

KW11-L

LINE FREQUENCY CLOCK TEST
MD-11-DZKWA-F

EP-DZKWA-F-DL
COPYRIGHT © 70-78
FICHE 1 OF 1

MAR 1978
digital
MADE IN USA

This microfiche card contains a grid of frames. The leftmost column consists of frames with text, likely a table of contents or index. The remaining frames contain waveforms and data plots, which are typical for a clock test report. The text in the frames is too small to be legible, but it appears to include various parameters and test results.

B01

EOF102420000411

00010000

780223 IDENTIFICATION

21H0R1DZKWFSEQ

00010000

780223
SEQ 0001

PRODUCT CODE: MAINDEC-11-DZKWA-F-D
PRODUCT NAME: LINE FREQUENCY CLOCK TEST
PRODUCT DATE: FEB 1978
MAINTAINER: DIAGNOSTIC ENGINEERING

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE OR EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1970, 1978 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

1.0 GENERAL PROGRAM INFORMATION

- 1.1 ABSTRACT
THIS PROGRAM TESTS THE KW11 LINE FREQUENCY CLOCK. IT VALIDATES PROPER OPERATION UNDER BOTH INTERRUPT AND NON-INTERRUPT MODES.
- 1.2 SYSTEM REQUIREMENTS
THIS PROGRAM IS DESIGNED TO RUN ON ANY PDP-11 WITH 4K OF MEMORY AND A KW11 LINE FREQUENCY CLOCK.
- 1.3 DEVICE ADDRESSES
THE DIAGNOSTIC ASSUMES THE LINE CLOCK ADDRESS IS 177546. VARIOUS TESTS IN THE DIAGNOSTIC ENSURE THAT THE LINE CLOCK IS NOT AFFECTED WHEN THE PAPER TAPE PUNCH (ADDRESS 177556) AND TELETYPE (ADDRESS 177566) ARE ACCESSED. THEREFORE, AN OCCASIONAL "2" MAY BE OUTPUT AT THE TELETYPE AND A CHARACTER MAY BE PUNCHED AT THE PAPER TAPE PUNCH. THESE ARE NOT ERRORS.

2.0 OPERATING INSTRUCTIONS

- 2.1 LOADING
PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED
A. ABSOLUTE LOADER PROGRAM MUST BE IN MEMORY
B. PLACE THE BINARY TAPE IN THE PAPER TAPE READER
C. LOAD ADDRESS 17500
D. DEPRESS START (TAPE SHOULD READ IN)
- 2.2 STARTING
PROGRAM STARTING ADDRESS IS 000200
A. LOAD ADDRESS 000200
B. SELECT SWITCH REGISTER OPTIONS (SEE SECTION 2.3)
- NOTE: IF THE PROCESSOR DOES NOT HAVE A HARDWARE SWITCH REGISTER, LOCATION # 176 SHOULD BE LOADED WITH THE APPROPRIATE SWITCH SETTINGS AFTER THE DIAGNOSTIC HAS BEEN LOADED INTO MEMORY. THE PROGRAM IS THEN STARTED AT ADDRESS 200.
- C. DEPRESS START (PROGRAM SHOULD START RUNNING)
- 2.3 SWITCH REGISTER OPTIONS
HERE IS A LIST OF CONSOLE SWITCHES AND THEIR EFFECT ON THE PROGRAM...
- | SWITCH | ACTION IF SET |
|--------|-------------------------------------------|
| 15 | HALT ON ERROR |
| 14 | LOOP ON CURRENTLY EXECUTING TEST |
| 13 | INHIBIT ERROR PRINTOUTS |
| 12 | (UNUSED) |
| 11 | INHIBIT ITERATIONS |
| 10 | BELL ON ERROR |
| 9 | LOOP ON ERROR |
| 8 | LOOP ON TEST SPECIFIED IN SWR(7:0) |
| 7-0 | # OF TEST TO LOOP ON (ONLY WHEN SWRB = 1) |
- 2.4 EXECUTION TIMES

EXECUTION TIME FOR THIS PROGRAM IS DEPENDENT ON THE MODEL OF PDP-11 IT IS BEING RUN ON. FOR A PDP-11/40 ABOUT 5 SECONDS IS NECESSARY TO DO 1 PASS OF THE PROGRAM WITHOUT ITERATIONS.

3.0 ERROR INFORMATION

3.1 STANDARD ERROR REPORTING PROCEDURES
ERROR PRINTOUTS CONSIST OF FROM 4 TO 8 COLUMNS OF DATA, A DATA HEADER, AND POSSIBLY A SHORT ERROR MESSAGE DESCRIBING THE ERROR. FOR EXAMPLE...

CLOCK FAILED TO INTERRUPT
PC PS SP TEST# LKS
002262 000344 000764 000007 000300

THE FIRST 4 COLUMNS OF OF THE ERROR MESSAGE ALWAYS SHOW THE CONTENTS OF THE PC, PS, SP, AND THE TEST NUMBER. MORE COLUMNS OF DATA ARE ADDED WHERE THEY MIGHT BE RELEVANT TO A PARTICULAR ERROR.

3.2 UNEXPECTED TRAP ERROR REPORTING
AN UNEXPECTED TRAP TO ADDRESS 4 CAUSES THE FOLLOWING MESSAGE TO BE PRINTED OUT...

TRAPPED TO LOC 4 FROM LOCATION "XXXXXX"
RESTARTING PROGRAM

IN THE ACTUAL MESSAGE THE "XXXXXX" IS REPLACED BY THE PC ADDRESS PUSHED ONTO THE STACK WHEN THE UNEXPECTED TRAP OCCURS. THE PROGRAM THEN TRYS TO RESTART ITSELF DESPITE SWITCH REGISTER SETTINGS.

3.3 POWER FAIL
IF A POWER FAIL CONDITION IS DETECTED THE FOLLOWING MESSAGE IS PRINTED...

POWER

AFTER PRINTING OUT THE MESSAGE THE PROGRAM TRYS TO RESTART ITSELF.

5.0 DEVICE INFORMATION

5.1 GENERAL INFORMATION
THE LINE CLOCK INTERRUPT VECTOR ADDRESS IS 100
THE LINE CLOCK PRIORITY LEVEL IS BR6

5.2 REGISTERS

LINE CLOCK STATUS REGISTER (LKS) 777546

! ! ! ! ! ! ! ! ! ! ! ! ! ! ! !
! ! ! ! ! ! ! ! 7 6 ! ! ! ! ! ! ! !

BIT6 IF SET MONITOR=1 CAUSES AN INTERRUPT

E01

BIT7 MONITOR BIT. SET BY CLOCK, CLEARED BY USER

SEQ 0004

7.0 LISTING

BASIC DEFINITIONS

57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112

000000
000040
000100
000140
000200
000240
000300
000340

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

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

```

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

```

.*"SWITCH REGISTER" SWITCH DEFINITIONS

```

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

```

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)

```

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

```

BASIC DEFINITIONS

```

113 .EQUIV BIT08,BIT8
114 .EQUIV BIT07,BIT7
115 .EQUIV BIT06,BIT6
116 .EQUIV BIT05,BIT5
117 .EQUIV BIT04,BIT4
118 .EQUIV BIT03,BIT3
119 .EQUIV BIT02,BIT2
120 .EQUIV BIT01,BIT1
121 .EQUIV BIT00,BIT0

```

;*BASIC "CPU" TRAP VECTOR ADDRESSES

```

122 ERRVEC= 4 ; TIME OUT AND OTHER ERRORS
123 RESVEC= 10 ; RESERVED AND ILLEGAL INSTRUCTIONS
124 TBITVEC=14 ; "T" BIT
125 TRTVEC= 14 ; TRACE TRAP
126 BPTVEC= 14 ; BREAKPOINT TRAP (BPT)
127 IOTVEC= 20 ; INPUT/OUTPUT TRAP (IOT) **SCOPE**
128 PWRVEC= 24 ; POWER FAIL
129 EMTVEC= 30 ; EMULATOR TRAP (EMT) **ERROR**
130 TRAPVEC=34 ; "TRAP" TRAP
131 TKVEC= 60 ; TTY KEYBOARD VECTOR
132 TPVEC= 64 ; TTY PRINTER VECTOR
133 PIRQVEC=240 ; PROGRAM INTERRUPT REQUEST VECTOR
134 ; MISCELLANEOUS EQUATES
135 LKS=177546
136 NOP=240
137 BUF2=774
138 BUF1=776

```

.SBTTL TRAP CATCHER

```

142 . =0
143 ; *ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
144 ; *SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
145 ; *LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
146 . =174
147 DISPREG: .WORD 0 ; SOFTWARE DISPLAY REGISTER
148 SWREG: .WORD 0 ; SOFTWARE SWITCH REGISTER

```

.SBTTL STARTING ADDRESS(ES)

```

152 JMP @#KSTART ; JUMP TO STARTING ADDRESS OF PROGRAM
153 *****

```

.SBTTL ACT11 HOOKS

```

154 ; HOOKS REQUIRED BY ACT11
155 $SVPC= . ; SAVE PC
156 . =46 ; 1) SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
157 $ENDAD ;
158 . =52 ; 2) SET LOC.52 TO ZERO
159 .WORD 0 ;
160 .=$SVPC ; RESTORE PC
161
162
163

```



```

164 ;*****
165 .SBTTL COMMON TAGS
166
167 ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
168 ;*USED IN THE PROGRAM.
169
170
171 001100 .=1100
172 001100 SCMTAG: .WORD 0 ;: START OF COMMON TAGS
173 001100 SPASS: .WORD 0 ;: CONTAINS PASS COUNT
174 001102 $STNM: .BYTE 0 ;: CONTAINS THE TEST NUMBER
175 001103 SERFLG: .BYTE 0 ;: CONTAINS ERROR FLAG
176 001104 SICNT: .WORD 0 ;: CONTAINS SUBTEST ITERATION COUNT
177 001106 SLPADR: .WORD 0 ;: CONTAINS SCOPE LOOP ADDRESS
178 001110 SLPERR: .WORD 0 ;: CONTAINS SCOPE RETURN FOR ERRORS
179 001112 SERTTL: .WORD 0 ;: CONTAINS TOTAL ERRORS L-TECTED
180 001114 SITEMB: .BYTE 0 ;: CONTAINS ITEM CONTROL BYTE
181 001115 SERMAX: .BYTE 1 ;: CONTAINS MAX. ERRORS PER TEST
182 001116 SERRPC: .WORD 0 ;: CONTAINS PC OF LAST ERROR INSTRUCTION
183 001120 SGOADR: .WORD 0 ;: CONTAINS ADDRESS OF 'GOOD' DATA
184 001122 SBODADR: .WORD 0 ;: CONTAINS ADDRESS OF 'BAD' DATA
185 001124 SGO DAT: .WORD 0 ;: CONTAINS 'GOOD' DATA
186 001126 SBODAT: .WORD 0 ;: CONTAINS 'BAD' DATA
187 001130 .WORD 0 ;: RESERVED--NOT TO BE USED
188 001132 .WORD 0
189 001134 .WORD 0
190 001136 177570 SWR: .WORD DSWR ;: ADDRESS OF SWITCH REGISTER
191 001140 177570 DISPLAY: .WORD DDISP ;: ADDRESS OF DISPLAY REGISTER
192 001142 177560 $TKS: 177560 ;: TTY KBD STATUS
193 001144 177562 $TKB: 177562 ;: TTY KBD BUFFER
194 001146 177564 $TPS: 177564 ;: TTY PRINTER STATUS REG. ADDRESS
195 001150 177566 $TPB: 177566 ;: TTY PRINTER BUFFER REG. ADDRESS
196 001152 $NULL: .BYTE 0 ;: CONTAINS NULL CHARACTER FOR FILLS
197 001153 $FILLS: .BYTE 2 ;: CONTAINS # OF FILLER CHARACTERS REQUIRED
198 001154 $FILLC: .BYTE 12 ;: INSERT FILL CHARS. AFTER A "LINE FEED"
199 001155 $STPFLG: .BYTE 0 ;: "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
200 001156 $REGAD: .WORD 0 ;: CONTAINS THE ADDRESS FROM
201 ;: WHICH ($REG0) WAS OBTAINED
202 001160 $REG0: .WORD 0 ;: CONTAINS (($REGAD)+0)
203 001162 $REG1: .WORD 0 ;: CONTAINS (($REGAD)+2)
204 001164 $REG2: .WORD 0 ;: CONTAINS (($REGAD)+4)
205 001166 $REG3: .WORD 0 ;: CONTAINS (($REGAD)+6)
206 001170 $REG4: .WORD 0 ;: CONTAINS (($REGAD)+10)
207 001172 $REG5: .WORD 0 ;: CONTAINS (($REGAD)+12)
208 001174 $REG6: .WORD 0 ;: CONTAINS (($REGAD)+14)
209 001176 $REG7: .WORD 0 ;: CONTAINS (($REGAD)+16)
210 001200 $TMP0: .WORD 0 ;: USER DEFINED
211 001202 $TMP1: .WORD 0 ;: USER DEFINED
212 001204 $TMP2: .WORD 0 ;: USER DEFINED
213 001206 $TMP3: .WORD 0 ;: USER DEFINED
214 001210 $TMP4: .WORD 0 ;: USER DEFINED
215 001212 $TMP5: .WORD 0 ;: USER DEFINED
216 001214 $TMP6: .WORD 0 ;: USER DEFINED
217 001216 $TMP7: .WORD 0 ;: USER DEFINED
218 001220 $TIMES: 0 ;: MAX. NUMBER OF ITERATIONS
219 001222 $ESCAPE: 0 ;: ESCAPE ON ERROR ADDRESS

```

J01

MAINDEC-11-DZKWA-F LINE FREQUENCY CLOCK PROGRAM
DZKWA.F.P11 09-NOV-77 00:00 COMMON TAGS

MACY11 30(1046) 07-DEC-77 12:50 PAGE 6

SEQ 0009

220 001224 177607 000377
221 001230 077
222 001231 015
223 001232 000012

\$BELL: .ASCIZ <207><377><377> :: CODE FOR BELL
\$QUES: .ASCII /?/ :: QUESTION MARK
\$CRLF: .ASCII <15> :: CARRIAGE RETURN
\$LF: .ASCIZ <12> :: LINE FEED

2224
2225 001234 000000
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241 001236
2242 001236 010454
2243 001240 010535
2244 001242 010616
2245 001244 000000
2246
2247 001246 010634
2248 001250 010666
2249 001252 010740
2250 001254 000000
2251
2252 001256 010754
2253 001260 011047
2254 001262 011120
2255 001264 000000
2256
2257 001266 011134
2258 001270 011233
2259 001272 011370
2260 001274 000000
2261
2262 001276 011406
2263 001300 011457
2264 001302 011530
2265 001304 000000
2266
2267 001306 011544
2268 001310 011602
2269 001312 011654
2270 001314 000000
2271
2272 001316 011670
2273 001320 011757
2274 001322 012036
2275 001324 000000
2276
2277 001326 012054
2278 001330 012130
2279 001332 012214

WORD: 000000

.SBTTL ERROR POINTER TABLE

;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
;*LOCATION SITE#B. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF SITE#B 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:

EM1 ;;
DH1 ;;PC PS SP TEST# LKS LKS
DT1 ;;\$ERRPC,\$REG7,\$REG6,\$REG5,LKS,\$GDDAT S/B ;;
0
EM2 ;;"CLOCK FAILED TO INTERRUPT"
DH2 ;;PC PS SP TEST# LKS ""
DT2 ;;\$ERRPC,\$REG7,\$REG6,\$REG5,LKS
0
EM3 ;;"CLOCK INTERRUPTED WHEN THE PROCESSOR PRIORITY WAS TOO HIGH"
DH3 ;;PC PS SP TEST# LKS ""
DT3 ;;\$ERRPC,\$REG7,\$REG6,\$REG5,LKS
0
EM4 ;;"CLOCK GIVES UNEQUAL # OF PULSES OVER TWO EQUAL PERIODS OF TIME"
DH4 ;;PC PS SP TEST# 1ST 2ND
DT4 ;;\$ERRPC,\$REG7,\$REG6,\$REG5,\$REG1,\$REG0
0
EM5 ;;"LKS REGISTER RESPONDS TO ANOTHER ADDRESS"
DH5 ;;PC PS SP TEST# ADDRESS
DT5 ;;\$ERRPC,\$REG7,\$REG6,\$REG5,\$GDADR
0
EM6 ;;"A NO SACK TIMEOUT HAS OCCURED"
DH6 ;;PC PS SP TEST# LKS ""
DT6 ;;\$ERRPC,\$REG7,\$REG6,\$REG5,LKS
0
EM7 ;;"WRONG CONDITION CODES WERE PUT ONTO STACK BY INTERRUPT"
DH7 ;;PC PS SP TEST# CC CC
DT7 ;;\$ERRPC,\$REG7,\$REG6,\$REG5,BUF1,\$GDDAT
0
EM10 ;;"WRONG PC PUT ONTO THE STACK BY AN INTERRUPT"
DH10 ;;PC PS SP TEST# ""
DT10 ;;\$ERRPC,\$REG7,\$REG6,\$REG5,BUF2,\$GDDAT

