOOL MESSAGE TO APT
1462 006122 001405          BEQ    628            ;;NO,GO CHECK FOR CONSOLE
1463 006124 010037 006134          NOV    R0,616         ;;SETUP MESSAGE ADDRESS FOR APT
1464 006130 004737 006354          JSR    PC,SATY3       ;;SPOOL MESSAGE TO APT
1465 006134 000000          618:  ,WORD  0          ;;MESSAGE ADDRESS
1466 006136 132737 000040 001215          628:  BITB   SPTCSUP,SENVM ;;APT CONSOLE SUPPRESSED
1467 006144 001003          BNE    608            ;;YES,SKIP TYPE OUT
1468 006146 112046          28:   MOVB   (R0)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1469 006150 001005          BNE    48             ;;BR IF IT ISN'T THE TERMINATOR
1470 006152 005726          TST   (SP)+          ;;IF TERMINATOR POP IT OFF THE STACK
1471 006154 012600          608:  NOV    (SP)+,R0     ;;RESTORE R0
1472 006156 062716 000002          38:   ADD    02,(SP)    ;;ADJUST RETURN PC
1473 006162 000002          RTI
```

```

1474 006164 122716 000011      48:  CMPB  8HT,(SP)      ;;BRANCH IF <HT>
1475 006170 001430              BEQ    88
1476 006172 122716 000200      CMPB  8CRLF,(SP)     ;;BRANCH IF NOT <CRLF>
1477 006176 001006              BNE   58
1478 006200 005726              TST   (SP)+          ;;POP <CR><LF> EQUIV
1479 006202 104401              TYPE  ;;TYPE A CR AND LF
1480 006204 001171              8CRLF
1481 006206 105037 006342      CLRB  8CHARCNT      ;;CLEAR CHARACTER COUNT
1482 006212 000755              BR    28             ;;GET NEXT CHARACTER
1483 006214 004737 006276      58:  JSR   PC,8TYPEC     ;;GO TYPE THIS CHARACTER
1484 006220 123726 001156      68:  CMPB  8FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
1485 006224 001390              BNE   28             ;;IF NO GO GET NEXT CHAR.
1486 006226 013746 001154      MOV   8NULL,-(SP)   ;;GET 8 OF FILLER CHARS. NEEDED
1487                                ;;AND THE NULL CHAR.
1488 006232 105366 000001      78:  DECB  1(8P)        ;;DOES A NULL NEED TO BE TYPED?
1489 006236 002770              BLT   68             ;;BR IF NO--GO POP THE NULL OFF OF STACK
1490 006240 004737 006276      JSR   PC,8TYPEC     ;;GO TYPE A NULL
1491 006244 105337 006342      DECB  8CHARCNT      ;;DO NOT COUNT AS A COUNT
1492 006250 000770              BR    78             ;;LOOP
1493
1494                                ;HORIZONTAL TAB PROCESSOR
1495
1496 006252 112716 000040      88:  MOVB  8' ,(8P)     ;;REPLACE TAB WITH SPACE
1497 006256 004737 006276      98:  JSR   PC,8TYPEC     ;;TYPE A SPACE
1498 006262 132737 000007 006342  BITB  87,8CHARCNT   ;;BRANCH IF NOT AT
1499 006270 001372              BNE   98             ;;TAB STOP
1500 006272 005726              TST   (8P)+         ;;POP SPACE OFF STACK
1501 006274 000726              BR    28             ;;GET NEXT CHARACTER
1502 006276 105777 172646      8TYPEC: TSTB  88TPB      ;;WAIT UNTIL PRINTER IS READY
1503 006302 100375              BPL   8TYPEC
1504 006304 116677 000002 172640  MOVB  2(8P),88TPB   ;;LOAD CHAR TO BE TYPED INTO DATA REG.
1505 006312 122766 000015 000002  CMPB  8CR,2(8P)     ;;IS CHARACTER A CARRIAGE RETURN?
1506 006320 001003              BNE   18             ;;BRANCH IF NO
1507 006322 105037 006342      CLRB  8CHARCNT      ;;YES--CLEAR CHARACTER COUNT
1508 006326 000406              BR    8TYPEX
1509 006330 122766 000012 000002  18:  CMPB  8LF,2(8P)    ;;IS CHARACTER A LINE FEED?
1510 006336 001402              BEQ   8TYPEX         ;;BRANCH IF YES
1511 006340 105227              INCB  (8P)+         ;;COUNT THE CHARACTER
1512 006342 000000      8CHARCNT: WORD 0    ;;CHARACTER COUNT STORAGE
1513 006344 000207      8TYPEX: RTS   PC
1514
1515                                ;SBTTL APT COMMUNICATIONS ROUTINE
1516
1517                                ;;*****
1518 006346 112737 000001 006612  8ATTY1: MOVB  81,8FFLG   ;;TO REPORT FATAL ERROR
1519 006354 112737 000001 006610  8ATTY3: MOVB  81,8HFLG   ;;TO TYPE A MESSAGE
1520 006362 000403              BR    8ATYC
1521 006364 112737 000001 006612  8ATTY4: MOVB  81,8FFLG   ;;TO ONLY REPORT FATAL ERROR
1522 006372              8ATYC:
1523 006372 010046              NOV   R0,-(8P)      ;;PUSH R0 ON STACK
1524 006374 010146              NOV   R1,-(8P)      ;;PUSH R1 ON STACK
1525 006376 105737 006610      TSTB  8HFLG        ;;SHOULD TYPE A MESSAGE?
1526 006402 001450              BEQ   58             ;;IF NOT: BR
1527 006404 122737 000001 001214  CMPB  8APTENV,8ENV  ;;OPERATING UNDER APT?
    
```

```

1528 006412 001031      DNE      30      ;;IF NOT: BR
1529 006414 132737 000100 001215  BITB    @APTSPOOL,SENV  ;;SHOULD SPOOL MESSAGES?
1530 006422 001428      BEQ      30      ;;IF NOT: BR
1531 006424 017600 000004      NOV     04(SP),R0      ;;GET MESSAGE ADDR,
1532 006430 062766 000002 000004      ADD     02,4(SP)      ;;BUMP RETURN ADDR,
1533 006436 008737 001174      TST     MSGTYPE      ;;SEE IF DONE W/ LAST XMISSION?
1534 006442 001378      DNE      10      ;;IF NOT: WAIT
1535 006444 010037 001210      NOV     R0,MSGAD      ;;PUT ADDR IN MAILBOX
1536 006450 108720 201      TSTB   (R0)+        ;;FIND END OF MESSAGE
1537 006452 001376      DNE      20
1538 006454 163700 001210      SUB     MSGAD,R0      ;;SUB START OF MESSAGE
1539 006460 006200      ASR     R0           ;;GET MESSAGE LNTH IN WORDS
1540 006462 010037 001212      NOV     R0,MSGGLT     ;;PUT LENGTH IN MAILBOX
1541 006466 012737 000004 001174      NOV     04,MSGTYPE    ;;TELL APT TO TAKE MSG.
1542 006474 000413      BR
1543 006476 017637 000004 006522 301      NOV     04(SP),48     ;;PUT MSG ADDR IN JSR LINKAGE
1544 006504 062766 000002 000004      ADD     02,4(SP)      ;;BUMP RETURN ADDRESS
1545 006512 013746 177776      NOV     177776,-(SP)  ;;PUSH 177776 ON STACK
1546 006516 004737 006064      JSR     PC,0TYPE     ;;CALL TYPE MACRO
1547 006522 000000      ,WORD  0
1548 006524      501
1549 006524 108737 006612 1001      TSTB   0FFLG        ;;SHOULD REPORT FATAL ERROR?
1550 006530 001416      BEQ     120         ;;IF NOT: BR
1551 006532 008737 001214      TST     SENV        ;;RUNNING UNDER APT?
1552 006536 001413      BEQ     120         ;;IF NOT: BR
1553 006540 008737 001174 1101      TST     MSGTYPE     ;;FINISHED LAST MESSAGE?
1554 006544 001378      DNE     110         ;;IF NOT: WAIT
1555 006546 017637 000004 001176      NOV     04(SP),0FATAL ;;GET ERROR 0
1556 006554 062766 000002 000004      ADD     02,4(SP)      ;;BUMP RETURN ADDR,
1557 006562 005237 001174      INC     MSGTYPE     ;;TELL APT TO TAKE ERROR
1558 006566 108037 006612 1201      CLRB   0FFLG        ;;CLEAR FATAL FLAG
1559 006572 108037 006611      CLRB   0LFLG        ;;CLEAR LOG FLAG
1560 006576 108037 006610      CLRB   0MFLG        ;;CLEAR MESSAGE FLAG
1561 006602 012601      NOV     (SP)+,R1     ;;POP STACK INTO R1
1562 006604 012600      NOV     (SP)+,R0     ;;POP STACK INTO R0
1563 006606 000207      RTS     PC           ;;RETURN
1564 006610      000      0MFLG: ,BYTE 0      ;;MESSAGE FLAG
1565 006611      000      0LFLG: ,BYTE 0      ;;LOG FLAG
1566 006612      000      0FFLG: ,BYTE 0      ;;FATAL FLAG
1567      006614      ,EVEN
1568      000200      APTSIZE=200
1569      000001      APTENV=001
1570      000100      APTSPOOL=100
1571      000040      APTCSUP=040
1572
1573
1574      ;
1575      ;THIS ROUTINE WILL PROTECT THE PROGRAM
1576      ;FROM INTERRUPTS,
1577      ;
1578      ;THE TRAP CATCHER IS SET UP FOR
1579      ;      ,WORD  ,+2
1580      ;      ,WORD  JSR      PC,R0
1581      ;
1581      ;ILLEGAL INTERRUPTS OR INTERRUPTS TO THE WRONG VECTOR

```

```

1582                           ;GOTO THE VECTOR AND PICK UP THE "+2" AS AN ADDRESS
1583                           ;AND "4700" AS NEW STATUS,
1584                           ;THE +2 AS A PC WILL CAUSE EXECUTION OF THE "JSR PC,R0" (AN ILLEGAL INSTR),
1585                           ;AND TRAP TO LOCATION "4", IN LOCATION 10 WE HAVE A
1586                           ;POINTER HERE, IF THIS CONDITION CAUSES A TRAP TO LOC,4
1587                           ;WE WILL REPORT IT IN THE SAME MANNER THAT WE WOULD
1588                           ;REPORT ANY OTHER ERROR,
1589
1590                           ;IF A BUSS ERROR TRAP DID OCCUR AND CAUSE A TRAP TO 4,
1591                           ;WE WILL HALT,
1592
1593   006614   011637   006666           IOTRD:   MOV     (6),TRTO           ;GET WHERE WE CAME TO,
1594   006620   162737   000004   006666       SUB     #4,TRTO           ;FORM REAL ADDR,
1595
1596   006626   023727   006666   001000       CMP     TRTO,#1000       ;DID TRAP COME FROM LESS THAN ADDR, 1000?