ERROR POINTER TABLE

280	001334	000000			0		
281						EM11	:"TRAPPED TRYING TO ACCESS LKS REGISTER"
282	001336	012232				DH11	:"(PC) (PS) (SP) TEST#"
283	001340	012310				DT11	:\$ERRPC, \$REG7, \$REG6, \$REG5
284	001342	012346					
285	001344	000000			0		
286							
287							
288							
289							
290	001400	012706	001000			KSTART:	MOV #1000, SP ; INITIALIZE THE STACK SO WE CAN CALL THE TYPEOUT
291	001404	005046					CLR -(SP)
292	001406	013746	000034				MOV 34, -(SP) ; SAVE CURRENT TRAP VECTOR
293	001412	012737	001422	000034			MOV #64\$, 34 ; SETUP NEW TRAP VECTOR
294	001420	104400					TRAP ; PUSH OLD PSW AN PC ON STACK
295	001422	016666	000002	000006	64\$:		MOV 2(SP), 6(SP) ;
296	001430	012716	001436				MOV #65\$, (SP) ; REPLACE OLD PC WITH NEW
297	001434	000002					RTI ; RESTORE PSW
298	001436	012637	000034		65\$:		MOV (SP)+, 34 ; RESTORE OLD TRAP VECTOR
299	001442	004737	006576				JSR PC, \$TYPE ; PRINTOUT STARTUP MESSAGE
300	001446	010306					STMS ; ADDRESS OF MESSAGE "MAINDEC-11-DZKWA-F"
301	001450					START:	
302	001450	012706	001100				MOV #SCMTAG, R6 ; FIRST LOCATION TO BE CLEARED
303	001454	005026					CLR (R6)+ ; CLEAR MEMORY LOCATION
304	001456	022706	001126				CMP #SBOOAT, R6 ; DONE?
305	001462	001374					BNE -6 ; LOOP BACK IF NO
306	001464	012706	001000				MOV #1000, SP ; SETUP THE STACK POINTER
307	001470	012737	006324	000020			MOV #SCOPE, @IOTVEC ; IOT VECTOR FOR SCOPE ROUTINE
308	001476	012737	000340	000022			MOV #340, @IOTVEC+2 ; LEVEL 7
309	001504	012737	007620	000030			MOV #ERROR, @EMTVEC ; EMT VECTOR FOR ERROR ROUTINE
310	001512	012737	000340	000032			MOV #340, @EMTVEC+2 ; LEVEL 7
311	001520	012737	010024	000034			MOV #TRAP, @TRAPVEC ; TRAP VECTOR FOR TRAP CALLS
312	001526	012737	000340	000036			MOV #340, @TRAPVEC+2 ; LEVEL 7
313	001534	012737	010060	000024			MOV #SPWRON, @PWAVEC ; POWER FAILURE VECTOR
314	001542	012737	000340	000026			MOV #340, @PWAVEC+2 ; LEVEL 7
315	001550	013737	006164	006156			MOV #ENDCT, SEOPCT ; SETUP END-OF-PROGRAM COUNTER
316	001556	005037	001220				CLR \$TIMES ; INITIALIZE NUMBER OF ITERATIONS
317	001562	005037	001222				CLR \$ESCAPE ; CLEAR THE ESCAPE ON ERROR ADDRESS
318	001566	112737	000001	001115			MOV #1, \$ERMAX ; ALLOW ONE ERROR PER TEST
319	001574	012737	001574	001106			MOV #, \$LPADR ; INITIALIZE THE LOOP ADDRESS FOR SCOPE
320	001602	012737	001602	001110			MOV #, \$LPERR ; SETUP THE ERROR LOOP ADDRESS
321	001610	013746	000004				MOV 2#4, -(SP) ; SAVE ERROR VECTOR
322	001614	013746	000006				MOV 2#6, -(SP)
323	001620	012737	001634	000004			MOV #64\$, 4 ; SET UP TIME OUT VECTOR
324	001626	005777	177304				TST \$SWR ; TRY TO REFERENCE HARDWARE SWR
325	001632	000407					BR 65\$; BRANCH IF NO TIMEOUT TRAP OCCURS
326	001634	012737	000176	001136	64\$:		MOV #SWREG, SWR ; POINT TO SOFTWARE SWR
327	001642	012737	000174	001140			MOV #DISPREG, DISPLAY ; POINT TO SOFTWARE DISPLAY REG
328	001650	022626					CMP (SP)+, (SP)+ ; RESTORE STACK
329	001652	012637	000006		65\$:		MOV (SP)+, 2#6 ; RESTORE ERROR VECTOR
330	001656	012637	000004				MOV (SP)+, 2#4
331	001662	005737	000042				TST 42 ; LOADED BY A MONITOR
332	001666	001401					BEQ T0001 ; BR IF NO
333	001670	000005					RESET ; YES--GENERATE AN INIT
334							
335							

```

336
337
338
339 001672 000004
340 001674 012737 001730 000004
341 001702 012737 000340 000006
342 001710 012737 001716 001106
343 001716 012706 001000
344 001722 005037 177546
345 001726 000401
346 001730 104011
347
348
349
350
351
352 001732 000004
353 001734 012737 006252 000004
354 001742 012737 000340 000006
355 001750 012737 001756 001106
356 001756 000005
357 001760 012737 000200 001124
358 001766 032737 000100 177546
359 001774 001401
360 001776 104001
361
362
363
364
365
366 002000 000004
367 002002 012737 006252 000004
368 002010 012737 000340 000006
369 002016 012737 000200 001124
370 002024 012737 002032 001106
371 002032 000005
372 002034 105737 177546
373 002040 100401
374 002042 104001
375
376
377
378
379
380 002044 000004
381 002046 012737 006252 000004
382 002054 012737 000340 000006
383 002062 012737 002070 001106
384 002070 012737 000200 001124
385 002076 005037 177546
386 002102 005000
387 002104 105737 177546
388 002110 100403
389 002112 005200
390 002114 001373
391 002116 104001

.SBTTL TEST THAT THE LKS CAN BE REFERENCED WITHOUT A BUS ERROR
:LKS ACCESS TEST
T0001: SCOPE
      MOV      #E0001, R#4      ;PREPARE FOR ADDRESSING THE LKS REGISTER. BAD HARDWARE
      MOV      #340, R#6      ;COULD CAUSE A TRAP TO 4
      MOV      #R0001, $LPADR  ;TIGHTEN UP THE SCOPE LOOP A BIT IN CASE OF AN ERROR
R0001: MOV      #1000, SP      ;SETUP THE STACK POINTER IN CASE OF AN ERROR
I0001: CLR      @LKS          ;JUST REFERENCE LKS. DONT WORRY IF IT DIDNT CLEAR YET
      BR       T0002          ;WE DIDNT TRAP IF WE REACH HERE. GO ON TO NEXT TEST
E0001: ERROR 11             ;ERROR:::TRAPED TRYING TO ACCESS THE LKS REGISTER

.SBTTL TEST THAT START CLEARS LINE CLOCK INTERRUPT ENABLE BIT
:TEST THAT START CLEARS LINE CLOCK INTERRUPT ENABLE BIT
T0002: SCOPE
      MOV      #TRAP0, R#4     ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
      MOV      #340, R#6     ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
      MOV      #R0002, $LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
R0002: RESET
      MOV      #200, $GDDAT    ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
      BIT      #100, LKS      ;TEST THE INTERRUPT ENABLE BIT
      BEQ      T0003
E0002: ERROR 1             ;ERROR, CLOCK INTERRUPT ENABLE NOT CLEARED BY INIT

.SBTTL TEST THAT START SETS CLOCK FLAG
:TEST THAT START SETS CLOCK FLAG
T0003: SCOPE
      MOV      #TRAP0, R#4     ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
      MOV      #340, R#6     ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
      MOV      #200, $GDDAT    ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
      MOV      #R0003, $LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
R0003: RESET
      TSTB     LKS            ;SHOULD SET THE CLOCK FLAG
      BMI     T0004          ;FIND OUT IF IT DID
      GO ON TO THE NEXT TEST IF IT SET THE CLOCK FLAG
E0003: ERROR 1             ;ERROR, CLOCK FLAG NOT SET BY INIT

.SBTTL TEST THAT CLOCK FLAG WILL SET AFTER SUFFICIENT PERIOD OF TIME (20 MS MIN)
:TEST THAT CLOCK FLAG WILL SET AFTER SUFFICIENT PERIOD OF TIME (20 MS MIN)
T0004: SCOPE
      MOV      #TRAP0, R#4     ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
      MOV      #340, R#6     ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
      MOV      #R0004, $LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
R0004: MOV      #200, $GDDAT    ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
      CLR      LKS            ;CLEAR THE CLOCK FLAG
      CLR      R0             ;AND A TIMER LOCATION
A0004: TSTB     LKS            ;IS CLOCK FLAG SET
      BMI     T0005
      INC      R0             ;NO, INCREMENT COUNT 003 WAIT FOR SOMEMORE
      BNE     A0004          ;WAIT SUFFICIENT AMOUNT OF TIME FOR CLOCK
E0004: ERROR 1             ;ERROR, CLOCK FLAG FAILED TO SET

```

NO1

MAINDEC-11-DZKWA-F LINE FREQUENCY CLOCK PROGRAM
 DZKWAR.P11 09-NOV-77 00:00

MACY11 30(1046) 07-DEC-77 12:50 PAGE 10
 TEST THAT CLOCK FLAG WILL SET AFTER SUFFICIENT PERIOD OF TIME (20 MS MIN)

SEQ 0013

```

392
393
394
395      .SBTTL TEST THAT INTERRUPT ENABLE BIT MAY BE SET
396      :TEST THAT INTERRUPT ENABLE BIT MAY BE SET
397      T0005: SCOPE
398      002120 000004          MOV      #TRAP0,2#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
399      002122 012737 006252 000004      MOV      #340,2#6      ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
400      002130 012737 000340 000006      MOV      #0,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
401      002136 012737 000100 001124      MOV      #R0005,$LPAOR ;SETUP LOOP BACK ADDRESS IN CASE OF ERROR
402      002144 012737 002164 001106      MOV      PR7,-(SP)     ;PUT NEW PS ON STACK
403      002152 013746 000340          MOV      #64$,-(SP)   ;PUT NEW PC ON STACK
404      002156 012746 002164          RTI                    ;POP NEW PC AND PS
405      002162 000002
406      002164 000002
407      002164 005037 177546      64$:
408      002170 005003          R0005: CLR      LKS
409      002172 105737 177546      CLR      R3           ;INITIALIZE A COUNTER LOCATION
410      002176 100404          TSTB     LKS          ;IS THE CLOCK FLAG SET?
411      002200 005203          BMI     B0005        ;IF SO, CONTINUE ON WITH THE TEST
412      002202 001373          INC     R3           ;IF NOT INCREMENT THE COUNTER LOCATION
413      002204 104001          BNE     A0005        ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
414      002206 000410          E0005: ERROR 1      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
415      002210          BR      T0006
416      002210 012737 000100 177546      B0005: MOV      #100,LKS ;CLEAR CLOCK FLAG AND SET INTERRUPT ENABLE
417      002216 032737 000100 177546      BIT     #100,LKS     ;IS INTERRUPT ENABLE SET?
418      002224 0010C1          BNE     T0006
419      002226 104001          E1005: ERROR 1      ;ERROR INTERRUPT ENABLE NOT SET
420
421
422      .SBTTL TEST THAT INTERRUPT ENABLE BIT MAY BE CLEARED
423      :TEST THAT INTERRUPT ENABLE BIT MAY BE CLEARED
424      T0006: SCOPE
425      002230 000004          MOV      #TRAP0,2#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
426      002232 012737 006252 000004      MOV      #340,2#6      ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
427      002240 012737 000340 000006      MOV      #0,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
428      002246 012737 000000 001124      MOV      #R0006,$LPAOR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
429      002254 012737 002274 001106      MOV      PR7,-(SP)     ;PUT NEW PS ON STACK
430      002262 013746 000340          MOV      #64$,-(SP)   ;PUT NEW PC ON STACK
431      002266 012746 002274          RTI                    ;POP NEW PC AND PS
432      002272 000002
433      002274 005037 177546      64$:
434      002300 005003          R0006: CLR      LKS
435      002302 105737 177546      CLR      R3           ;INITIALIZE A COUNTER LOCATION
436      002306 100404          TSTB     LKS          ;IS THE CLOCK FLAG SET?
437      002310 005203          BMI     B0006        ;IF SO, CONTINUE ON WITH THE TEST
438      002312 001373          INC     R3           ;IF NOT INCREMENT THE COUNTER LOCATION
439      002314 104001          BNE     A0006        ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
440      002316 000412          E0006: ERROR 1      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
441      002320          BR      T0007
442      002320 012737 000100 177546      B0006: MOV      #100,LKS ;CLEAR CLOCK FLAG AND SET INTERRUPT ENABLE
443      002326 005037 177546      CLR      LKS         ;CLEAR INTERRUPT ENABLE
444      002332 032737 000100 177546      BIT     #100,LKS     ;TEST THE INTERRUPT ENABLE BIT
445      002340 001401          BEQ     T0007        ;IS INTERRUPT ENABLE CLEARED
446      002342 104001          E10006: ERROR 1    ;ERROR, ERROR INTERRUPT BIT CAN NOT BE CLEARED
447

```

```

448
449
450 .SBTTL TEST THAT CLOCK INTERRUPTS TO CORRECT VECTOR ADDRESS
451 :TEST THAT CLOCK INTERRUPTS TO CORRECT VECTOR ADDRESS
452 002344 000004
453 002346 012737 006252 000004
454 002354 012737 000340 000006
455 002362 012737 000300 001124
456 002370 012737 002412 001106
457 002376 012737 002474 000100
458 002404 012737 000340 000102
459 002412 012706 001000
460 002416 005046
461 002420 012746 002426
462 002424 000002
463 002426 005037 177546
464 002432 005003
465 002434 105737 177546
466 002440 100404
467 002442 005203
468 002444 001373
469 002446 104001
470 002450 000415
471 002452
472 002452 012737 000100 177546
473 002460 005000
474 002462 005200
475 002464 000240
476 002466 001375
477 002470 104002
478 002472 000404
479 002474 105737 177546
480 002500 100401
481 002502 104001
482
483
484
485 .SBTTL TEST THAT CLOCK WILL INTERRUPT WITH PROCESSOR AT PRIORITY 5
486 :TEST THAT CLOCK WILL INTERRUPT WITH PROCESSOR AT PRIORITY 5
487 :NOTE: IF THERE IS NO HARDWARE 'PSW' THIS TEST WILL BE SKIPPED
488 002504 000004
489 002506 012737 006252 000004
490 002514 012737 000340 000006
491 002522 012737 002544 001106
492 002530 012737 002654 000100
493 002536 012737 000340 000102
494 002544 005J37 177546
495 002550 013746 000004
496 002554 012737 002572 000004
497 002562 012737 000240 177776
498 002570 000404
499 002572 022626
500 002574 012637 000004
501 002600 000431
502 002602 012637 000004
503 002606 012706 001000

T0007: SCOPE
MOV #TRAP0,2#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
MOV #340,2#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
MOV #300,$GOODAT ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
MOV #R0007,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
MOV #00007,100 ;SET UP VECTOR RETURN POINTER
MOV #340,2#102 ;1 INTERRUPT IS ENOUGH
R0007: MOV #1000,SP ;GET STACK READY FOR INTERRUPTS
CLR -(SP) ;DROP CPU PRIORITY LEVEL
MOV #1$,-(SP) ;SET RETURN ADDRESS ON STACK FROM 'RTI'
RTI ;RETURN TO NEXT INSTRUCTION

1$: CLR LKS ;INITIALIZE A COUNTER LOCATION
CLR R3 ;IS THE CLOCK FLAG SET?
A0007: TSTB LKS ;IF SO, CONTINUE ON WITH THE TEST
BMI B0007 ;IF NOT INCREMENT THE COUNTER LOCAT'ON
INC R3 ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
BNE A0007 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
E0007: ERROR 1
BR T0010

B0007: MOV #100,LKS ;ENABLE INTERRUPT
CLR RO
C0007: INC RO
NOP ;STALL FOR TIME
BNE C0007 ;WAIT FOR INTERRUPT
E10007: ERROR 2
BR T0010
D0007: TSTB LKS ;ENTER HERE IF INTERRUPTED
BMI T0010
E20007: ERROR ;ERROR, INTERRUPT NOT CAUSED BY CLOCK

T0010: SCOPE
MOV #TRAP0,2#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
MOV #340,2#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
MOV #R0010,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
MOV #00010,100 ;SET UP VECTOR RETURN POINTER
MOV #340,2#102 ;NO INTERRUPTS ALLOWED AFTER THE FIRST ONE
R0010: CLR LKS
MOV 4, -(SP) ;SAVE BUS TIMEOUT VECTOR CONTENTS
MOV #1$,4 ;SET A SERVICE 'PC' IN CASE TIMEOUT OCCURS
MOV #PR5,PS ;DO WE HAVE A HARDWARE 'PSW'?
BR 2$ ;BRANCH IF YES
1$: CMP (SP)+,(SP)+ ;RESTORE STACK FROM TIMEOUT
MOV (SP)+,4 ;RESTORE BUS TIMEOUT VECTOR CONTENTS
BR T0011
2$: MOV (SP)+,4 ;RESTORE BUS TIMEOUT VECTOR CONTENTS
MOV #1000,SP ;INITIALIZE THE STACK POINTER