1597   006634   003402                       BLE     #2                ;
1598
1599   006636   000000                       10:     HALT                ;NO! MUST BE A BUSS ILLEGAL ADDR, TIME OUT,
1600                                                                                   ;ADDRESS CONTAINED IN TRTO,
1601
1602   006640   000776                       20:     BR      #10             ;DON'T ALLOW A CONTINUE
1603   006642                                                                           20:
1604   006642   113737   001102   006672       MOVB   #STNM,TSTNM       ;SAVE TEST NUMBER,
1605
1606   006650   016637   000004   006670       MOV     4(6),TRFRO       ;GET TRAPPED FROM ADDR,
1607   006656   062706   000004               ADD     #4,R6
1608
1609   006662   104006                       ERROR   6               ;ERROR! ILLEGAL INTERRUPT
1610                                                                                   ;OR INTERRUPT TO WRONG
1611                                                                                   ;VECTOR - IF TEST NUMBER
1612                                                                                   ;IS LESS THAN 2, ITS LIKELY
1613                                                                                   ;(BUT NOT EXCLUSIVELY) TO BE A
1614                                                                                   ;DEVICE OTHER THAN THE DEVICE
1615                                                                                   ;TO BLAME,
1616                                                                                   ;IF THE INTERRUPT OCCURRED
1617                                                                                   ;DURING AN INTERRUPT TEST, I'D
1618                                                                                   ;SUSPECT A PROBLEM WITH THE
1619                                                                                   ;DEVICE,
1620                                                                                   ;IF THE ADDRESS THE INTERRUPT
1621                                                                                   ;VECTOR TO IS WITHIN THE RANGE
1622                                                                                   ;OF VECTORS ASSIGNED TO THE DEVICE,
1623                                                                                   ;THEN I'D SUSPECT THE DEVICE
1624                                                                                   ;INTERRUPTED ILLEGALLY,
1625                                                                                   ;IF THE ADDRESS THE INTERRUPT
1626                                                                                   ;VECTORED TO IS OUTSIDE OF THE
1627                                                                                   ;RANGE ASSIGNED TO THE DEVICE,
1628                                                                                   ;I'D SUSPECT THAT THE
1629                                                                                   ;DEVICE PUT THE WRONG VECTOR ON
1630                                                                                   ;THE BUSS DURING THE INTERRUPT
1631                                                                                   ;PROCESS,
1632                                                                                   ;FOR THIS ERROR - DON'T
1633                                                                                   ;USE "LOOP ON ERROR" OPTION,
1634                                                                                   ;ALSO EXPECT THE INTERRUPT TEST TO
1635                                                                                   ;REPORT THAT THE DEVICE DIDN'T

```

74


```
1744 007322 000
1745
1746 007324
1747 007324 001116 001362 001126 DT1: .EVEN
1748 007332 001124 000000 .WORD SERRPC,RADDR,SDDAT,SGDDAT,0
1749