```

```

504 002612 005003          CLR      R3          ; INITIALIZE A COUNTER LOCATION
505 002614 105737 177546 A0010: TSTB   LKS          ; IS THE CLOCK FLAG SET?
506 002620 100404          BMI     B0010        ; IF SO, CONTINUE ON WITH THE TEST
507 002622 005203          INC     R3           ; IF NOT INCREMENT THE COUNTER LOCATION
508 002624 001373          BNE     A0010        ; AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
509 002626 104001          E0010: ERROR 1      ; CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
510 002630 000415          BR      T0011
511 002632 000415          B0010:
512 002634 012737 000100 177546 MOV     #100,LKS     ; ENABLE INTERRUPT
513 002640 005000          CLR     R0
514 002642 005200          C0010: INC     R0
515 002644 000240          NOP
516 002646 001375          BNE     C0010        ; STALL FOR SOME TIME
517 002650 104002          E10010: ERROR 2    ; WAIT FOR INTERRUPT
518 002652 000404          BR      T0011        ; ERROR, INTERRUPT FAILED TO OCCUR
519 002654 105737 177546 D0010: TSTB   LKS          ; ENTER HERE IF INTERRUPTED
520 002660 100401          BMI     T0011
521 002662 104001          E20010: ERROR 1    ; ERROR, INTERRUPT DID NOT CLEAR THE CLOCK FLAG
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
002664 000004          .SBTTL TEST THAT CLOCK WILL NOT INTERRUPT WITH PROCESSOR PRIORITY 6
002666 012737 006252 000004 ; TEST THAT CLOCK WILL NOT INTERRUPT WITH PROCESSOR PRIORITY 6
002674 012737 000340 000006 ; NOTE: IF THERE IS NO HARDWARE 'PSW' THIS TEST WILL BE SKIPPED
002702 012737 002710 001106 T0011: SCOPE
002710 005037 177546 MOV     #TRAPO,2#4   ; SETUP VECTOR IN CASE OF UNFORSEEN PROBLEMS
002714 005003          MOV     #340,2#6    ; NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
002716 105737 177546 MOV     #A0011,$LPADR ; SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
002722 100404          R0011: CLR     LKS
002724 005203          CLR     R3
002726 001373          A0011: TSTB   LKS          ; INITIALIZE A COUNTER LOCATION
002730 104001          BMI     B0011        ; IS THE CLOCK FLAG SET?
002732 000443          INC     R3           ; IF SO, CONTINUE ON WITH THE TEST
002734          BNE     A0011        ; IF NOT INCREMENT THE COUNTER LOCATION
002736          E0011: ERROR 1      ; AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
002738          BR      T0012      ; CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
002740          B0011:
002742          MOV     #1000,SP   ; INITIALIZE THE STACK POINTER
002744          MOV     4-(SP)    ; SAVE BUS TIMEOUT VECTOR CONTENTS
002746          MOV     #1$ 4     ; SET SERVICE 'PC' IN CASE TIMEOUT OCCURS
002748          MOV     #PR6,PS  ; DO WE HAVE A HARDWARE 'PSW'?
002750          BR      2$
002752          CMP     (SP)+,(SP)+ ; RESTORE STACK FROM BUS TIMEOUT
002754          MOV     (SP)+,4  ; RESTORE BUS TIMEOUT VECTOR CONTENTS
002756          BR      T0012
002758          MOV     (SP)+,4  ; RESTORE BUS TIMEOUT VECTOR CONTENTS
002760          MOV     #E1011,100 ; SET UP VECTOR RETURN
002762          MOV     #100,LKS ; ENABLE INTERRUPT
002764          CLR     R3
002766          T0012:
002768          C0011: TSTB   LKS          ; INITIALIZE A COUNTER LOCATION
002770          BMI     D0011        ; IS THE CLOCK FLAG SET?
002772          INC     R3           ; IF SO, CONTINUE ON WITH THE TEST
002774          BNE     C0011        ; IF NOT INCREMENT THE COUNTER LOCATION
002776          E1011: ERROR 1    ; AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
002778          BR      T0011    ; CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
002780          D0011:
003004 012737 000100 000100 MOV     #100,LKS
003012 005003          CLR     R3
003014 105737 177546 C0011: TSTB   LKS          ; INITIALIZE A COUNTER LOCATION
003020 100404          BMI     D0011        ; IS THE CLOCK FLAG SET?
003022 005203          INC     R3           ; IF SO, CONTINUE ON WITH THE TEST
003024 001373          BNE     C0011        ; IF NOT INCREMENT THE COUNTER LOCATION
003026 104001          E1011: ERROR 1      ; AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
003030 000715          BR      T0011        ; CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
003032          D0011:

```



```

560 003032 000240      NOP
561 003034 000240      NOP
562 003036 000401      BR
563 003040 104003      E20011: ERROR 3
564
565
566
567 .SBTTL TEST THAT RESET SETS CLOCK FLAG
568 :TEST THAT RESET SETS CLOCK FLAG
569 003042 000004      T0012: SCOPE
570 003044 012737 000200 001124      MOV #200,$GDDAT ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
571 003052 012737 006252 000004      MOV #TRAP0,2#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
572 003060 012737 000340 000006      MOV #340,2#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
573 003066 012737 003074 001106      MOV #R0012,$LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
574 003074 005037 177546      R0012: CLR LKS ;CLEAR CLOCK FLAG
575 003100 005003      CLR R3 ;INITIALIZE A COUNTER LOCATION
576 003102 105737 177546      A0012: TSTB LKS ;IS THE CLOCK FLAG SET?
577 003106 100404      BMI B0012 ;IF SO, CONTINUE ON WITH THE TEST
578 003110 005203      INC R3 ;IF NOT INCREMENT THE COUNTER LOCATION
579 003112 001373      BNE A0012 ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
580 003114 104001      E0012: ERROR 1 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
581 003116 000407      BR T0013
582 003120
583 003120 005037 177546      B0012: CLR LKS
584 003124 000005      RESET ;SHOULD SET CLOCK FLAG
585 003126 105737 177546      TSTB LKS
586 003132 100401      BMI T0013
587 003134 104001      E10012: ERROR 1 ;ERROR, RESET DIDN'T SET CLOCK FLAG
588
589
590
591 .SBTTL TEST LINE CLOCK REPEATABILITY
592 :TEST LINE CLOCK REPEATABILITY
593 :MAKE SURE THAT OVER TWO EQUAL PERIODS OF TIME
594 :THE CLOCK PUTS OUT THE SAME NUMBER OF PULSES
595 003136 000004      T0013: SCOPE
596 003140 005000      R0013: CLR R0 ;CLEAR 1ST TIME COUNT
597 003142 005001      CLR R1 ;CLEAR 1ST CLOCK COUNT
598 003144 013746 000340      MOV PR7,-(SP) ;PUT NEW PS ON STACK
599 003150 012746 003156      MOV #64$,-(SP) ;PUT NEW PC ON STACK
600 003154 000002      RTI ;POP NEW PC AND PS
601 003156
602 003156 012737 003156 001106      64$: R1013: MOV #R1013,$LPADR ;ERROR IN NEXT FEW INSTRUCTIONS CAUSES A SHORT SCOPE LO
603 003164 005037 177546      CLR LKS
604
605 003170 005003      CLR R3 ;SYNC ON CLOCK FLAG A COUPLE OF TIMES
606 003172 105737 177546      A0013: TSTB LKS ;INITIALIZE A COUNTER LOCATION
607 003176 100404      BMI B0013 ;IS THE CLOCK FLAG SET?
608 003200 005203      INC R3 ;IF SO, CONTINUE ON WITH THE TEST
609 003202 001373      BNE A0013 ;IF NOT INCREMENT THE COUNTER LOCATION
610 003204 104001      E0013: ERROR 1 ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
611 003206 000524      BR T0014 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
612 003210
613 003210 012737 003210 001106      B0013: MOV #R2013,$LPADR ;MAKE SCOPE LOOP SHORT IN CASE OF AN ERROR
614 003216 005037 177546      CLR LKS
615 003222 005003      CLR R3 ;INITIALIZE A COUNTER LOCATION

```

616	003224	105737	177546		C0013:	TSTB	LKS		: IS THE CLOCK FLAG SET?
617	003230	100404				BMI	D0013		: IF SO, CONTINUE ON WITH THE TEST
618	003232	005203				INC	R3		: IF NOT INCREMENT THE COUNTER LOCATION
619	003234	001373				BNE	C0013		: AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
620	003236	104001			E10013:	ERROR 1			: CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
621	003240	000507				BR	T0014		
622	003242				D0013:				
623	003242	005037	177546			CLR	LKS		
624	003246	032737	000200	177546	1\$:	BIT	#200,LKS		: IS THE CLOCK FLAG SET ?
625	003254	001374				BNE	1\$: NO, WAIT FOR IT
626	003256	005037	177546			CLR	LKS		
627	003262	105737	177546		F0013:	TSTB	LKS		: IS CLOCK FLAG SET
628	003266	100003				BPL	G0013		: NO
629	003270	005201				INC	R1		: +1 TO CLOCK COUNT
630	003272	005037	177546			CLR	LKS		: CLEAR CLOCK IF SET
631	003276	005200			G0013:	INC	R0		: +1 TO TIME COUNT
632	003300	001370				BNE	F0013		: REPEAT UNTIL R0=0
633	003302	005000				CLR	R0		: CLEAR 2ND TIME COUNT
634	003304	005002				CLR	R2		: CLEAR 2ND CLOCK COUNT
635	003306	012737	003306	001106	R3013:	MOV	#R3013,\$LPADR		: INSURE A SHORT SCOPE LOOP
636	003314	005037	177546			CLR	LKS		
637									: SYNC ON CLOCK FLAG TWICE
638	003320	005003				CLR	R3		: INITIALIZE A COUNTER LOCATION
639	003322	105737	177546		H0013:	TSTB	LKS		: IS THE CLOCK FLAG SET?
640	003326	100404				BMI	J0013		: IF SO, CONTINUE ON WITH THE TEST
641	003330	005203				INC	R3		: IF NOT INCREMENT THE COUNTER LOCATION
642	003332	001373				BNE	H0013		: AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
643	003334	104001			E20013:	ERROR 1			: CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
644	003336	000450				BR	T0014		
645	003340				J0013:				
646	003340	012737	003340	001106	R4013:	MOV	#R4013,\$LPADR		: INSURE A SHORT SCOPE LOOP
647	003346	005037	177546			CLR	LKS		
648	003352	005003				CLR	R3		: INITIALIZE A COUNTER LOCATION
649	003354	105737	177546		K0013:	TSTB	LKS		: IS THE CLOCK FLAG SET?
650	003360	100404				BMI	L0013		: IF SO, CONTINUE ON WITH THE TEST
651	003362	005203				INC	R3		: IF NOT INCREMENT THE COUNTER LOCATION
652	003364	001373				BNE	K0013		: AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
653	003366	104001			F3013:	ERROR 1			: CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
654	003370	000433				BR	T0014		
655	003372				L0013:				
656	003372	012737	003140	001106	MOV	#R0013,\$LPADR			: MUST LOOP BACK TO BEGINING OF TEST IF EEROR COMES NOW
657	003400	005037	177546			CLR	LKS		
658	003404	032737	000200	177546	1\$:	BIT	#200,LKS		: IS CLOCK FLAG SET ?
659	003412	001374				BNE	1\$: NO, WAIT
660	003414	005037	177546			CLR	LKS		
661	003420	105737	177546		M0013:	TSTB	LKS		: IS CLOCK FLAG SET
662	003424	100003				BPL	N0013		: NO
663	003426	005202				INC	R2		: +1 TO CLOCK COUNT
664	003430	005037	177546			CLR	LKS		: CLEAR CLOCK IF SET
665	003434	005200			N0013:	INC	R0		: +1 TO TIME COUNT
666	003436	001370				BNE	M0013		: REPEAT UNTIL R0=0
667	003440	020102				CMP	R1,R2		: IS 1ST CLOCK COUNT EQUAL TO 2ND CLOCK COUNT?
668	003442	001406				BEG	T0014		: YES
669	003444	010137	001162		E40013:	MOV	R1,\$REG1		: GET R1 READY FOR PRINTOUT
670	003450	010237	001164			MOV	R2,\$REG2		: GET R2 READY FOR PRINTOUT
671	003454	104004				ERROR 4			: ERROR, CLOCK FLAG OCCURRED DIFFERENT

672 003456 000240 NOP ;NUMBER OF TIMES IN EQUAL PERIODS

673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727

003460 000004
003462 005037 001124
003466 012737 003530 001106
003474 012737 157546 001120
003502 012737 003546 000004
003510 012737 000340 000006
003516 013746 000340
003522 012746 003530
003526 000002
003530
003530 012706 001000
003534 005037 177546
003540 012777 000100 175352
003546 032737 000100 177546
003554 001401
003556 104005
003560 005037 177546
003564 012737 003576 000004
003572 005077 175322

```
.SBTTL LINE CLOCK REGISTER ADDRESSING TEST
:LINE CLOCK REGISTER ADDRESSING TEST
:TEST THAT THE "LKS" REGISTER CAN NOT BE ADDRESSED AS ANYTHING BUT 177546
:SET A LOCATION THAT IS CLOSE(DIFFERS BY 1 BIT) TO THE LKS REGISTER
:TO 100. IF THE LKS ALSO CHANGES, THEN SIGNAL AN ERROR
↑0014: SCOPE
      CLR      $GDDAT
      MOV      #R0014,$LPAOR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
      MOV      #157546,$GDADR ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 13 CLEAR
      MOV      #A0014,4 ;SETUP VECTOR IN CASE IT IS NONEXISTANT
      MOV      #340,6
      MOV      PR7,-(SP) ;:PUT NEW PS ON STACK
      MOV      #64$,-(SP) ;:PUT NEW PC ON STACK
      RTI ;:POP NEW PC AND PS

64$:
R0014: MOV      #1000,SP ;SETUP THE STACK
      CLR      LKS
I0014: MOV      #100,$GDADR ;SET THE "CLOSE" ADDRESS TO = 100
A0014: BIT      #100,LKS ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
      BEQ      B0014
E0014: ERROR 5 ;IT AFFECTED "LKS" -- ERROR
B0014: CLR      LKS
      MOV      #T0015,4
      CLR      $GDADR ;CLEAR OUT THE "CLOSE" ADDRESS
```

```
.SBTTL LINE CLOCK REGISTER ADDRESSING TEST
:LINE CLOCK REGISTER ADDRESSING TEST
↑0015: SCOPE
      MOV      #A0015,4 ;SETUP VECTOR IN CASE IT IS NONEXISTANT
      MOV      #340,6
      CLR      $GDDAT
      MOV      #R0015,$LPAOR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
      MOV      #177146,$GDADR ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 8 CLEAR
      MOV      PR7,-(SP) ;:PUT NEW PS ON STACK
      MOV      #64$,-(SP) ;:PUT NEW PC ON STACK
      RTI ;:POP NEW PC AND PS

64$:
R0015: MOV      #1000,SP ;SETUP THE STACK
      CLR      LKS
I0015: MOV      #100,$GDADR ;SET THE "CLOSE" ADDRESS TO = 100
A0015: BIT      #100,LKS ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
      BEQ      B0015
E0015: ERROR 5 ;IT AFFECTED "LKS" -- ERROR
B0015: CLR      LKS
      MOV      #T0016,4
      CLR      $GDADR ;CLEAR OUT THE "CLOSE" ADDRESS
```

.SBTTL LINE CLOCK REGISTER ADDRESSING TEST

```

728 ;LINE CLOCK REGISTER ADDRESSING TEST
729 003714 000004 000004 000004 000004 000004
730 003716 012737 004002 000004 000004
731 003724 012737 000340 000006
732 003732 005037 001124
733 003736 012737 003764 001106
734 003744 012737 177446 001120
735 003752 013746 000340
736 003756 012746 003764
737 003762 000002
738 003764
739 003764 012706 001000
740 003770 005037 177546
741 003774 012777 000100 175116
742 004002 032737 000100 177546
743 004010 001401
744 004012 104005
745 004014 005037 177546
746 004020 012737 004032 000004
747 004026 005077 175066
748
749
750
751 .SBTTL LINE CLOCK REGISTER ADDRESSING TEST
752 ;LINE CLOCK REGISTER ADDRESSING TEST
753 004032 000004
754 004034 012737 004120 000004
755 004042 012737 000340 000006
756 004050 005037 001124
757 004054 012737 004102 001106
758 004062 012737 177556 001120
759 004070 013746 000340
760 004074 012746 004102
761 004100 000002
762 004102
763 004102 012706 001000
764 004106 005037 177546
765 004112 012777 000100 175000
766 004120 032737 000100 177546
767 004126 001401
768 004130 104005
769 004132 005037 177546
770 004136 012737 004150 000004
771 004144 005077 174750
772
773
774
775 .SBTTL LINE CLOCK REGISTER ADDRESSING TEST
776 ;LINE CLOCK REGISTER ADDRESSING TEST
777 004150 000004
778 004152 012737 004236 000004
779 004160 012737 000340 000006
780 004166 005037 001124
781 004172 012737 004220 001106
782 004200 012737 177566 001120
783 004206 013746 000340