1750 007336 001116 001126 001124 DT3: .WORD SERRPC,SDDAT,SGDDAT,0
1751 007344 000000
1752
1753 007346 001116 001336 000000 DT4: .WORD SERRPC,LKS,0
1754
1755 007354 006672 001116 006666 DT6: .WORD TSTNM,SERRPC,TRTO,TRFRO
1756 007362 006670
1757
1758 007364 000000 000000 DF0: .WORD 0,0
1759
1760 007370 000000 NENST: .WORD 0 ;MEMORY WILL BE WRITTEN
1761 ;FROM HERE UP
1762
1763 000001 .END
```

ABASE	=	177546	1550	240	201	347			
ACDW1	=	000000	240	203					
ACDW2	=	000000	240						
ACPUOP	=	000000	240	255					
ADCK	=	001356	3500	417*	536*	557	953		
ADDW0	=	000000	240						
ADDW1	=	000000	240						
ADDW10	=	000000	240						
ADDW11	=	000000	240						
ADDW12	=	000000	240						
ADDW13	=	000000	240						
ADDW14	=	000000	240						
ADDW15	=	000000	240						
ADDW2	=	000000	240						
ADDW3	=	000000	240						
ADDW4	=	000000	240						
ADDW5	=	000000	240						
ADDW6	=	000000	240						
ADDW7	=	000000	240						
ADDW8	=	000000	240						
ADDW9	=	000000	240						
ADEVCT	=	000000	240	246					
ADEVH	=	000000	240	202					
AENV	=	000000	240	251					
AENVH	=	000000	240	252					
AFATAL	=	000000	240	243					
AMADR1	=	000000	240	260					
AMADR2	=	000000	240	272					
AMADR3	=	000000	240	275					
AMADR4	=	000000	240	278					
ANAMS1	=	000000	240	262					
ANAMS2	=	000000	240	270					
ANAMS3	=	000000	240	273					
ANAMS4	=	000000	240	276					
AMSGAD	=	000000	240	248					
AMSGLG	=	000000	240	249					
AMSGTY	=	000000	240	242					
AMTYP1	=	000000	240	263					
AMTYP2	=	000000	240	271					
AMTYP3	=	000000	240	274					
AMTYP4	=	000000	240	277					
APASS	=	000000	240	245					
APRIOR	=	000000	240						
APTC&U	=	000040	1466	15710					
APTENV	=	000001	1022	1459	1527	15690			
APTSIZ	=	000200	408	15680					
APTSPO	=	000100	1461	1529	15700				
ASWREG	=	000000	240	253					
ATESTN	=	000000	240	244					
AUNIT	=	000000	240	247					
AUSWR	=	000000	240	254					
AVECT1	=	000100	1560	240	279	349	350		
AVECT2	=	000000	240	280					
BAD	=	001364	3610	418*	851	857*	888*	903	

BIT0	=	000001	1390	770																
BIT00	=	000001	1290	139																
BIT01	=	000002	1280	138																
BIT02	=	000004	1270	137																
BIT03	=	000010	1260	136																
BIT04	=	000020	1250	135																
BIT05	=	000040	1240	134																
BIT06	=	000100	1230	133																
BIT07	=	000200	1220	132																
BIT08	=	000400	1210	131	1119															
BIT09	=	001000	1200	130	1033	1127														
BIT1	=	000002	1380																	
BIT10	=	002000	1190	1010																
BIT11	=	004000	1180	1134																
BIT12	=	010000	1170																	
BIT13	=	020000	1160	1017																
BIT14	=	040000	1150	1105																
BIT15	=	100000	1140																	
BIT2	=	000004	1370																	
BIT3	=	000010	1360																	
BIT4	=	000020	1350																	
BIT5	=	000040	1340																	
BIT6	=	000100	1330	574	576	595	626	960												
BIT7	=	000200	1320	694																
BIT8	=	000400	1310																	
BIT9	=	001000	1300																	
BLKCK	=	001366	3620	4790	8020	859	8640	925												
BPTVEC	=	000014	1460																	
CKEND	=	003710	937	9490																
CKSWR	=	104407	1006	1032	1103	1104	16860													
CK1	=	003652	930	936	9360	940	947													
CR	=	000015	540	1505	1515															
CRLF	=	000200	550	447	1476	1515														
DDISP	=	177570	610	221	396															
DFC	=	007364	300	315	322	329	336	343	17500											
DH1	=	007165	306	313	17240															
DH3	=	007222	320	17300																
DH4	=	007247	327	334	17350															
DH6	=	007265	341	17390																
DISPLA	=	001142	2210	3960	4040	10090	11490													
DISPRE	=	000174	360	404																
DSWR	=	177570	600	220	395															
DT1	=	007324	307	314	17470															
DT3	=	007336	321	17500																
DT4	=	007346	320	335	17530															
DT6	=	007354	342	17550																
EMTVEC	=	000030	1400	3820	3830															
EM1	=	006754	305	16920																
EM2	=	007005	312	16980																