```

```

0016: SCOPE
MOV #A0016,4 ;SETUP VECTOR IN CASE IT IS NONEXISTANT
MOV #340,6
CLR $GDDAT
MOV #R0016,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
MOV #177446,$GDADR ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 6 CLEAR
MOV PR7,-(SP) ;PUT NEW PS ON STACK
MOV #64$,-(SP) ;PUT NEW PC ON STACK
RTI ;POP NEW PC AND PS

```

```

64$:
R0016: MOV #1000,SP ;SETUP THE STACK
CLR LKS
I0016: MOV #100,$SGDADR ;SET THE "CLOSE" ADDRESS TO = 100
A0016: BIT #100,LKS ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
BEQ B0016
E0016: ERROR 5 ;IT AFFECTED "LKS" -- ERROR
B0016: CLR LKS
MOV #T0017,4
CLR $SGDADR ;CLEAR OUT THE "CLOSE" ADDRESS

```

```

0017: SCOPE
MOV #A0017,4 ;SETUP VECTOR IN CASE IT IS NONEXISTANT
MOV #340,6
CLR $GDDAT
MOV #R0017,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
MOV #177556,$GDADR ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 3 SET
MOV PR7,-(SP) ;PUT NEW PS ON STACK
MOV #64$,-(SP) ;PUT NEW PC ON STACK
RTI ;POP NEW PC AND PS

```

```

64$:
R0017: MOV #1000,SP ;SETUP THE STACK
CLR LKS
I0017: MOV #100,$SGDADR ;SET THE "CLOSE" ADDRESS TO = 100
A0017: BIT #100,LKS ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
BEQ B0017
E0017: ERROR 5 ;IT AFFECTED "LKS" -- ERROR
B0017: CLR LKS
MOV #T0020,4
CLR $SGDADR ;CLEAR OUT THE "CLOSE" ADDRESS

```

```

0020: SCOPE
MOV #A0020,4 ;SETUP VECTOR IN CASE IT IS NONEXISTANT
MOV #340,6
CLR $GDDAT
MOV #R0020,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
MOV #177566,$GDADR ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 4 SET
MOV PR7,-(SP) ;PUT NEW PS ON STACK

```

DZKWA.F.P11 09-NOV-77 00:00

LINE CLOCK REGISTER ADDRESSING TEST

```

784 004212 012746 004220      MOV      #64$,-(SP)      ;;PUT NEW PC ON STACK
785 004216 000002      RTI                    ;; POP NEW PC AND PS
786 004220      64$:
787 004220 012706 001000      R0020:  MOV      #1000,SP      ;SETUP THE STACK
788 004224 005037 177546      CLR      LKS
789 004230 012777 000100 174662  I0020:  MOV      #100,JSGDADR      ;SET THE "CLOSE" ADDRESS TO = 100
790 004236 032737 000100 177546  A0020:  BIT      #100,LKS        ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
791 004244 001401      BEQ      B0020
792 004246 104005      E0020:  ERROR 5      ;IT AFFECTED "LKS" -- ERROR
793 004250 005037 177546      B0020:  CLR      LKS
794 004254 012737 004266 000004  MOV      #T0021 ,4
795 004262 005077 174632      CLR      JSGDADR      ;CLEAR OUT THE "CLOSE" ADDRESS

```

.SBTTL LINE CLOCK REGISTER ADDRESSING TEST

:LINE CLOCK REGISTER ADDRESSING TEST

```

800      T0021:  SCOPE
801 004266 000004      MOV      #A0021,4      ;SETUP VECTOR IN CASE IT IS NONEXISTANT
802 004270 012737 004354 000004  MOV      #340,6
803 004276 012737 000340 000006  CLR      $GDADR
804 004304 005037 001124      MOV      #R0021,$LPADR      ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
805 004310 012737 004336 001106  MOV      #177746,$GDADR      ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 7 SET
806 004316 012737 177746 001120  MOV      PR7 -(SP)
807 004324 013746 000340      MOV      #64$,-(SP)      ;PUT NEW PS ON STACK
808 004330 012746 004336      RTI                    ;PUT NEW PC ON STACK
809 004334 000002      ; POP NEW PC AND PS
810 004336      64$:
811 004336 012706 001000      R0021:  MOV      #1000,SP      ;SETUP THE STACK
812 004342 005037 177546      CLR      LKS
813 004346 012777 000100 174544  I0021:  MOV      #100,JSGDADR      ;SET THE "CLOSE" ADDRESS TO = 100
814 004354 032737 000100 177546  A0021:  BIT      #100,LKS        ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
815 004362 001401      BEQ      B0021
816 004364 104005      E0021:  ERROR 5      ;IT AFFECTED "LKS" -- ERROR
817 004366 005037 177546      B0021:  CLR      LKS
818 004372 012737 004404 000004  MOV      #T0022 ,4
819 004400 005077 174514      CLR      JSGDADR      ;CLEAR OUT THE "CLOSE" ADDRESS

```

.SBTTL LINE CLOCK REGISTER HIGH BYTE TEST

:LINE CLOCK REGISTER HIGH BYTE TEST

:MAKE SURE THE LKS REGISTER LOW BYTE RESPONDS TO THE HIGH BYTES ADDRESS

```

820      T0022:  SCOPE
821 004404 000004      JMP      T0023
822 004406 000137 004520      MOV      #TRAP0,2#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
823 004412 012737 006252 000004  MOV      #340,2#6      ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
824 004420 012737 000340 000006  MOV      #100,$GDADR
825 004426 012737 000100 001124  MOV      #T0022,$LPADR      ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
826 004434 012737 004462 001106  MOV      #177547,$GDADR      ;HIGH BYTE OF THE LKS REGISTER
827 004442 012737 177547 001120  MOV      PR7 -(SP)
828 004450 013746 000340      MOV      #64$,-(SP)      ;PUT NEW PS ON STACK
829 004460 000002      RTI                    ;PUT NEW PC ON STACK
830 004462 000002      ; POP NEW PC AND PS
831 004462      64$:
832 004462 012706 001000      R0022:  MOV      #1000,SP      ;SETUP THE STACK
833 004466 005037 177546      CLR      LKS
834 004472 112777 000100 174420  I0022:  MOVB   #100,JSGDADR      ;SET THE HIGH BYTE ADDRESS TO = 100

```

LINE CLOCK REGISTER HIGH BYTE TEST

```

840 004500 032737 000100 177546 BIT #100,LKS ;MAKE SURE THAT "LKS" LOW BYTE WAS AFFECTED
841 004506 001001 BNE A0022
842 004510 104005 E0022: ERROR 5 ;SHOULD HAVE SET BIT 7. ERROR
843 004512 005037 177546 A0022: CLR LKS
844 004516 000005 RESET
845
846
847
848 .SBTTL CLOCK FLAG BIT TEST
849 :CLOCK FLAG BIT TEST
850 004520 000004 T0023: SCOPE
851 004522 012737 006252 000004 MOV #TRAP0,2#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
852 004530 012737 000340 000006 MOV #340,2#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
853 004536 012737 000200 001124 MOV #200,$GDDAT
854 004544 012737 004552 001106 MOV #R0023,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
855 004552 005037 177546 R0023: CLR LKS
856 004556 005003 CLR R3 ;INITIALIZE A COUNTER LOCATION
857 004560 105737 177546 A0023: TSTB LKS ;IS THE CLOCK FLAG SET?
858 004564 100404 BMI B0023 ;IF SO CONTINUE ON WITH THE TEST
859 004566 005203 INC R3 ;IF NOT INCREMENT THE COUNTER LOCATION
860 004570 001373 BNE A0023 ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
861 004572 104001 E0023: ERROR 1 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
862 004574 000410 BR T0024
863 004576
864 004576 012737 000200 177546 T0023: MOV #200,LKS ;MOVE A 1 INTO THE CLOCK FLAG BIT
865 004604 023737 177546 001124 CMP LKS,$GDDAT ;SHOULD NOT AFFECT THE FLAG BIT
866 004612 001401 BEQ T0024
867 004614 104001 E10023: ERROR 1 ;CLOCK FLAG DID NOT CLEAR
868
869
870
871 .SBTTL INTERRUPT TEST
872 004616 000004 T0024: SCOPE
873 004620 012737 006252 000004 MOV #TRAP0,2#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
874 004626 012737 000340 000006 MOV #340,2#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
875 004634 012737 000200 001124 MOV #200,$GDDAT ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
876 004642 012737 004662 001106 MOV #R0024,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
877 004650 012737 004722 000100 MOV #E0024,100
878 004656 005037 177546 CLR LKS ;ALLOW CLOCK INTERRUPTS
879 004662 R0024:
880 004662 013746 000340 MOV PR7,-(SP) ;;PUT NEW PS ON STACK
881 004666 012746 004674 MOV #64$,-(SP) ;;PUT NEW PC ON STACK
882 004672 000002 RTI ;;POP NEW PC AND PS
883 004674
884 004674 012737 000300 177546 64$: MOV #300,LKS
885 004702 005227 000000 A0024: INC #0 ;WAIT FOR 20+ MS
886 004706 001375 BNE A0024 ;LOOP BACK IF NOT DONE WAITING
887 004710 000005 RESET ;RESET SHOULD CLEAR INTERRUPT ENABLE
888 004712 023737 001124 177546 CMP $GDDAT,LKS ;AND LEAVE THE CLOCK FLAG SET
889 004720 001401 BEQ T0025 ;GO ON TO THE NEXT TEST IF IT DID
890 004722 104001 E0024: ERROR 1 ;RESET SET INTERRUPT ENABLE OR CLEARED CLOCK FLAG
891
892
893
894 .SBTTL NO SACK TIMEOUT TEST
895 :NO SACK TIMEOUT TEST
    
```

```

896 004724 000004 T0025: SCOPE
897 004726 012737 006252 000004 MOV #TRAP0,2#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
898 004734 012737 000340 000006 MOV #340,2#E ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
899 004742 012737 000300 001124 MOV #300,$GDCAT ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
900 004750 012737 004772 001106 MOV #R0025,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
901 004756 012737 000340 000102 MOV #340,102 ;NO INTERRUPTS AFTER THE FIRST ONE
902 004764 012737 005054 000100 MOV #C0025,'00
903 004772 005037 177546 R0025: CLR LKS
904 004776 013746 000340 MOV PR7,-(SP) ;: PUT NEW PS ON STACK
905 005002 012746 005010 MOV #64$,-(SP) ;: PUT NEW PC ON STACK
906 005006 000002 RTI ;: POP NEW PC AND PS
907 005010
908 005010 005003 64$: CLR R3 ;: INITIALIZE A COUNTER LOCATION
909 005012 105737 177546 A0025: TSTB LKS ;: IS THE CLOCK FLAG SET?
910 005016 100404 BMI B0025 ;: IF SO, CONTINUE ON WITH THE TEST
911 005020 005203 INC R3 ;: IF NOT INCREMENT THE COUNTER LOCATION
912 005022 001373 BNE A0025 ;: AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
913 005024 104001 E0025: ERROR 1 ;: CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
914 005026 000417 BR T0026
915 005030
916 005030 005046 B0025: CLR -(SP) ;: DROP CPU PRIORITY LEVEL
917 005032 012746 005040 MOV #1$,-(SP) ;: SET RETURN 'PC' FROM THE RTI
918 005036 000002 RTI ;: RETURN TO NEXT INSTRUCTION
919 005040 012737 000100 177546 1$: MOV #100,LKS ;: ENABLE CLOCK INTERRUPTS
920 005046 000001 WAIT ;: THE ONLY WAY TO LEAVE HERE WITHOUT ERROR IS TO INTERRUPT
921 005050 104006 E10025: ERROR 6 ;: IF IT ERROR IS HERE ITS BECAUSE OF A "NO-SACK" TIMEOUT
922 005052 000405 BR T0026
923 005054 022737 000300 177546 C0025: CMP #300,LKS ;: FIND OUT WHAT THE INTERRUPT DID TO THE CLOCK STATUS REG
924 005062 001401 BEQ T0026 ;: IT CLEARED THE INTERRUPT ENABLE OR THE FLAG BIT
925 005064 104001 E20025: ERROR 1
926
927
928
929
930 .SBTTL RESET TEST
931 005066 000004 :RESET TEST
932 005070 012737 006252 000004 T0026: SCOPE
933 005076 012737 000340 000006 MOV #TRAP0,2#4 ;: SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
934 005104 012737 000200 001124 MOV #340,2#E ;: NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
935 005112 012737 005146 001106 MOV #200,$GDDAT ;: HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
936 005120 013746 000340 MOV #R0026,$LPADR ;: INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
937 005124 012746 005132 MOV PR7,-(SP) ;: PUT NEW PS ON STACK
938 005130 000002 MOV #64$,-(SP) ;: PUT NEW PC ON STACK
939 005132 RTI ;: POP NEW PC AND PS
940 005132 012737 005204 000100 64$: MOV #E0026,100
941 005140 012737 000140 000102 MOV #140,102 ;: SETUP STATUS FOR AFTER THE INTERRUPT
942 005146 005037 177546 R0026: CLR LKS
943 005152 012706 001000 MOV #1000,SP ;: SETUP THE STACK
944 005156 012737 000100 177546 MOV #100,LKS ;: SET INTERRUPT ENABLE BIT NOW
945 005164 005227 000000 A0026: INC #0 ;: WAIT FOR CLOCK FLAG TO SET
946 005170 001375 BNE A0026
947 005172 000005 I0026: RESET ;: RESET SHOULD NOT CLEAR THE FLAG
948 005174 023737 177546 001124 CMP LKS,$GDDAT ;: FIND OUT IF ID DIT DID OR NOT
949 005202 001401 BEQ T0027
950 005204 104001 E0026: ERROR 1 ;: RESET DID NOT INITIALIZE THE LKS WORD CORRECTLY
951

```

```

952
953
954      .SBTTL  CLOCK FLAG BIT TEST
955      :CLOCK FLAG BIT TEST
956      :MAKE SURE IT DOESNT CLEAR WHEN YOU TRY TO SET IT VIA A 'MOV' INSTRUCTION
957      T0027: SCOPE
958      005206 000004      MOV      #TRAP0,2#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
959      005210 012737 006252 000004      MOV      #340,2#6      ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
960      005216 012737 000340 000006      MOV      #200,$GDDAT    ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
961      005224 012737 000200 001124      MOV      #R0027,$LPADR  ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
962      005232 012737 005252 001106      MOV      #R0027,$LPADR
963      005240 005037 000102      CLR      102
964      005244 012737 005334 000100      MOV      #T0030,100
965      005252 005037 177546      R0027:  CLR      LKS
966      005256 013746 000340      MOV      PR7,-(SP)      ;:PUT NEW PS ON STACK
967      005262 012746 005270      MOV      #64$,-(CP)    ;:PUT NEW PC ON STACK
968      005266 000002      RTI      ;:POP NEW PC AND PS
969      005270 012706 001000      64$:   MOV      #1000,SP      ;:SETUP THE STACK
970      005274 005037 001234      CLR      WORD          ;:SETUP A COUNTER LOCATION TO = 0
971      005300 005237 001234      A0027: INC      WORD
972      005304 001375      BNE      A0027          ;:WASTE TIME LOOPING UNTIL THE COUNTER REACHES 0
973      005306 012737 000300 177546      MOV      #300,LKS      ;:SET INTERRUPT ENABLE AND TRY TO SET THE CLOCK FLAG
974      005314 012737 000200 177546      MOV      #200,LKS      ;:TRY TO SET IT AGAIN
975      005322 023737 177546 001124      CMP      LKS,$GDDAT    ;:DID THE CLOCK FLAG STAY SET?
976      005330 001401      BEQ      T0030         ;:IF NOT GO ON TO THE NEXT TEST
977      005332 104001      E0027: ERROR 1        ;:ERROR - MOVED A '1' INTO THE CLOCK FLAG BIT AND IT STAY
978
979
980
981      .SBTTL  CLOCK FLAG AFTER INTERRUPT TEST
982      :SEE IF AN INTERRUPT CLEARS THE CLOCK FLAG
983      T0030: SCOPE
984      005334 000004      MOV      #TRAP0,2#4      ;:SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
985      005336 012737 006252 000004      MOV      #340,2#6      ;:NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
986      005344 012737 000340 000006      CLR      LKS          ;:NO CLOCK INTERRUPTS BEFORE WE ARE READY
987      005352 012737 005037 177546      MOV      #100,$GDDAT    ;:HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
988      005356 012737 000100 001124      MOV      #R0030,$LPADR  ;:INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
989      005364 012737 005372 001106      MOV      #R0030,$LPADR
990      005372 012737 005440 000100      R0030: MOV      #A0030,100 ;:SETUP CLOCK INTERRUPT VECTOR
991      005400 005037 000102      CLR      102
992      005404 005046      CLR      -(SP)         ;:PRIORITY LEVEL WILL ALLOW FURTHER INTERRUPTS
993      005406 012746 005414      MOV      #1$,-(SP)     ;:DROP CPU PRIORITY LEVEL
994      005412 000002      RTI      ;:SET RETURN 'PC' FROM THE RTI
995      005414 012706 001000      1$:   MOV      #1000,SP    ;:RETURN TO NEXT INSTRUCTION
996      005420 005037 177546      CLR      LKS          ;:SETUP THE STACK
997      005424 105037 001234      CLR      WORD
998      005430 012737 000100 177546      MOV      #100,LKS      ;:CLEAR OUT A COUNTER LOCATION
999      005436 000001      WAIT     ;:ENABLE CLOCK INTERRUPTS NOW
1000     005440 012737 005472 000100      A0030: MOV      #E0030,100 ;:WAIT FOR AN INTERRUPT
1001     005446 005046      CLR      -(SP)         ;:ERROR IF WE INTERRUPT AGAIN
1002     005450 012746 005456      MOV      #B0030,-(SP)  ;:DROP CPU PRIORITY LEVEL
1003     005454 000002      RTI      ;:SET RETURN 'PC' FROM THE RTI
1004     005456 105237 001234      B0030: INCB   WORD     ;:RETURN TO NEXT INSTRUCTION
1005     005462 001375      BNE      B0030        ;:DO A NOTHING LOOP FOR A VERY SHORT PERIOD OF TIME
1006     005464 005037 177546      CLR      LKS          ;:WE SHOULD INCREMENT TO 0 LONG BEFORE AN INTERRUPT COMES
1007     005470 000401      BR       T0031
1007     005472 104001      E0030: ERROR 1        ;:INTERRUPT DID NOT CLEAR THE CLOCK FLAG

```



```

1008
1009
1010
1011      .SBTTL NO INTERRUPT AT PRIORITY 7 TEST
1012      :TEST THAT CLOCK WILL NOT INTERRUPT WITH PROCESSR AT PRIORITY 7
1013      T0031: SCOPE
1014      005474 000004          006252 000004      MOV      #TRAP0,2#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
1015      005476 012737          000340 000006      MOV      #340,2#6      ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
1016      005512 012737          005520 001106      MOV      #R0031,$LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
1017      005520 005037          177546      R0031:  CLR      LKS
1018      005524 005003          177546      A0031:  CLR      R3      ;INITIALIZE A COUNTER LOCATION
1019      005526 105737          177546      A0031:  TSTB   LKS      ;IS THE CLOCK FLAG SET?
1020      005532 100404          177546      A0031:  BMI    B0031      ;IF SO, CONTINUE ON WITH THE TEST
1021      005534 005203          177546      A0031:  INC    R3      ;IF NOT INCREMENT THE COUNTER LOCATION
1022      005536 001373          177546      A0031:  BNE   A0031      ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
1023      005540 104001          177546      E0031:  ERROR 1
1024      005542 000431          177546      E0031:  BR     T0032      ;CLOCK FLAG DID NOT SET F.. TER A WAITING PERIOD > 20 MS
1025      005544 000000          177546      B0031:
1026      005544 012706          001000      B0031:  MOV    #1000,SP      ;INITIALIZE THE STACK POINTER
1027      005546 013746          000340      B0031:  MOV    PR7,-(SP)    ;PUT NEW PS ON STACK
1028      005548 012746          005562      B0031:  MOV    #64$,-(SP)  ;PUT NEW PC ON STACK
1029      005550 000002          177546      B0031:  RTI    ;POP NEW PC AND PS
1030      005562 000000          177546      64$:
1031      005562 012737          005540 000100      MOV    #E0031,100    ;SET UP VECTOR RETURN
1032      005570 012737          000100 177546      MOV    #100,LKS      ;ENABLE INTERRUPT
1033      005576 005003          177546      C0031:  CLR    R3      ;INITIALIZE A COUNTER LOCATION
1034      005600 105737          177546      C0031:  TSTB  LKS      ;IS THE CLOCK FLAG SET?
1035      005604 100404          177546      C0031:  BMI   D0031      ;IF SO, CONTINUE ON WITH THE TEST
1036      005606 005203          177546      C0031:  INC   R3      ;IF NOT INCREMENT THE COUNTER LOCATION
1037      005610 001373          177546      C0031:  BNE  C0031      ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
1038      005612 104001          177546      E10031: ERROR 1
1039      005614 000404          177546      E10031: BR    T0032      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
1040      005616 000000          177546      D0031:
1041      005616 000240          177546      D0031:  NOP
1042      005620 000240          177546      D0031:  NOP      ;GIVE CLOCK EXTRA TIME TO INTERRUPT
1043      005622 000401          177546      E20031: BR    T0032
1044      005624 104003          177546      E20031: ERROR 3      ;ERROR, CLOCK INTERRUPTED WITHOUT HAVING PRIORITY
1045
1046
1047
1048      .SBTTL CC PUSH TEST FOR CLOCK INTERRUPTS
1049      :TEST THAT CLOCK INTERRUPT PUSHES CONDITION CODES ONTO STACK
1050      T0032: SCOPE
1051      005626 000004          006252 000004      MOV      #TRAP0,2#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
1052      005630 012737          000340 000006      MOV      #340,2#6      ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
1053      005636 012737          005652 001106      MOV      #R0032,$LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
1054      005644 005037          177546      R0032:  CLR      LKS
1055      005652 005003          177546      A0032:  CLR      R3      ;INITIALIZE A COUNTER LOCATION
1056      005656 105737          177546      A0032:  TSTB   LKS      ;IS THE CLOCK FLAG SET?
1057      005664 100404          177546      A0032:  BMI   B0032      ;IF SO, CONTINUE ON WITH THE TEST
1058      005666 005203          177546      A0032:  INC    R3      ;IF NOT INCREMENT THE COUNTER LOCATION
1059      005670 001373          177546      A0032:  BNE   A0032      ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
1060      005672 104001          177546      E0032:  ERROR 1
1061      005674 000433          177546      E0032:  BR     T0033      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
1062      005676 000000          177546      B0032:
1063      005676 012706          001000      B0032:  MOV    #1000,SP      ;INITIALIZE THE STACK POINTER

```

DZKWA.F.P11

09-NOV-77 00:00

CC PUSH TEST FOR CLOCK INTERRUPTS

```

1064 005702 005037 000776 CLR BUF1
1065 005706 005037 000774 CLR BUF2
1066 005712 012737 005742 000100 MOV #C0032,100 ;SET UP VECTOR RETURN
1067 005720 012737 000100 177546 MOV #100,LKS ;ENABLE INTERRUPT
1068 005726 005046 CLR -(SP) ;DROP CPU PRIORITY LEVEL
1069 005730 012746 005736 MOV #1$,-(SP) ;SET RETURN 'PC' FROM THE RTI
1070 005734 000002 RTI ;RETURN TO NEXT INSTRUCTION
1071 005736 000277 1$: +SEC!SEV!SEZ!SEN ;SET ALL CONDITION CODES
1072 005740 000001 WAIT ;WAIT FOR INTERRUPT
1073 005742 022737 000017 000776 C0032: CMP #17,BUF1
1074 005750 001405 BEQ T0033
1075 005752 012737 000017 001124 MOV #17,$GDOAT ;ERROR DID NOT PUSH CORRECT PS ONTO STACK
1076 005760 104007 ERROR 7
1077 005762 000721 BR T0032

1078
1079
1080
1081 .SBTTL PC PUSH TEST FOR CLOCK INTERRUPTS
1082 .TEST THAT CLOCK INTERRUPT PUSHES THE PROGRAM COUNTER ONTO STACK
1083 005764 000004 T0033: SCOPE
1084 005766 012737 006252 000004 MOV #TRAP0,2#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
1085 005774 012737 000340 000006 MOV #340,2#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
1086 006002 012737 006010 001106 MOV #R0033,$LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
1087 006010 005037 177546 R0033: CLR LKS
1088 006014 005003 A0033: CLR R3 ;INITIALIZE A COUNTER LOCATION
1089 006016 105737 177546 A0033: TSTB LKS ;IS THE CLOCK FLAG SET?
1090 006022 100404 BMI B0033 ;IF SO, CONTINUE ON WITH THE TEST
1091 006024 005203 INC R3 ;IF NOT INCREMENT THE COUNTER LOCATION
1092 006026 001373 BNE A0033 ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
1093 006030 104001 E0033: ERROR 1 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
1094 006032 000432 BR T0034
1095 006034 B0033:
1096 006034 012706 001000 MOV #1000,SP ;INITIALIZE THE STACK POINTER
1097 006040 005037 000776 CLR BUF1
1098 006044 005037 000774 CLR BUF2
1099 006050 012737 006100 000100 MOV #C0033,100 ;SET UP VECTOR RETURN
1100 006056 012737 000100 177546 MOV #100,LKS ;ENABLE INTERRUPT
1101 006064 005046 CLR -(SP) ;DROP CPU PRIORITY LEVEL
1102 006066 012746 006074 MOV #1$,-(SP) ;SET RETURN 'PC' FROM THE RTI
1103 006072 000002 RTI ;RETURN TO NEXT INSTRUCTION
1104 006074 000277 1$: +SEC!SEV!SEZ!SEN ;SET ALL CONDITION CODES
1105 006076 000001 WAIT ;WAIT FOR INTERRUPT
1106 006100 022737 006100 000774 C0033: CMP #C0033,BUF2
1107 006106 001404 BEQ T0034
1108 006110 012737 006100 001124 E10033: MOV #C0033,$GDOAT ;ERROR, DID NOT PUSH CORRECT PC ONTO STACK
1109 006116 104010 ERROR 10
1110
1111
1112
1113 .SBTTL END OF PASS INDICATING
1114 006120 000004 T0034: SCOPE
1115 006122 005037 177546 CLR LKS ;TURN THE CLOCK OFF
1116 006126 000005 RESET ;TURN EVERYTHING OFF
1117
1118 ;*****
1119

```

END OF PASS ROUTINE

```

1120
1121
1122
1123
1124
1125
1126
1127 006130
1128 006130 000004
1129 006132 005037 001102
1130 006136 005037 001220
1131 006142 005237 001100
1132 006146 042737 100000 001100
1133 006154 005327
1134 006156 000001
1135 006160 003022
1136 006162 012737
1137 006164 000001
1138 006166 006156
1139 006170 104400 006232
1140 006174 013746 001100
1141 006200 104404
1142 006202 104400 006247
1143 006206
1144
1145 006206 013700 000042
1146 006212 001405
1147 006214 000005
1148 006216 004710
1149 006220 000240
1150 006222 000240
1151 006224 000240
1152 006226
1153 006226 000137 001672
1154 006232 005015 047105 020104
1155 006240 040520 051523 021440
1156 006246 000
1157 006247 377 377 000
1158 006252 005046
1159 006254 004737 006576
1160 006260 010363
1161 006262 011646
1162 006264 162716 000002
1163 006270 104401
1164 006272 104400 001231
1165 006276 005046
1166 006300 004737 006576
1167 006304 010424
1168 006306 000240
1169 006310 000240
1170 006312 000240
1171 006314 000240
1172 006316 000005
1173 006320 000137 001450
1174
1175

```

```

.SBTTL END OF PASS ROUTINE
;*INCREMENT THE PASS NUMBER ($PASS)
;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
;*IF THERES A MONITOR GO TO IT
;*IF THERE ISN'T JUMP TO T0001

$EOP:
SCOPE
CLR $STNM ;; ZERO THE TEST NUMBER
CLR $TIMES ;; ZERO THE NUMBER OF ITERATIONS
INC $PASS ;; INCREMENT THE PASS NUMBER
BIC #100000,$PASS ;; DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;; LOOP?
$EOPCT: .WORD 1
BGT $DOAGN ;; YES
MOV (PC)+,(PC)+ ;; RESTORE COUNTER
$ENDCT: .WORD 1
$EOPCT
TYPE $ENOMG ;; TYPE "END PASS #"
MOV $PASS,-(SP) ;; SAVE $PASS FOR TYPEOUT
TYPDS $ENULL ;; GO TYPE--DECIMAL ASCII WITH SIGN
TYPE , $ENULL ;; TYPE A NULL CHARACTER

$GET42:
MOV #42,R0 ;; GET MONITOR ADDRESS
BEQ $DOAGN ;; BRANCH IF NO MONITOR
RESET ;; CLEAR THE WORLD
$ENDAD: JSR PC,(R0) ;; GO TO MONITOR
NOP ;; SAVE ROOM
NOP ;; FOR
NOP ;; ACT11

$DOAGN:
JMP #T0001 ;; RETURN
$ENOMG: .PSCIZ <15><12>/END PASS #/

$ENULL: .BYTE -1,-1,0 ;; NULL CHARACTER STRING
TRAPD: CLR -(SP)
JSR PC,$TYPE ;; PRINTOUT "TRAPPED TO 4 FROM "
TRPM2S ;; ADDRESS OF THE MESSAGE
MOV (SP),-(SP) ;; GET THE ADDRESS WHERE THE TRAP OCCURED
SIB #2,(SP) ;; MAKE IT RIGHT
TYPDC ;; TYPE OUT ADDRESS IN OCTAL
TYPE , $CRLF ;; PRINTOUT A CARRIAGE RETURN-LINE FEED
CLR -(SP)
JSR PC,$TYPE ;; PRINTOUT RESTARTING MESSAGE
TRPM2S ;; ADDRESS OF RESTART MESSAGE
NOP
NOP
NOP
NOP
RESET
JMP START
;*****

```

```

1176 .SBTTL SCOPE HANDLER ROUTINE
1177
1178 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
1179 ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
1180 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
1181 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1182 ;*SW14=1 LOOP ON TEST
1183 ;*SW11=1 INHIBIT ITERATIONS
1184 ;*SW09=1 LOOP ON ERROR
1185 ;*SW08=1 LOOP ON TEST IN SWR<7:0>
1186 ;*CALL
1187 ;*
1188 ;* SCOPE ; ;SCOPE=IOT
1189
1189 006324 $SCOPE:
1190 006324 000240 NOP
1191 006326 032777 040000 172602 1$: BIT @BIT14,@SWR ; ;LOOP ON PRESENT TEST?
1192 006334 001111 BNE $OVER ; ;YES IF SW14=1
1193 ;*****START OF CODE FOR THE XOR TESTER*****
1194 006336 000416 $XTSTR: BR 6$ ; ;IF RUNNING ON THE "XOR" TESTER CHANGE
1195 ; ;THIS INSTRUCTION TO A "NOP" (NOP=240)
1196 006340 013746 000004 MOV @ERRVEC, -(SP) ; ;SAVE THE CONTENTS OF THE ERROR VECTOR
1197 006344 012737 006364 000004 MOV @5$,@ERRVEC ; ;SET FOR TIMEOUT
1198 006352 005737 177060 TST @177060 ; ;TIME OUT ON XOR?
1199 006356 012637 000004 MOV (SP)+,@ERRVEC ; ;RESTORE THE ERROR VECTOR
1200 006362 000463 BR $SVLAD ; ;GO TO THE NEXT TEST
1201 006364 022626 5$: CMP (SP)+,(SP)+ ; ;CLEAR THE STACK AFTER A TIME OUT
1202 006366 012637 000004 MOV (SP)+,@ERRVEC ; ;RESTORE THE ERROR VECTOR
1203 006372 000423 BR 7$ ; ;LOOP ON THE PRESENT TEST
1204 006374 6$: ;*****END OF CODE FOR THE XOR TESTER*****
1205 006374 032777 000400 172534 BIT @BIT08,@SWR ; ;LOOP ON SPEC. TEST?
1206 006402 001404 BEQ 2$ ; ;BR IF NO
1207 006404 127737 172526 001102 CMPB @SWR,$TSTNM ; ;ON THE RIGHT TEST? SWR<7:0>
1208 006412 001464 BEQ $OVER ; ;BR IF YES
1209 006414 105737 001103 2$: TSTB $ERFLG ; ;HAS AN ERROR OCCURRED?
1210 006420 001421 BEQ 3$ ; ;BR IF NO
1211 006422 123737 001115 001103 CMPB $ERMAX,$ERFLG ; ;MAX. ERRORS FOR THIS TEST OCCURRED?
1212 006430 101015 BHI 3$ ; ;BR IF NO
1213 006432 032777 001000 172476 BIT @BIT09,@SWR ; ;LOOP ON ERROR?
1214 006440 001404 BEQ 4$ ; ;BR IF NO
1215 006442 013737 001110 001106 7$: MOV $LPERR,$LPADR ; ;SET LOOP ADDRESS TO LAST SCOPE
1216 006450 000443 BR $OVER
1217 006452 105037 001103 4$: CLRB $ERFLG ; ;ZERO THE ERROR FLAG
1218 006456 005037 001220 CLR $TIMES ; ;CLEAR THE NUMBER OF ITERATIONS TO MAKE
1219 006462 000415 BR 1$ ; ;ESCAPE TO THE NEXT TEST
1220 006464 032777 004000 172444 3$: BIT @BIT11,@SWR ; ;INHIBIT ITERATIONS?
1221 006472 001011 BNE 1$ ; ;BR IF YES
1222 006474 005737 001100 TST $PASS ; ;IF FIRST PASS OF PROGRAM
1223 006500 001406 BEQ 1$ ; ;INHIBIT ITERATIONS
1224 006502 005237 001104 INC $ICNT ; ;INCREMENT ITERATION COUNT
1225 006506 023737 001220 001104 CMP $TIMES,$ICNT ; ;CHECK THE NUMBER OF ITERATIONS MADE
1226 006514 002021 BGE $OVER ; ;BR IF MORE ITERATION REQUIRED
1227 006516 012737 000001 001104 1$: MOV @1,$ICNT ; ;REINITIALIZE THE ITERATION COUNTER
1228 006524 013737 006574 001220 MOV $MXCNT,$TIMES ; ;SET NUMBER OF ITERATIONS TO DO
1229 006532 105237 001102 $SVLAD: INCB $TSTNM ; ;COUNT TEST NUMBERS
1230 006536 011637 001106 MOV (SP),$LPADR ; ;SAVE SCOPE LOOP ADDRESS
1231 006542 011637 001110 MOV (SP),$LPERR ; ;SAVE ERROR LOOP ADDRESS
    
```