EM3	=	007051	319	17060																
EM4	=	007072	326	17100																
EM5	=	007115	333	17150																
EM6	=	007144	340	17200																
ERRVEC	=	000004	1420	393	3940	4050	526	5270	5330	5500	1110	11110	11130	11160						

55

GNS	U	446	816	822	831	983	1678	1679	1680	1681	1682	1684	1686	1687
GTSWR	104406	441	16848											
HILIM	001344	3528	484	502	775	928	932							
HT	000011	528	1474	1515										
IOTRD	006614	34	15938											
IOTVEC	000020	1478	3808	3818										
KVECT	001340	3498	6438	6668	6678	7268	7498	7508						
KVECTP	001342	3508												
LF	000012	538	1509	1515										
LKS	001336	3478	529	5348	5748	575	576	5938	594	6268	629	636	6688	6838
		686	693	7008	702	8678	956	1753						
LMWR	001370	3638	3708	4728	4788	484	4918	4928	935					
LOLIM	001346	3538	368	472	497	773	862	929	931					
LOOP	002002	4508	843											
LOW2	001350	3548	862											
MCLK	003020	523	7528											
MEMST	007370	353	354	17608										
MEMWR	001360	3598	4718	4958	8658	933								
NOCLK	001372	3648	521	826										
PC	0000007	738	8078	8108	8378	842	10198	10258	10788	12078	14648	14838	14908	14978
		15118	15138	15468	15638									
PCOUNT	001352	3568	4168	824	9498									
PDBAD	003352	8538	904											
PDOWN	003344	413	8518	854	917									
PD2	003356	852	8578	912										
PD3	003554	911	9178											
PFLAG	001354	3578	5638	578	597	6158	633	655	6818	690	704	711	7248	741
		7688	9078	910										
PIRG	177772	598												
PIRGVE	000240	1538												
PRO	000000	768												
PR1	000040	778												
PR2	000100	788												
PR3	000140	798												
PR4	000200	808												
PR5	000240	818												
PR6	000300	828												
PR7	000340	838												
PS	177776	568	57											
PSW	177776	578												
PUEND	003746	954	961	9658										
PUP	003506	887	9028											
PUPBD	004026	905	9878	988										
PU2	003532	9108	915											
PWRVEC	000024	1488	4138	4148	8878	9058	9128	9178						
RADDR	001362	3608	5088	5098	510	7808	7818	782	876	9428	9438	9668	1747	
RDCHR	104410	1266	16878											
RDLIN	104411	16888												
RESVEC	000010	1438												
RO	0000000	648	3718	8348	837	868	8788	8808	9748	9768	978	1050	10518	10528
		10598	10608	10618	10628	10638	1064	1069	10748	10768	1080	1082	1382	13928
		1396	1412	1413	14268	1457	14588	1463	1468	14718	1523	15318	1535	1536
		15388	15398	1540	15628	1655	16568	1657	16588	16598	16608	16618		

R1	=0000001	650	473*	476*	478	491	505	511	769*	772*	778	783	869	973*
		1383	1396*	1397	1401	1425*	1524	1561*						
R2	=0000002	660	498*	502	508	627*	631*	684*	688*	738*	739*	773*	775	780
		870	972*	1384	1395*	1399*	1402	1409*	1410*	1411	1416*	1424*		
R3	=0000003	670	368*	370	497*	498	871	971*	1262	1263*	1264	1267*	1268	1272
		1274	1276*	1278*	1325	1334*	1340*	1341*	1344*	1349*	1350*	1351	1360*	1365
		1393*	1394*	1408*	1411*	1420*	1421*	1423*						
R4	=0000004	680	872	929*	931*	936	942	970*	1326	1328*	1329*	1330*	1331	1332*
		1346	1348*	1356*	1359*									
R5	=0000005	690	873	928*	932*	935*	936	969*	1327	1333*	1335*	1337*	1338*	1339*
		1340	1358*	1386	1388*	1390*	1397*	1401*	1416	1422*				
R6	=0000006	700	374*	375*	376	664*	745*	877	902*	1607*				
R7	=0000007	710												
SP	=0000006	720	378*	393*	401*	405	818*	824*	1014	1035*	1038*	1050*	1055*	1076
		1080*	1110*	1113	1115	1116	1145	1146	1150*	1167*	1168*	1169	1176*	1179*
		1180*	1184*	1188*	1189	1192*	1196	1198	1200	1201*	1208	1210	1212*	1213
		1215*	1216*	1217*	1218*	1219*	1234*	1235*	1238*	1239*	1240	1244*	1245*	1246
		1249	1251	1253*	1262*	1267	1278	1279*	1280*	1281*	1317*	1318	1319	1320*
		1325*	1326*	1327*	1333	1338	1359	1360	1361*	1362*	1382*	1383*	1384*	1385*
		1386*	1387*	1388	1391*	1404	1406*	1408	1418	1420	1422	1423	1424	1425
		1426	1428*	1429*	1457*	1458	1468*	1470	1471	1472*	1474	1476	1478	1484
		1486*	1488*	1496*	1500	1504	1505	1509	1523*	1524*	1531	1532*	1543	1544*
		1545*	1555	1556*	1561	1562	1655*	1656	1666*	1667*				
SR6	003504	877*	897*	902										
STACK	= 001100	478	378											