SCOPE HANDLER ROUTINE

```

1232 006546 005037 001222          CLR          $ESCAPE          ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
1233 006552 112737 000001 001115  MOVB         $1, $ERMAX       ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
1234 006560 013777 001102 172352 $OVER:  MOV      $STIM, $DISPLAY  ;; DISPLAY TEST NUMBER
1235 006566 013716 001106          MOV      $LPADR, (SP)       ;; FUDGE RETURN ADDRESS
1236 006572 000002          RTI                          ;; FIXES PS
1237 006574 000010          $MXCNT: 10                ;; MAX. NUMBER OF ITERATIONS
1238 ;*****
1239
1240 .SBTTL  TYPE ROUTINE
1241
1242 ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1243 ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1244 ;*NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1245 ;*NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1246 ;*NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1247 ;*
1248 ;*CALL:
1249 ;*1) USING A TRAP INSTRUCTION
1250 ;*
1251 ;*OR
1252 ;*
1253 ;*
1254 ;*
1255
1256 006576 105737 001155          $TYPE:  TSTB         $STPFLG      ;; IS THERE A TERMINAL?
1257 006602 100002          BPL          1$                ;; BR IF YES
1258 006604 000000          HALT                          ;; HALT HERE IF NO TERMINAL
1259 006606 000407          BR          3$                ;; LEAVE
1260 006610 010046          1$:  MOV          RO, -(SP)      ;; SAVE RO
1261 006612 017600 000002          MOV      $2(SP), RO          ;; GET ADDRESS OF ASCIZ STRING
1262 006616 112046          2$:  MOVB         (RO)+, -(SP)  ;; PUSH CHARACTER TO BE TYPED ONTO STACK
1263 006620 001005          BNE          4$                ;; BR IF IT ISN'T THE TERMINATOR
1264 006622 005726          TST         (SP)+            ;; IF TERMINATOR POP IT OFF THE STACK
1265 006624 012600          60$: MOV          (SP)+, RO      ;; RESTORE RO
1266 006626 062716 000002          3$:  ADD          #2, (SP)      ;; ADJUST RETURN PC
1267 006632 000002          RTI                          ;; RETURN
1268 006634 122716 000011          4$:  CMPB         #THT, (SP)    ;; BRANCH IF <HT>
1269 006640 001426          BEQ          8$                ;; BRANCH IF NOT <CRLF>
1270 006642 122716 000200          CMPB         #TCRLF, (SP)
1271 006646 001004          BNE          5$                ;; POP <CR><LF> EQUIV
1272 006650 005726          TST         (SP)+            ;; TYPE A CR AND LF
1273 006652 104400          TYPE
1274 006654 001231          $CRLF
1275 006656 000757          BR          2$                ;; GET NEXT CHARACTER
1276 006660 004737 006742          5$:  JSR          PC, $TYPEPC    ;; GO TYPE THIS CHARACTER
1277 006664 123726 001154          6$:  CMPB         $FILLC, (SP)+  ;; IS IT TIME FOR FILLER CHARS.?
1278 006670 001352          BNE          2$                ;; IF NO GO GET NEXT CHAR.
1279 006672 013746 001152          MOV      $NULL, -(SP)        ;; GET # OF FILLER CHARS. NEEDED
1280 ; AND THE NULL CHAR.
1281 006676 105366 000001          7$:  DECB         1(SP)          ;; DOES A NULL NEED TO BE TYPED?
1282 006702 002770          BLT          6$                ;; BR IF NO--GO POP THE NULL OFF OF STACK
1283 006704 004737 006742          JSR          PC, $TYPEPC    ;; GO TYPE A NULL
1284 006710 105337 007006          DECB         $CHARCNT        ;; DO NOT COUNT AS A COUNT
1285 006714 000770          BR          7$                ;; LOOP
1286
1287 ;HORIZONTAL TAB PROCESSOR

```

```

1288
1289 006716 112716 000040 8$: MOVB #40,(SP) ;; REPLACE TAB WITH SPACE
1290 006722 004737 006742 9$: JSR PC,$TYPEC ;; TYPE A SPACE
1291 006726 132737 000007 007006 BITB #7,$CHARCNT ;; BRANCH IF NOT AT
1292 006734 001372 BNE 9$ ;; TAB STOP
1293 006736 005726 TST (SP)+ ;; POP SPACE OFF STACK
1294 006740 000726 BR 2$ ;; GET NEXT CHARACTER
1295 006742 105777 172200 $TYPEC: TSTB $STPS ;; WAIT UNTIL PRINTER IS READY
1296 006746 100375 BPL $TYPEC
1297 006750 116677 000002 172172 MOVB 2(SP),$STPB ;; LOAD CHAR TO BE TYPED INTO DATA REG.
1298 006756 122766 000015 000002 CMPB #15,2(SP) ;; BRANCH IF
1299 006764 001003 BNE 1$ ;; NOT <CR>
1300 006766 105037 007006 CLRB $CHARCNT
1301 006772 000406 BR $TYPEX ;; EXIT
1302 006774 122766 000012 000002 1$: CMPB #12,2(SP) ;; BRANCH IF
1303 007002 002002 BGE $TYPEX ;; <LF>
1304 007004 105227 INCB (PC)+ ;; INC SPACE
1305 007006 000000 $CHARCNT: WORD 0 ;; COUNT
1306 007010 000207 $TYPEX: RTS PC
1307 EQUATES
1308 THT=11
1309 TCRLF=200
1310
1311 ;*****
1312
1313 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
1314
1315 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
1316 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
1317 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
1318 ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
1319 ;*REPLACED WITH SPACES.
1320 ;*CALL:
1321 ;* MOV NUM,-(SP) ;; PUT THE BINARY NUMBER ON THE STACK
1322 ;* TYPDS ;; GO TO THE ROUTINE
1323
1324 $TYPDS:
1325 007012 010046 MOV R0,-(SP) ;; PUSH R0 ON STACK
1326 007014 010146 MOV R1,-(SP) ;; PUSH R1 ON STACK
1327 007016 010246 MOV R2,-(SP) ;; PUSH R2 ON STACK
1328 007020 010346 MOV R3,-(SP) ;; PUSH R3 ON STACK
1329 007022 010546 MOV R5,-(SP) ;; PUSH R5 ON STACK
1330 007024 012746 020200 MOV #20200,-(SP) ;; SET BLANK SWITCH AND SIGN
1331 007030 016605 000020 MOV 20(SP),R5 ;; GET THE INPUT NUMBER
1332 007034 100004 BPL 1$ ;; BR IF INPUT IS POS.
1333 007036 005405 NEG R5 ;; MAKE THE BINARY NUMBER POS.
1334 007040 112766 000055 000001 1$: MOVB #'-',1(SP) ;; MAKE THE ASCII NUMBER NEG.
1335 007046 005000 CLR R0 ;; ZERO THE CONSTANTS INDEX
1336 007050 012703 007226 MOV #SDBLK,R3 ;; SETUP THE OUTPUT POINTER
1337 007054 112723 000040 MOVB #'',(R3)+ ;; SET THE FIRST CHARACTER TO A BLANK
1338 007060 005002 2$: CLR R2 ;; CLEAR THE BCD NUMBER
1339 007062 016001 007216 3$: MOV $DTBL(R0),R1 ;; GET THE CONSTANT
1340 007066 160105 SUB R1,R5 ;; FORM THIS BCD DIGIT
1341 007070 002402 BLT 4$ ;; BR IF DONE
1342 007072 005202 INC R2 ;; INCREASE THE BCD DIGIT B 1
1343 007074 000774 BR 3$

```

E03

MAINDEC-11-DZKWA-F LINE FREQUENCY CLOCK PROGRAM
 DZKWA.F.P11 09-NOV-77 00:00

MACY11 30(1046) 07-DEC-77 12:50 PAGE 27
 CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

SEG 0030

```

1344 007076 060105          4$:  ADD    R1,R5          ;; ADD BACK THE CONSTANT
1345 007100 005702          TST    R2              ;; CHECK IF BCD DIGIT=0
1346 007102 001002          BNE    5$              ;; FALL THROUGH IF 0
1347 007104 105716          TSTB   (SP)           ;; STILL DOING LEADING 0'S?
1348 007106 100407          BMI    7$              ;; BR IF YES
1349 007110 106316          5$:  ASLB   (SP)           ;; MSD?
1350 007112 103003          BCC    6$              ;; BR IF NO
1351 007114 116663 000001 177777 MOVB   1(SP),-1(R3)    ;; YES--SET THE SIGN
1352 007122 052702 000060 6$:  BIS    #'0,R2        ;; MAKE THE BCD DIGIT ASCII
1353 007126 052702 000040 7$:  BIS    #' ',R2        ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
1354 007132 110223          MOVB   R2,(R3)+       ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
1355 007134 005720          TST    (R0)+          ;; JUST INCREMENTING
1356 007136 020027 000010 CMP    RO,#10          ;; CHECK THE TABLE INDEX
1357 007142 002746          BLT    2$              ;; GO DO THE NEXT DIGIT
1358 007144 003002          BGT    8$              ;; GO TO EXIT
1359 007146 010502          MOV    R5,R2          ;; GET THE LSD
1360 007150 000764          BR     6$              ;; GO CHANGE TO ASCII
1361 007152 105726          8$:  TSTB   (SP)+        ;; WAS THE LSD THE FIRST NON-ZERO?
1362 007154 100003          BPL    9$              ;; BR IF NO
1363 007156 116663 177777 177776 9$:  MOVB   -1(SP),-2(R3)  ;; YES--SET THE SIGN FOR TYPING
1364 007164 105013          CLRB   (R3)           ;; SET THE TERMINATOR
1365 007166 012605          MOV    (SP)+,R5       ;; POP STACK INTO R5
1366 007170 012603          MOV    (SP)+,R3       ;; POP STACK INTO R3
1367 007172 012602          MOV    (SP)+,R2       ;; POP STACK INTO R2
1368 007174 012601          MOV    (SP)+,R1       ;; POP STACK INTO R1
1369 007176 012600          MOV    (SP)+,R0       ;; POP STACK INTO R0
1370 007200 104400 007226 YPE    $DBLK          ;; NOW TYPE THE NUMBER
1371 007204 016666 000002 000004 MOV    2(SP),4(SP)    ;; ADJUST THE STACK
1372 007212 012616          MOV    (SP)+,(SP)
1373 007214 000002          RTI                    ;; RETURN TO USER
1374 007216 023420          $DTBL: 10000.
1375 007220 001750          1000.
1376 007222 000144          100.
1377 007224 000012          10.
1378 007226 000004          $DBLK: .BLKW 4
1379
1380 *****
1381 .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
1382
1383 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
1384 ;*OCTAL (ASCII) NUMBER AND TYPE IT.
1385 ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
1386 ;*CALL:
1387 ;*      MOV    NUM,-(SP)          ;; NUMBER TO BE TYPED
1388 ;*      TYPOS          ;; CALL FOR TYPEOUT
1389 ;*      .BYTE  N          ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
1390 ;*      .BYTE  M          ;; M=1 OR 0
1391 ;*
1392 ;*
1393 ;*
1394 ;*      ;; 1=TYPE LEADING ZEROS
1395 ;*      ;; 0=SUPPRESS LEADING ZEROS
1396 ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
1397 ;*$TYPOS OR $TYPOC
1398 ;*CALL:
1399 ;*      MOV    NUM,-(SP)          ;; NUMBER TO BE TYPED
1400 ;*      TYPON          ;; CALL FOR TYPEOUT

```

F03

MAINDEC-11-DZKWA-F LINE FREQUENCY CLOCK PROGRAM
 DZKWA.F.P11 09-NOV-77 00:00

MACY11 30(1046) 07-DEC-77 12:50 PAGE 28
 BINARY TO OCTAL (ASCII) AND TYPE

SEQ 0031

```

1400      ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
1401      ;*CALL:
1402      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
1403      ;*      TYPOC      ;;CALL FOR TYPEOUT
1404
1405      007236 017646 000000      $TYPOS: MOV      2(SP),-(SP)      ;; PICKUP THE MODE
1406      007242 116637 000001 007461  MOVB     1(SP),%OFILL      ;; LOAD ZERO FILL SWITCH
1407      007250 112637 007463      MOVB     (SP)+,%OMODE+1      ;; NUMBER OF DIGITS TO TYPE
1408      007254 062716 000002      ADD      #2,(SP)      ;; ADJUST RETURN ADDRESS
1409      007260 000406      $TYPON: BR
1410      007262 112737 000001 007461  $TYPOC: MOVB     #1,%OFILL      ;; SET THE ZERO FILL SWITCH
1411      007270 112737 000006 007463  MOVB     #6,%OMODE+1      ;; SET FOR SIX(6) DIGITS
1412      007276 112737 000005 007460  $TYPON: MOVB     #5,%OCNT      ;; SET THE ITERATION COUNT
1413      007304 010346      MOV      R3,-(SP)      ;; SAVE R3
1414      007306 010446      MOV      R4,-(SP)      ;; SAVE R4
1415      007310 010546      MOV      R5,-(SP)      ;; SAVE R5
1416      007312 113704 007463      MOVB     %OMODE+1,R4      ;; GET THE NUMBER OF DIGITS TO TYPE
1417      007316 005404      NEG      R4
1418      007320 062704 000006      ADD      #6,R4      ;; SUBTRACT IT FOR MAX. ALLOWED
1419      007324 110437 007462      MOVB     R4,%OMODE      ;; SAVE IT FOR USE
1420      007330 113704 007461      MOVB     %OFILL,R4      ;; GET THE ZERO FILL SWITCH
1421      007334 016605 000012      MOV      12(SP),R5      ;; PICKUP THE INPUT NUMBER
1422      007340 005003      CLR      R3      ;; CLEAR THE OUTPUT WORD
1423      007342 006105      1$: ROL      R5      ;; ROTATE MSB INTO "C"
1424      007344 000404      BR      3$      ;; GO DO MSB
1425      007346 006105      2$: ROL      R5      ;; FORM THIS DIGIT
1426      007350 006105      ROL      R5
1427      007352 006105      ROL      R5
1428      007354 010503      MOV      R5,R3
1429      007356 006103      3$: ROL      R3      ;; GET LSB OF THIS DIGIT
1430      007360 105337 007462      DECB     %OMODE      ;; TYPE THIS DIGIT?
1431      007364 100016      BPL     7$      ;; BR IF NO
1432      007366 042703 177770      BIC     #177770,R3      ;; GET RID OF JUNK
1433      007372 001002      BNE     4$      ;; TEST FOR 0
1434      007374 005704      TST     R4      ;; SUPPRESS THIS 0?
1435      007376 001403      BEQ     5$      ;; BR IF YES
1436      007400 005204      4$: INC      R4      ;; DON'T SUPPRESS ANYMORE 0'S
1437      007402 052703 000060      BIS     #'0,R3      ;; MAKE THIS DIGIT ASCII
1438      007406 052703 000040      5$: BIS     #' ,R3      ;; MAKE ASCII IF NOT ALREADY
1439      007412 110337 007456      MOVB     R3,%S      ;; SAVE FOR TYPING
1440      007416 104400 007456      TYPE     #S      ;; GO TYPE THIS DIGIT
1441      007422 105337 007460      7$: DECB     %OCNT      ;; COUNT BY 1
1442      007426 003347      BGT     2$      ;; BR IF MORE TO DO
1443      007430 002402      BLT     6$      ;; BR IF DONE
1444      007432 005204      INC     R4      ;; INSURE LAST DIGIT ISN'T A BLANK
1445      007434 000744      BR      2$      ;; GO DO THE LAST DIGIT
1446      007436 012605      6$: MOV      (SP)+,R5      ;; RESTORE R5
1447      007440 012604      MOV      (SP)+,R4      ;; RESTORE R4
1448      007442 012603      MOV      (SP)+,R3      ;; RESTORE R3
1449      007444 016666 000002 000004      MOV      2(SP),4(SP)      ;; SET THE STACK FOR RETURNING
1450      007452 012616      MOV      (SP)+,(SP)
1451      007454 000002      RTI
1452      007456      8$: .BYTE 0      ;; RETURN
1453      007457      .BYTE 0      ;; STORAGE FOR ASCII DIGIT
1454      007460      .BYTE 0      ;; TERMINATOR FOR TYPE ROUTINE
1455      007461      .BYTE 0      ;; OCTAL DIGIT COUNTER
1455      007461      .BYTE 0      ;; ZERO FILL SWITCH
    
```



```

1456 007462 000000
1457
1458
1459
1460
1461
1462
1463
1464
1465 007464
1466 007464 104400 001231
1467 007470 010046
1468 007472 005000
1469 007474 153700 001114
1470 007500 001004
1471
1472 007502 013746 001116
1473
1474 007506 104401
1475 007510 000426
1476 007512 005300
1477 007514 006300
1478 007516 006300
1479 007520 006300
1480 007522 062700 001236
1481 007526 012037 007536
1482 007532 001404
1483 007534 104400
1484 007536 000000
1485 007540 104400 001231
1486 007544 012037 007554
1487 007550 001404
1488 007552 104400
1489 007554 000000
1490 007556 104400 001231
1491 007562 011000
1492 007564 001004
1493 007566 012600
1494 007570 104400 001231
1495 007574 000207
1496 007576
1497 007576 013046
1498 007600 104401
1499 007602 005710
1500 007604 001770
1501 007606 104400 007614
1502 007612 000771
1503 007614 020040 000
1504 007620
1505
1506
1507
1508
1509
1510
1511

```

```

$OMODE: .WORD 0 ;:NUMBER OF DIGITS TO TYPE
;*****
.SBTTL ERROR MESSAGE TIMEOUT ROUTINE
;:THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
;:ERROR IS TO BE REPORTED. IT THEN OBTAINS FROM THE "ERROR TABLE" ($ERRTB)
;:AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
$ERRTYP:
TYPE $SCLF ;:"CARRIAGE RETURN" & "LINE FEED"
MOV RO, -(SP) ;:SAVE RO
CLR RO ;:PICKUP THE ITEM INDEX
BISB @($ITEMB,RO)
BNE 1$ ;:IF ITEM NUMBER IS ZERO, JUST
;:TYPE THE PC OF THE ERROR
MOV $ERRPC, -(SP) ;:SAVE $ERRPC FOR TIMEOUT
;:ERROR ADDRESS
TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
BR 6$ ;:GET OUT
1$: DEC RO ;:ADJUST THE INDEX SO THAT IT WILL
ASL RO ;:WORK FOR THE ERROR TABLE
ASL RO
ASL RO
ADD @($ERRTB,RO) ;:FORM TABLE POINTER
MOV (RO)+, 2$ ;:PICKUP "ERROR MESSAGE" POINTER
BEQ 3$ ;:SKIP TIMEOUT IF NO POINTER
TYPE "ERROR MESSAGE" ;:TYPE THE "ERROR MESSAGE"
;:"ERROR MESSAGE" POINTER GOES HERE
2$: .WORD 0 ;:"CARRIAGE RETURN" & "LINE FEED"
TYPE $SCLF ;:PICKUP "DATA HEADER" POINTER
MOV (RO)+, 4$ ;:SKIP TIMEOUT IF 0
BEQ 5$ ;:TYPE THE "DATA HEADER"
TYPE "DATA HEADER" ;:"DATA HEADER" POINTER GOES HERE
3$: .WORD 0 ;:"CARRIAGE RETURN" & "LINE FEED"
TYPE $SCLF ;:PICKUP "DATA TABLE" POINTER
MOV (RO), RO ;:GO TYPE THE DATA
BNE 7$ ;:RESTORE RO
MOV (SP)+, RO ;:"CARRIAGE RETURN" & "LINE FEED"
6$: .WORD 0 ;:RETURN
TYPE $SCLF
RTS PC
7$:
;:SAVE @ (RO)+ FOR TIMEOUT
MOV @ (RO)+, -(SP)
TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
TST (RO) ;:IS THERE ANOTHER NUMBER?
BR IF NO
8$: TYPE TWO(2) SPACES ;:TYPE TWO(2) SPACES
BR 7$ ;:LOOP
;:TWO(2) SPACES
;:ASCIZ / /
.EVEN
;*****
.SBTTL ERROR HANDLER ROUTINE
;:THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.
;:SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
;:AND GO TO $ERRTYP ON ERROR

```

H03

```

1512      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1513      ;*SW15=1      HALT ON ERROR
1514      ;*SW13=1      INHIBIT ERROR TYPEOUTS
1515      ;*SW10=1      BELL ON ERROR
1516      ;*SW09=1      LOOP ON ERROR
1517      ;*CALL
1518      ;*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
1519
1520      $ERROR:
1521      007620 010637 001174      MOV      SP,$REG6      ;GET THE CURRENT STACK POINTER VALUE
1522      007624 162737 000004 001174      SJR      #4,$REG6      ;RESTORE IT TO ITS "PRE ERROR TRAP" VALUE FOR PR
1523      007632 016637 000002 001176      MOV      2(SP),$REG7      ;GET THE PS OFF OF THE STACK
1524      007640 005037 001172      CLR      $REG5      ;PREPARE "$REG5" TO HOLD THE TEST #
1525      007644 113737 001102 001172      MOVB    $TSTNM,$REG5      ;TEST # IS HELD IN THE LOW BYTE OF "$TSTNM"
1526      007652 010037 001160      MOV      RO,$REG0      ;MOST OF THE TIME RO HAS GOOD STUFF IN IT ALSO
1527      007656 105237 001103      7$:      INCB    $ERFLG      ;SET THE ERROR FLAG
1528      007662 001775      BEQ      7$      ;DON'T LET THE FLAG GO TO ZERO
1529      007664 013777 001102 171246      MOV      $TSTNM,$DISPLAY ;DISPLAY TEST NUMBER AND ERROR FLAG
1530      007672 032777 002000 171236      BIT      #BIT10,$SWR      ;BELL ON ERROR?
1531      007700 001402      BEQ      1$      ;NO - SKIP
1532      007702 104400 001224      TYPE    $BELL      ;RING BELL
1533      007706 005237 001112      1$:      INC      $ERTTL      ;COUNT THE NUMBER OF ERRORS
1534      007712 011637 001116      MOV      (SP),$ERRPC      ;GET ADDRESS OF ERROR INSTRUCTION
1535      007716 162737 000002 001116      SUB      #2,$TRPC
1536      007724 117737 171166 001114      MOVB    $ERRPC,$ITEMB    ;STRIP AND SAVE THE ERROR ITEM CODE
1537      007732 032777 020000 171176      BIT      #BIT13,$SWR      ;SKIP TYPEOUT IF SET
1538      007740 001004      BNE      20$      ;SKIP TYPEOUTS
1539      007742 004737 007464      JSR      PC,$ERRTYP      ;GO TO USER ERROR ROUTINE
1540      007746 104400 001231      TYPE    $CARLF
1541      007752      20$:
1542      007752 005777 171160      2$:      TST      $SWR      ;HALT ON ERROR
1543      007756 100001      BPL      31$      ;SKIP IF CONTINUE
1544      007760 000000      HALT
1545      007762 022737 006216 000042      31$:    CMP      #SENDAD,$#42    ;ACT-11 AUTO-ACCEPT?
1546      007770 001001      BNE      3$      ;BRANCH IF NO
1547      007772 000000      HALT
1548      007774 032777 001000 171134      3$:      BIT      #BIT09,$SWR      ;LOOP ON ERROR SWITCH SET?
1549      010002 001402      BEQ      4$      ;BR IF NO
1550      010004 013716 001110      MOV      $LPERR,(SP)      ;FUDGE RETURN FOR LOOPING
1551      010010 005737 001222      4$:      TST      $ESCAPE      ;CHECK FOR AN ESCAPE ADDRESS
1552      010014 001402      BEQ      5$      ;BR IF NONE
1553      010016 013716 001222      MOV      $ESCAPE,(SP)    ;FUDGE RETURN ADDRESS FOR ESCAPE
1554      010022      5$:
1555      010022 000002      RTI      ;RETURN
1556      ;*****
1557
1558      .SBTTL TRAP DECODER
1559
1560      ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
1561      ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
1562      ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
1563      ;*GO TO THAT ROUTINE.
1564
1565      $TRAP:  MOV      RO,-(SP)      ;SAVE RO
1566      010026 016600 000002      MOV      2(SP),RO      ;GET TRAP ADDRESS
1567      010032 005740      TST      -(RO)      ;BACKUP BY 2

```

```

1568 010034 111000          MOVB   (R0),R0          ;;GET RIGHT BYTE OF TRAP
1569 010036 006300          ASL    R0              ;;POSITION FOR INDEXING
1570 010040 016000 010046    MOV    $TRPAD(F0),R0   ;;INDEX TO TABLE
1571 010044 000200          RTS    R0              ;;GO TO ROUTINE
1572
1573
1574          .SBTTL  TRAP TABLE
1575
1576          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
1577          ;*BY THE "TRAP" INSTRUCTION.
1578
1579          :          ROUTINE
1580          :          -----
1581 010046          $TRPAD:
1582 010046 006576          $TYPE  ;;CALL=TYPE      TRAP+0(104400)  TTY TYPEOUT ROUTINE
1583 010050 007262          $TYPOC ;;CALL=TYPOC     TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
1584 010052 007236          $TYPOS ;;CALL=TYPOS     TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
1585 010054 007276          $TYPON ;;CALL=TYPON     TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
1586 010056 007012          $TYPDS ;;CALL=TYPDS     TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)
1587          ;*****
1588
1589          .SBTTL  POWER DOWN AND UP ROUTINES
1590
1591          :POWER DOWN ROUTINE
1592 010060 012737 010222 000024 $PWRDN: MOV    $ILLUP,2,$PWRVEC ;;SET FOR FAST UP
1593 010066 012737 000340 000026    MOV    #340,2,$PWRVEC+2 ;;PRIO:7
1594 010074 010046          MOV    R0,-(SP)        ;;PUSH R0 ON STACK
1595 010076 010146          MOV    R1,-(SP)        ;;PUSH R1 ON STACK
1596 010100 010246          MOV    R2,-(SP)        ;;PUSH R2 ON STACK
1597 010102 010346          MOV    R3,-(SP)        ;;PUSH R3 ON STACK
1598 010104 010446          MOV    R4,-(SP)        ;;PUSH R4 ON STACK
1599 010106 010546          MOV    R5,-(SP)        ;;PUSH R5 ON STACK
1600 010110 013746 010450          MOV    POWPUS,-(SP)    ;;PUSH POWPUS ON STACK
1601 010114 013746 010452          MOV    POWPOP,-(SP)    ;;PUSH POWPOP ON STACK
1602 010120 013746 010240          MOV    POWMES,-(SP)    ;;PUSH POWMES ON STACK
1603 010124 013746 001672          MOV    T0001,-(SP)    ;;PUSH T0001 ON STACK
1604 010130 010637 010226          MOV    SP,$SAVR6       ;;SAVE SP
1605 010134 012737 010146 000024    MOV    $PWRUP,2,$PWRVEC ;;SET UP VECTOR
1606 010142 000000          HALT
1607 010144 000776          BR    .-2             ;;HANG UP
1608
1609          :POWER UP ROUTINE
1610 010146 013706 010226          $PWRUP: MOV    $SAVR6,SP   ;;GET SP
1611 010152 005037 010226          CLR    $SAVR6         ;;WAIT LOOP FOR THE TTY
1612 010156 005237 010226          IS:   INC    $SAVR6    ;;WAIT FOR THE INC
1613 010162 001375          BNE    IS             ;;OF <POWPUS>,<POWPOP>,<POWMES>,<T0001> WORD
1614 010164 012605          MOV    (SP)+,R5       ;;POP STACK INTO R5
1615 010166 012604          MOV    (SP)+,R4       ;;POP STACK INTO R4
1616 010170 012603          MOV    (SP)+,R3       ;;POP STACK INTO R3
1617 010172 012602          MOV    (SP)+,R2       ;;POP STACK INTO R2
1618 010174 012601          MOV    (SP)+,R1       ;;POP STACK INTO R1
1619 010176 012600          MOV    (SP)+,R0       ;;POP STACK INTO R0
1620 010200 012737 010060 000024    MOV    $PWRDN,2,$PWRVEC ;;SET UP THE POWER DOWN VECTOR
1621 010206 012737 000340 000026    MOV    #340,2,$PWRVEC+2 ;;PRIO:7
1622 010214 104400          TYPE
1623 010216 010230          $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER

```

```

1624 010220 000002          RTI
1625 010222 000000          $ILLUP: HALT                ;: THE POWER UP SEQUENCE WAS STARTED
1626 010224 000776          BR                .-2        ;: BEFORE THE POWER DOWN WAS COMPLETE
1627 010226 000000          $SAVR6: 0              ;: PUT THE SP HERE
1628 010230 005015 047520 042527 $POWER: .ASCIZ <15><12>"POWER"
1629 010236 000122
1630
1631 010240 005015 042522 052123 POWMES: .EVEN
1632 010246 051101 044524 043516          .ASCIZ <15> <12> "RESTARTING AFTER A POWER FIALURE" <15> <12> <12>
1633 010254 040440 052106 051105
1634 010262 040440 050040 053517
1635 010270 051105 043040 040511
1636 010276 052514 042522 005015
1637 010304 000012
1638 010306 005015 046412 026504 STMES: .ASCIZ <15><12><12>"MD-11-DZKWA-F LINE FREQUENCY CLOCK TEST"<15><12>
1639 010314 030461 042055 045532
1640 010322 040527 043055 046040
1641 010330 047111 020105 051106
1642 010336 050505 042525 041516
1643 010344 020131 046103 041517
1644 010352 020113 042524 052123
1645 010360 005015 000
1646
1647 010363 124 040522 050120 TRPMES: .ASCIZ "TRAPPED TO LOC 4 FROM LOCATION "
1648 010370 042105 052040 020117
1649 010376 047514 020103 020064
1650 010404 051106 046517 046040
1651 010412 041517 052101 047511
1652 010420 020116 000040
1653 010424 042522 052123 051101 TRPM2S: .ASCIZ "RESTARTING PROGRAM"
1654 010432 044524 043516 050040
1655 010440 047522 051107 046501
1656 010446 000
1657
1658 010450 010450          .EVEN
1659 010452 001234          POWPUS: WORD
1660 010454 020040 020040 020040          POWPOP: WORD
1661 010462 020040 020040 020040          EM1: .ASCIZ "                LKS LKS "
1662 010470 020040 020040 020040
1663 010476 020040 020040 020040
1664 010504 020040 020040 020040
1665 010512 020040 045514 020123
1666 010520 020040 020040 045514
1667 010526 020123 020040 020040
1668 010534 000
1669 010535 050 041520 020051 DH1: .ASCIZ "(PC) (PS) (SP) TEST# WAS S/B "
1670 010542 020040 024040 051520
1671 010550 020051 020040 024040
1672 010556 050123 020051 020040
1673 010564 052040 051505 021524
1674 010572 020040 053440 051501
1675 010600 020040 020040 051440
1676 010606 041057 020040 020040
1677 010614 000040
1678
1679 010616 001116 001176 001174          .EVEN
          DT1: $ERRPC,$REG7,$REG6,$REG5,LKS,$GDDAT