START	001374	42	3680											
STKLMT	= 177774	580												
SWR	001140	2200	376	395*	397	403*	410*	439	1010	1017	1029	1033	1105	1119
		1121	1127	1134	1163	1200*								
SWREG	000176	378	403	439	1163	1176								
SW0	= 000001	1110												
SW00	= 000001	1010	111											
SW01	= 000002	1000	110											
SW02	= 000004	990	109											
SW03	= 000010	980	108											
SW04	= 000020	970	107											
SW05	= 000040	960	106											
SW06	= 000100	950	105											
SW07	= 000200	940	104											
SW08	= 000400	930	103											
SW09	= 001000	920	102											
SW1	= 000002	1100												
SW10	= 002000	910												
SW11	= 004000	900												
SW12	= 010000	890												
SW13	= 020000	880												
SW14	= 040000	870												
SW15	= 100000	860												
SW2	= 000004	1090												
SW3	= 000010	1080												
SW4	= 000020	1070												
SW5	= 000040	1060												
SW6	= 000100	1050												
SW7	= 000200	1040												

55

SUSWR	001220	2540																	
SVECT1	001244	2790																	
SVECT2	001246	2800																	
SXTSTR	004370	11000																	
SSGET4	000000	8360																	
SOFILL	005635	13100	13220	1332	13670														
S40CAT	***** U	1019	1105																
.	= 007372	270	330	350	380	410	164	1650	1670	1690	1700	176	1770	1790					
		1010	1990	236	377	389	390	4470	8320	844	8450	9040	1041	10070					
		1152	1153	1156	12050	1206	1292	14350	1515	15670	17460								
SASTA	***** U	1519	1522																
SX	= 001000	1700	101																

COMMEN	1540														
DFC	3020	300	315	322	329	336	343								
ENDCOM	1540														
ENDPAS	7930	813													
ERROR	400	513	539	545	583	601	639	659	696	715	747	785	945	962	1609
ESCAPE	1540														
GETPRI	1540														
GETSWR	1540	4340													
MULT	1540														
NEWTST	1540	453	516	552	610	670	710	755	788						
POP	1540	1422	1561	1562											
PR	1590	419	564	616	644	728									
PUSH	1540	1301	1522	1524	1545										
REPORT	1540														
SCOPE	490	519	555	613	679	721	764	791							
SETPRI	1540														
SETTRA	16700	1679	1680	1681	1682	1684	1686	1687	1688						
SETUP	1540	372													
SKIP	1540	503	590	599	641	698	709	776							
SLASH	1540														
SPACE	1540														
STARS	1540	162	173	175	182	195	236	239	453	463	516	518	552	554	610
	612	670	670	710	720	755	763	788	790	796	993	1043	1090	1155	1188
	1226	1255	1294	1371	1438	1517	1649								
SWRSU	1540	3910													
TRMTRP	16700														
TYPBIN	1540														
TYPDEC	1540														
TYPNAM	1540	429													
TYPNUM	1540														
TYPOCS	10	1540													
TYPOCT	1540	818	824	1055	1079	1176									
TYPTXT	1540	814	820	829	981										
ZZ1	4520	455	6690	672	7540	757									
88CHRE	1930														
88CMTN	1930														
88ESCA	1540														
88NEWT	1540	453	516	552	610	670	710	755	788						
88SET	16700	1679	1680	1681	1682	1684	1686	1687	1688						
88SETH	4070														
88SKIP	1540	503	590	599	641	698	709	776							
,EQUAT	10	44													
,HEADE	10														
,SETTR	10														
,SETUP	10	367													
,SWRHI	10	13													
,SWRLO	240														
,TRMTR	10														
,SACT1	10	160													
,SAPT0	2370														
,SAPTH	10	171													
,SAPTY	10	1518													
,SCMTA	10	193													
,SEOP	10	794													

.SERRO	10	991
.SERRT	10	1041
.SREAD	10	1153
.SSCOP	10	1088
.STRAP	10	1647
.STYPD	10	1369
.STYPE	10	1436
.STYPO	10	1292

15
4

	492	664	667	745	750	1063	1192	1201	1320	1330	1401	1472	1532	1544	1556
ADD	492	664	667	745	750	1063	1192	1201	1320	1330	1401	1472	1532	1544	1556
ASL	1607														
ASLB	1060	1061	1062	1215	1216	1217	1659								
ASR	1406														
BCC	1539														
BEQ	1407														
	409	438	475	485	503	506	522	771	776	779	827	835	852	860	863
	923	926	934	937	940	954	961	975	1008	1011	1034	1037	1065	1070	1083
	1120	1122	1124	1128	1137	1172	1199	1214	1347	1462	1475	1510	1526	1530	1550