```

K03

MAINDEC-11-DZKWA-F LINE FREQUENCY CLOCK PROGRAM
DZKWA.F.P11 09-NOV-77 00:00

MACY11 30(1046) 07-DEC-77 12:50 PAGE 33
POWER DOWN AND UP ROUTINES

SEQ 0036

1680	010624	001172	177546	001124	
1681	010632	000000			0
1682	010634	046103	041517	020113	EM2: .ASCIZ "CLOCK FAILED TO INTERRUPT"
1683	010642	040506	046111	042105	
1684	010650	052040	020117	047111	
1685	010656	042524	051122	050125	
1686	010664	000124			
1687	010666	050050	024503	020040	DH2: .ASCIZ "(PC) (PS) (SP) TEST# (LKS) "
1688	010674	020040	050050	024523	
1689	010702	020040	020040	051450	
1690	010710	024520	020040	020040	
1691	010716	042524	052123	020043	
1692	010724	020040	046050	051513	
1693	010732	020051	020040	000	
1694		010740			.EVEN
1695	010740	001116	001176	001174	DT2: \$ERRPC,\$REG7,\$REG6,\$REG5,LKS
1696	010746	001172	177546		
1697	010752	000000			0
1698	010754	046103	041517	020113	EM3: .ASCIZ "CLOCK INTERRUPTED WHEN THE PROCESSOR PRIORITY WAS TOO HIGH"
1699	010762	047111	042524	051122	
1700	010770	050125	042524	020104	
1701	010776	044127	047105	052040	
1702	011004	042510	050040	047522	
1703	011012	042503	051523	051117	
1704	011020	050040	044522	051117	
1705	011026	052111	020131	040527	
1706	011034	020123	047524	020117	
1707	011042	044510	044107	000	
1708	011047	050	041520	020051	DH3: .ASCIZ "(PC) (PS) (SP) TEST# (LKS) "
1709	011054	020040	024040	051520	
1710	011062	020051	020040	024040	
1711	011070	050123	020051	020040	
1712	011076	052040	051505	021524	
1713	011104	020040	024040	045514	
1714	011112	024523	020040	000040	
1715					.EVEN
1716	011120	001116	001176	001174	DT3: \$ERRPC,\$REG7,\$REG6,\$REG5,LKS
1717	011126	001172	177546		
1718	011132	000000			0
1719	011134	046103	041517	020113	EM4: .ASCIZ "CLOCK GIVES UNEQUAL # OF PULSES OVER TWO EQUAL PERIODS OF TIME"
1720	011142	044507	042526	020123	
1721	011150	047125	050505	040525	
1722	011156	020114	020043	043117	
1723	011164	050040	046125	042523	
1724	011172	020123	053117	051105	
1725	011200	052040	047527	042440	
1726	011206	052521	046101	050040	
1727	011214	051105	047511	051504	
1728	011222	047440	020106	044524	
1729	011230	042515	000		
1730	011233	050	041520	020051	DH4: .ASCIZ "(PC) (PS) (SP) TEST# 1ST 2ND"<15><12>"
1731	011240	020040	024040	051520	
1732	011246	020051	020040	024040	
1733	011254	050123	020051	020040	
1734	011262	052040	051505	021524	
1735	011270	020040	030440	052123	

1736	011276	020040	020040	031040
1737	011304	042116	005015	020040
1738	011312	020040	020040	020040
1739	011320	020040	020040	020040
1740	011328	020040	020040	020040
1741	011336	020040	020040	020040
1742	011344	020040	020040	020040
1743	011352	020040	020040	020040
1744	011360	020040	044522	042117
1745	011368	020040	042520	044522
1746	011376	042117	000	
1747	011378	011370	081176	001174
1748	011378	081116	081162	001164
1749	011378	081172		
1750	011400	000000		
1751	011400	047516	020123	042522
1752	011400	047516	051105	051105
1753	011400	047516	047520	047520
1754	011400	047516	047524	047524
1755	011400	047516	044124	044124
1756	011400	047516	042104	042104
1757	011400	047516	000	
1758	011400	047516	020051	020051
1759	011472	020040	024040	051520
1760	011500	020051	020040	024040
1761	011500	050123	020051	020040
1762	011514	020040	051505	021524
1763	011522	042522	040440	042104
1764		011530	051523	000
1765	011530	001116	081176	001174
1766	011536	001172	001120	
1767	011542	000000		
1768	011544	020101	047516	051440
1769	011552	041501	020113	044524
1770	011560	042515	052517	020124
1771	011566	040510	020123	041517
1772	011574	052503	042522	000104
1773	011602	050050	024503	020040
1774	011610	020040	050050	024523
1775	011616	020040	020040	051450
1776	011624	024520	020040	020040
1777	011632	042524	052123	020043
1778	011640	020040	046050	051513
1779	011646	020051	020040	000
1780		011654		
1781	011654	001116	001176	001174
1782	011662	001172	177546	
1783	011666	000000		
1784	011670	051127	047117	020107
1785	011676	047503	042116	052111
1786	011704	047511	020116	047503
1787	011712	042504	020123	042527
1788	011720	042522	050040	052125
1789	011726	047440	052116	020117
1790	011734	052123	041501	020113
1791	011742	054502	044440	052116

```

.EVEN
DT4:  $ERRPC,$REG7,$REG6,$REG5,$REG1,$REG2
0
EM5:  .ASCIZ "LKS REGISTER RESPONDS TO ANOTHER ADDRESS"
DMS:  .ASCIZ "(PC) (PS) (SP) TEST# ADDRESS"
.EVEN
DT5:  $ERRPC,$REG7,$REG6,$REG5,$GDADR
0
EM6:  .ASCIZ "A NO SACK TIMEOUT HAS OCCURED"
DMS:  .ASCIZ "(PC) (PS) (SP) TEST# (LKS) "
.EVEN
DT6:  $ERRPC,$REG7,$REG6,$REG5,LKS
0
EM7:  .ASCIZ "WRONG CONDITION CODES WERE PUT ONTO STACK BY INTERRUPT"

```

```

1792 011750 051105 052522 052120
1793 011756 000
1794 011757 050 041520 020051 DH7: .ASCIZ "(PC) (PS) (SP) TEST# CC WAS CC S/B"
1795 011764 020040 024040 051520
1796 011772 020051 020040 024040
1797 012000 050123 020051 020040
1798 012006 052040 051505 021524
1799 012014 020040 041440 020103
1800 012022 040527 020123 041440
1801 012030 020103 027523 000102
1802
1803 012036 001116 001176 001174 .EVEN
1804 012044 001172 000776 001124 DT7: $ERRPC,$REG7,$REG6,$REG5,BUF1,$GDDAT
1805 012052 000000 0
1806 012054 051127 047117 020107 EM10: .ASCIZ "WRONG PC PUT ONTO THE STACK BY AN INTERRUPT"
1807 012062 041520 050040 052125
1808 012070 047440 052116 020117
1809 012076 044124 020105 052123
1810 012104 041501 020113 054502
1811 012112 040440 020116 047111
1812 012120 042524 051122 050125
1813 012126 000124
1814 012130 050050 024503 020040 DH10: .ASCIZ "(PC) (PS) (SP) TEST# a(SP)WAS a(SP)S/B "
1815 012136 020040 050050 024523
1816 012144 020040 020040 051450
1817 012152 024520 020040 020040
1818 012160 042524 052123 020043
1819 012166 040040 051450 024520
1820 012174 040527 020123 024100
1821 012202 050123 051451 041057
1822 012210 020040 000
1823 012214 012214 .EVEN
1824 012214 001116 001176 001174 DT10: $ERRPC,$REG7,$REG6,$REG5,BUF2,$GDDAT
1825 012222 001172 000774 001124
1826 012230 000000 0
1827 012232 051124 042531 020104 EM11: .ASCIZ "TRYED TO ACCESS THE LKS REGISTER, AND TRAPPED"
1828 012240 047524 040440 041503
1829 012246 051505 020123 044124
1830 012254 020105 045514 020123
1831 012262 042522 044507 052123
1832 012270 051105 020054 047101
1833 012276 020104 051124 050101
1834 012304 042520 000104
1835 012310 050050 027503 020040 DH11: .ASCIZ "(PC) (PS) (SP) TEST#"
1836 012316 020040 050050 024523
1837 012324 020040 020040 051450
1838 012332 024520 020040 020040
1839 012340 042524 052123 000043
1840
1841 012346 001116 001176 001174 .EVEN
1842 012354 001172 DT11: $ERRPC,$REG7,$REG6,$REG5
1843 012356 000000 0
1844 000001 .END

```

A0004	002104	387#	390				
A0005	002172	408#	411				
A0006	002302	435#	438				
A0007	002434	465#	468				
A0010	002614	505#	508				
A0011	002716	534#	537				
A0012	003102	576#	579				
A0013	003172	606#	609				
A0014	003546	685	694#				
A0015	003664	706	718#				
A0016	004002	730	742#				
A0017	004120	754	766#				
A0020	004236	778	790#				
A0021	004354	802	814#				
A0022	004512	841	843#				
A0023	004560	857#	860				
A0024	004702	885#	886				
A0025	005012	909#	912				
A0026	005164	945#	946				
A0027	005300	971#	972				
A0030	005440	989	999#				
A0031	005526	1019#	1022				
A0032	005660	1056#	1059				
A0033	006016	1089#	1092				
BIT0	= 000001	121#					
BIT00	= 000001	111#	121				
BIT01	= 000002	110#	120				
BIT02	= 000004	109#	119				
BIT03	= 000010	108#	118				
BIT04	= 000020	107#	117				
BIT05	= 000040	106#	116				
BIT06	= 000100	105#	115				
BIT07	= 000200	104#	114				
BIT08	= 000400	103#	113	1205			
BIT09	= 001000	102#	112	1213	1548		
BIT1	= 000002	120#					
BIT10	= 002000	101#	1530				
BIT11	= 004000	100#	1220				
BIT12	= 010000	99#					
BIT13	= 020000	98#	1537				
BIT14	= 040000	97#	1191				
BIT15	= 100000	96#					
BIT2	= 000004	119#					
BIT3	= 000010	118#					
BIT4	= 000020	117#					
BIT5	= 000040	116#					
BIT6	= 000100	115#					
BIT7	= 000200	114#					
BIT8	= 000400	113#					
BIT9	= 001000	112#					
BPTVEC	= 000014	128#					
BUF1	= 000776	140#	1064#	1073	1097#	1803	
BUF2	= 000774	139#	1065#	1098#	1106	1824	
B0005	002210	409	414#				
B0006	002320	436	441#				
B0007	002452	466	471#				

B0010	002632	506	511#		
B0011	002734	535	540#		
B0012	003120	577	582#		
B0013	003210	607	612#		
B0014	003560	695	697#		
B0015	003676	719	721#		
B0016	004014	743	745#		
B0017	004132	767	769#		
B0020	004250	791	793#		
B0021	004366	815	817#		
B0023	004576	858	863#		
B0025	005030	910	915#		
B0030	005456	1001	1003#	1004	
B0031	005544	1020	1025#		
B0032	005676	1057	1062#		
B0033	006034	1090	1095#		
C0007	002462	474#	476		
C0010	002642	514#	516		
C0011	003014	553#	556		
C0013	003224	616#	619		
C0025	005054	902	923#		
C0031	005600	1034#	1037		
C0032	005742	1066	1073#		
C0033	006100	1099	1106#	1108	
DI9P =	177570	43#	191		
DH1	010535	243	1669#		
DH10	012130	278	1814#		
DH11	012310	283	1835#		
DH2	010666	248	1687#		
DH3	011047	253	1708#		
DH4	011233	258	1730#		
DH5	011457	263	1757#		
DH6	011602	268	1773#		
DH7	011757	273	1794#		
DISPLA	001140	191#	327*	1234*	1529*
DISPRE	000174	149#	327		
DSMR =	177570	42#	190		
DT1	010616	244	1679#		
DT10	012214	279	1824#		
DT11	012346	284	1841#		
DT2	010740	249	1695#		
DT3	011120	254	1716#		
DT4	011370	259	1747#		
DT5	011530	264	1765#		
DT6	011654	269	1781#		
DT7	012036	274	1803#		
D0007	002474	457	479#		
D0010	002654	492	519#		
D0011	003032	554	559#		
D0013	003242	617	622#		
D0031	005616	1035	1040#		
EMTVEC=	000030	131#	309*	310*	
EM1	010454	242	1660#		
EM10	012054	277	1806#		
EM11	012232	282	1827#		
EM2	010634	247	1682#		

EM3	010754	252	1698#				
EM4	011134	257	1719#				
EM5	011406	262	1750#				
EM6	011544	267	1768#				
EM7	011670	272	1784#				
ERRVEC=	000004	124#	1196#	1197*	1199*	1202*	
E0001	001730	340#	346#				
E0002	001776	360#					
E0003	002042	374#					
E0004	002116	391#					
E0005	002204	412#					
E0006	002314	439#					
E0007	002446	469#					
E0010	002626	509#					
E0011	002730	538#					
E0012	003114	580#					
E0013	003204	610#					
E0014	003556	696#					
E0015	003674	720#					
E0016	004012	744#					
E0017	004130	768#					
E0020	004246	792#					
E0021	004364	816#					
E0022	004510	842#					
E0023	004572	861#					
E0024	004722	877#	890#				
E0025	005024	913#					
E0026	005204	940#	950#				
E0027	005332	977#					
E0030	005472	999#	1007#				
E0031	005540	1023#	1031#				
E0032	005672	1060#					
E0033	006030	1093#					
E10006	002342	446#					
E10007	002470	477#					
E10010	002650	517#					
E10012	003134	587#					
E10013	003236	620#					
E10023	004614	867#					
E10025	005050	921#					
E10031	005612	1038#					
E10033	006110	1108#					
E1005	002226	418#					
E1011	003026	550#	557#				
E20007	002502	481#					
E20010	002662	521#					
E20011	003040	563#					
E20013	003334	643#					
E20025	005064	925#					
E20031	005624	1044#					
E30013	003366	653#					
E40013	003444	669#					
F0013	003262	627#	632#				
GNS =	***** U	148#	1582#	1583	1584	1585	1586
G0013	003276	628#	631#				
H0013	003322	639#	642#				

SENDCT	006164	315	1137*																
SENDMG	006232	1139	1154*																
SENULL	006247	1142	115'																
SEOP	006130	1127*																	
SEOPCT	006156	315*	1134*	1138															
SERFLG	001103	175*	1180	1209	1211	1217*	1238	1527*	1556										
SERMAX	001115	181*	318*	1211	1233*	1258													
SEROR	007620	309	1520*																
SERRPC	001116	182*	1472	1534*	1535*	1536	1556	1679	1695	1716	1747	1765	1781	1803					
		1824	1841																
SERRTB	001236	241*	1480																
SERRTY	007464	1465*	1539																
SERTTL	001112	179*	1533*	1556															
SESCAP	001222	219*	317*	1232*	1551	1553	1556												
SFILLC	001154	198*	1277	1311															
SFILLS	001153	197*	1311																
SGDAOR	001120	183*	684*	693*	699*	710*	717*	723*	734*	741*	747*	758*	765*	771*					
		782*	789*	795*	806*	813*	819*	832*	839*	1765									
SGODAT	001124	185*	357*	369*	384*	400*	427*	455*	570*	682*	708*	732*	756*	780*					
		804*	830*	853*	865	875*	888	899*	934*	948	960*	975	987*	1075*					
		1108*	1679	1803	1824														
		1143*																	
SGETH2	006206	18																	
SHO =	000000	18																	
SICNT	001104	176*	1224*	1225	1227*	1237													
SILLUP	010222	1592	1625*																
SITEMB	001114	180*	1469	1536*	1556														
SLF	001232	223*	1311	1556															
SLPADR	001106	177*	319*	342*	355*	370*	383*	401*	428*	456*	491*	531*	573*	602*					
		613*	635*	646*	656*	683*	709*	733*	757*	781*	805*	831*	854*	876*					
		900*	935*	961*	988*	1016*	1053*	1086*	1215*	1230*	1235	1237							
SLPERR	001110	178*	320*	1215	1231*	1237	1550												
SMAIL =	***** U	331	1230	1262	1542														
SMXCNT	006574	1228	1237*																
SMULL	001152	196*	1279	1311															
SOCNT	007460	1412*	1441*	1454*															
SOMODE	007462	1407*	1411*	1416	1419*	1430*	1456*												
SOVER	006560	1192	1208	1216	1226	1234*													
SPASS	001100	173*	1131*	1132*	1140	1154	1222	1238											
SPWER	010230	1623	1628*																
SPWRON	010060	313	1592*	1620															
SPWRMG	010216	1623*																	
SPWRUP	010146	1605	1610*																
SQUES	001230	221*	1311	1556															
SROCHR =	***** U	1587																	
SRODEC =	***** U	1587																	
SROLIN =	***** U	1587																	
SROOCT =	***** U	1587																	
SREGAD	001156	200*																	
SREGO	001160	202*	1526*																
SREG1	001162	203*	669*	1747															
SREG2	001164	204*	670*	1747															
SREG3	001166	205*																	
SREG4	001170	206*																	
SREG5	001172	207*	1524*	1525*	1679	1695	1716	1747	1765	1781	1803	1824	1841						
SREG6	001174	208*	1521*	1522*	1679	1695	1716	1747	1765	1781	1803	1824	1841						
SREG7	001176	209*	1523*	1679	1695	1716	1747	1765	1781	1803	1824	1841							

.SCMTA	1#	8#	164
.SDB2D	1#		
.SDB2D	1#		
.SDIV	1#		
.SEOP	1#	8#	1118
.SERRO	1#	8#	1505
.SERRT	1#	8#	1457
.SMULT	1#		
.SPOWE	1#	8#	1587
.SRAND	1#		
.SRDEE	1#		
.SRDOC	1#		
.SREAO	1#		
.SR2AZ	1#		
.SSAVE	1#		
.SSB2D	1#		
.SSB2D	1#		
.SSCOP	1#	8#	1174
.SSIZE	1#		
.SSUPR	1#		
.STRAP	1#	8#	1556
.STYPB	1#		
.STYPD	1#	8#	1311
.STYPE	1#	8#	1238
.STYPO	1#	8#	1379
.S4OCA	1#		

. ABS. 012360 000

ERRORS DETECTED: 0

DZKRAF.BIN, DZKRAF.LST/CRF/SOL/NL: *OC=DZKRAF.SML, DZKRAF.P11

RUN-TIME: 8 10 .8 SECONDS

RUN-TIME RATIO: 88/20=4.3

CORE USED: 28K (55 PAGES)

K04