	1552														
BGE	1140														
BGT	809	1211	1252	1354	1415										
BHI	1126														
BIC	806	1168	1185	1212	1239	1245	1253	1344							
BIS	574	1219	1349	1350	1409	1410									
BISB	1052														
BIT	576	595	960	1010	1017	1033	1105	1119	1127	1134					
BITB	408	474	770	922	1461	1466	1498	1529							
BLE	1597														
BLO	486														
BLOS	1265														
BLT	1209	1250	1355	1398	1414	1489									
BMI	630	687	1405												
BNE	377	398	432	436	440	558	577	579	596	598	632	634	656	689	691
	703	708	712	740	742	881	904	911	980	1018	1023	1053	1075	1106	1135
	1164	1170	1190	1197	1204	1241	1247	1269	1275	1345	1403	1460	1467	1469	1477
	1485	1499	1506	1528	1534	1537	1554								
BPL	1030	1166	1182	1237	1243	1343	1389	1419	1454	1503					
BR	400	442	445	490	493	537	543	560	590	599	641	662	698	709	743
	815	821	830	854	891	915	930	947	982	988	1028	1058	1085	1108	1114
	1117	1130	1133	1193	1220	1222	1248	1271	1321	1336	1357	1400	1417	1456	1482
	1492	1501	1508	1520	1542	1602									
CLR	371	375	386	387	407	415	416	417	418	473	479	495	563	592	593
	615	627	648	681	683	684	700	701	724	738	768	769	803	804	864
	867	888	921	957	1051	1132	1147	1179	1180	1334	1392	1395			
CLRB	1131	1276	1421	1481	1507	1558	1559	1560							
CMP	376	397	439	484	502	505	775	778	862	910	936	939	1115	1139	1163
	1169	1189	1194	1208	1210	1240	1246	1249	1251	1264	1413	1596			
CMPB	437	1022	1121	1125	1171	1203	1268	1274	1459	1474	1476	1484	1505	1509	1527
COM	927														
DEC	688	807	880	1059											
DECB	976	1342	1353	1488	1491										
ENT	48														
HALT	487	559	853	890	914	987	1031	1455	1599						
INC	431	631	802	805	857	865	907	949	1013	1138	1218	1348	1356	1399	1557
INCB	739	1007	1143	1511											
IOT	49														
JMP	42	523	842												
JSR	837	1019	1025	1207	1464	1483	1490	1497	1546						
NOV	368	369	370	374	378	380	381	382	383	384	385	389	390	393	394
	395	396	401	403	404	405	410	413	414	421	422	465	438	469	470
	471	472	476	478	491	497	498	508	510	511	526	527	529	533	534
	535	550	566	567	575	581	594	619	620	626	636	637	643	646	647
	666	693	694	702	722	726	730	731	749	752	768	772	773	780	782

JS
62

	783	810	818	824	834	868	869	870	871	872	873	874	875	876	877
	878	879	887	902	905	912	917	924	928	929	931	932	938	941	942
	956	966	967	968	969	970	971	972	973	974	1009	1014	1035	1038	1050
	1055	1064	1069	1074	1076	1080	1110	1111	1113	1116	1129	1141	1142	1148	1146
	1149	1150	1176	1200	1205	1234	1235	1262	1263	1278	1279	1280	1281	1317	1325
	1326	1327	1333	1340	1358	1359	1360	1361	1362	1382	1383	1384	1385	1386	1387
	1388	1393	1396	1416	1422	1423	1424	1425	1426	1428	1429	1457	1458	1463	1471
	1486	1523	1524	1531	1535	1540	1541	1543	1545	1555	1561	1562	1593	1606	1655
	1656	1660	1666	1667											
MOV8	388	443	467	1016	1024	1144	1148	1167	1184	1238	1244	1267	1272	1318	1319
	1322	1323	1324	1328	1331	1332	1351	1391	1394	1408	1411	1420	1468	1496	1504
	1518	1519	1521	1604	1658										
NEG	1329	1390													
NOP	464	654	838	839	840										
RESET	836														
ROL	1335	1337	1338	1339	1341										
RTI	402	423	868	621	648	732	985	1040	1151	1206	1254	1282	1363	1430	1473
	1641	1668													
RTS	1078	1513	1563	1661											
SUB	509	781	943	1015	1397	1538	1594								
TRAP	1670	1679	1688	1681	1682	1684	1686	1687	1688						
TST	435	821	857	878	897	633	655	690	704	711	741	826	851	859	903
	925	933	953	978	1029	1036	1082	1112	1136	1198	1213	1346	1402	1412	1470
	1478	1500	1533	1551	1553	1657									
TSTB	629	686	1123	1165	1181	1236	1242	1404	1418	1483	1502	1525	1536	1549	
,ASCII	233	234													
,ASCIZ	232	235	447	817	823	832	974	1086	1286	1287	1288	1290	1692	1698	1706
	1710	1715	1720	1724	1730	1735	1719								
,BLKB	1285														
,BLKW	1435														
,BYTE	202	203	208	209	217	218	224	227	228	229	251	252	262	263	270
	271	273	274	276	277	844	1026	1027	1283	1284	1364	1365	1366	1367	1864
	1565	1566													
,DSABL	1223														
,ENABL	1	1156													
,END	1763														
,ENDC	6	20	22	23	24	48	140	154	163	167	169	174	176	183	196
	200	202	238	231	232	233	237	240	262	270	273	276	279	288	281
	282	283	286	367	378	379	382	384	386	387	389	391	412	433	439
	445	447	454	455	463	464	465	466	804	817	818	819	820	853	854
	555	556	591	600	611	612	613	614	642	671	672	678	679	688	699
	710	719	728	721	722	723	756	757	763	764	765	766	777	788	798
	791	792	797	799	808	803	809	812	813	817	823	832	834	836	842
	844	845	984	994	997	1007	1014	1019	1020	1021	1029	1040	1041	1044	1059
	1088	1091	1094	1099	1105	1107	1118	1121	1122	1123	1125	1127	1134	1138	1143
	1145	1149	1152	1153	1156	1157	1159	1187	1223	1227	1255	1256	1263	1265	1268
	1270	1286	1292	1298	1372	1439	1468	1518	1519	1522	1549	1564	1660	1666	1689
	1678	1679	1688	1681	1682	1683	1684	1685	1686	1687	1688	1689			
,EQUIV	48	49	57	102	103	104	105	106	107	108	109	110	111	130	131
	132	133	134	135	136	137	138	139							
,EVEN	240	447	817	823	832	845	984	1087	1567	1746					
,IF	2	20	21	22	23	24	46	112	140	162	165	167	173	175	182
	195	199	201	230	231	232	236	237	239	262	270	273	276	279	288
	281	282	283	284	286	287	373	378	388	382	384	386	387	389	407

	432	433	434	437	446	453	455	463	465	466	503	516	518	520	552
	554	556	590	599	610	612	614	641	670	672	678	680	698	709	718
	720	722	723	755	757	763	765	766	776	788	790	792	796	797	798
	799	800	802	808	811	813	816	822	831	834	836	842	844	845	983
	993	996	1007	1010	1017	1019	1020	1022	1029	1033	1040	1041	1043	1058	1074
	1090	1093	1098	1104	1105	1117	1119	1120	1121	1123	1124	1125	1134	1136	1144
	1146	1151	1152	1153	1155	1157	1158	1159	1187	1226	1227	1255	1263	1264	1268
	1269	1285	1286	1292	1294	1371	1438	1459	1517	1519	1522	1549	1564	1649	1655
	1659	1670	1679	1680	1681	1682	1683	1684	1686	1687	1688	1689			
.IFF	20	22	23	24	46	163	167	169	174	176	183	196	199	202	230
	237	240	378	432	433	453	454	455	464	466	504	517	518	519	520
	553	554	555	556	591	600	611	612	613	614	642	671	672	679	680
	699	710	719	720	721	722	756	757	764	765	777	789	790	791	792
	797	799	802	809	812	844	994	996	1010	1040	1041	1044	1059	1088	1091
	1118	1121	1122	1125	1152	1156	1159	1227	1229	1234	1255	1256	1265	1269	1286
	1295	1372	1439	1518	1650	1656									
.IFT	447	817	823	832	984	1020	1133	1229	1234						
.IFTF	447	817	823	832	984	1019	1131	1174	1227	1230					
.IIF	1	6	11	12	17	18	19	20	23	24	236	240	379	382	386
	387	389	390	433	798	803	804	819	825	844	845	997	998	999	1000
	1001	1006	1032	1040	1041	1056	1081	1094	1095	1096	1097	1098	1099	1103	1132
	1133	1149	1152	1153	1156	1177	1278	1286	1292	1515	1678	1679	1680	1681	1682
	1684	1686	1687	1688											
.IRP	367	483	516	552	610	670	718	755	788	802	1104	1382	1422	1523	1524
	1545	1561	1562												
.LIST	1	23	33	154	230	237	240	367	391	433	434	447	453	465	516
	520	552	556	610	614	670	680	718	722	755	765	788	792	803	817
	823	832	836	984	1040	1098	1255	1670	1678	1679	1680	1681	1682	1683	1684
	1685	1686	1687	1688	1689										
.MACRO	24	159	193	302	407	482	669	754	793	1670					
.MCALL	1	154	237	391	434										
.MEXIT	283														
.NLIST	1	23	33	154	230	237	240	367	391	433	434	447	453	465	516
	520	552	556	610	614	670	680	718	722	755	765	788	792	803	817
	823	832	836	984	1040	1098	1255	1670	1678	1679	1680	1681	1682	1683	1684
	1685	1686	1687	1688	1689										
.PAGE	193	206													
.REPT	33														
.SBTTL	13	25	44	160	171	193	237	286	372	429	434	453	516	552	610
	670	718	755	788	794	919	951	991	1041	1088	1153	1292	1369	1436	1515
	1647	1670	1690												
.TITLE	1														
.WORD	33	34	36	37	39	168	187	188	189	190	191	192	201	204	205
	206	207	210	211	212	213	214	215	216	219	220	221	242	243	244
	245	246	247	248	249	253	254	255	268	272	275	278	279	280	281
	282	283	347	349	350	352	353	354	356	357	358	359	360	361	362
	363	364	808	811	843	1067	1072	1368	1468	1812	1847	1843	1844	1845	1877
	1747	1750	1753	1755	1758	1760									

MAINDEC-11-DVKPA-A
DVKPA,P11

NACT11 27(663) 2-MAY-77 15:03 PAGE 48

SEG 0065

*DVKPA,DVKPA/SOL/CRF=DVKPA
RUN-TIME: 73 22 4 SECONDS
CORE USED: 25K

25
25