

KW11-L

LINE FREQUENCY CLOCK TEST
MD-11-DZKWA-E

EP-DZKWA-E-DL-A

OCT 1976

COPYRIGHT ©1976

digital

FICHE 1 OF 1

Made in U.S.A.

The microfiche card contains 40 frames of test data, organized in a 10x4 grid. Each frame displays a small table or chart with various numerical and graphical data points, typical of a clock test report. The data appears to be organized into columns, possibly representing different test parameters or time intervals. The frames are arranged in 10 rows and 4 columns.

167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220

167400
000000

001100

177776

177774

177772

177570

177570

000000

000001

000002

000003

000004

000005

000006

```

%
.ABS
.ENABL AMA
.LIST ME
.NLIST MC MD, CND
$SWR=167400
$TN=0

.ENABL ABS
.MCALL .HEADER .SCATCH .SEOP .EQUAT .SWRHI .SWRLO .SSCOPE .SETUP
.MCALL .STYPOCT .STYPDEC .STRAP .SPOWER .SERROR .STYPE .STARS .SERRTYP .SCMTAG
.MCALL .SETPRI .GETPRI .SACT11
.TITLE MAINDEC-11-DZKWA-E LINE FREQUENCY CLOCK PROGRAM
;*COPYRIGHT (C) 1970,1972,1975,1976
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY J. COMEAU
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZGAC-B1),AUG 29,1975.
;*

.SBTTL OPERATIONAL SWITCH SETTINGS
;*
;* SWITCH USE
;* -----
;* 15 HALT ON ERROR
;* 14 LOOP ON TEST
;* 13 INHIBIT ERROR TYPEOUTS
;* 11 INHIBIT ITERATIONS
;* 10 BELL ON ERROR
;* 9 LOOP ON ERROR
;* 8 LOOP ON TEST IN SWR<7:0>
;* 7-0 #OF TEST TO LOOP ON IF SWR<9> IS SET

.SBTTL BASIC DEFINITIONS
;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
PS= 177776 ;;PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLMT= 177774 ;;STACK LIMIT REGISTER
PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570 ;;HARDWARE SWITCH REGISTER
DDISP= 177570 ;;HARDWARE DISPLAY REGISTER

;*GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0 ;;GENERAL REGISTER
R1= %1 ;;GENERAL REGISTER
R2= %2 ;;GENERAL REGISTER
R3= %3 ;;GENERAL REGISTER
R4= %4 ;;GENERAL REGISTER
R5= %5 ;;GENERAL REGISTER
R6= %6 ;;GENERAL REGISTER
```

000007
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

R7= %7
.EQUIV R6,SP
.EQUIV R7,PC

:: GENERAL REGISTER
:: STACK POINTER
:: PROGRAM COUNTER

::*PRIORITY LEVEL DEFINITIONS

PRO= 0
PR1= 40
PR2= 100
PR3= 140
PR4= 200
PR5= 240
PR6= 300
PR7= 340

:: PRIORITY LEVEL 0
:: PRIORITY LEVEL 1
:: PRIORITY LEVEL 2
:: PRIORITY LEVEL 3
:: PRIORITY LEVEL 4
:: PRIORITY LEVEL 5
:: PRIORITY LEVEL 6
:: 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

```

279 000004
280 000002
281 000001
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319 000174
320 000176
321
322
323 000200 000137 001400
324
325
326
327
328 000204
329 000046
330 000046 006166
331 000052
332 000052 000000
333 000204

```

```

BIT02= 4
BIT01= 2
BIT00= 1
.EQUIV BIT09,BIT9
.EQUIV BIT08,BIT8
.EQUIV BIT07,BIT7
.EQUIV BIT06,BIT6
.EQUIV BIT05,BIT5
.EQUIV BIT04,BIT4
.EQUIV BIT03,BIT3
.EQUIV BIT02,BIT2
.EQUIV BIT01,BIT1
.EQUIV BIT00,BIT0

```

```

;*BASIC "CPU" TRAP VECTOR ADDRESSES
ERRVEC= 4 ;: TIME OUT AND OTHER ERRORS
RESVEC= 10 ;: RESERVED AND ILLEGAL INSTRUCTIONS
TBITVEC= 14 ;: "T" BIT
TRTVEC= 14 ;: TRACE TRAP
BPTVEC= 14 ;: BREAKPOINT TRAP (BPT)
IOTVEC= 20 ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
PWRVEC= 24 ;: POWER FAIL
EMTVEC= 30 ;: EMULATOR TRAP (EMT) **ERROR**
TRAPVEC= 34 ;: "TRAP" TRAP
TKVEC= 60 ;: TTY KEYBOARD VECTOR
TPVEC= 64 ;: TTY PRINTER VECTOR
PIRQVEC= 240 ;: PROGRAM INTERRUPT REQUEST VECTOR
;MISCELANHUS EQUATES
LKS=177546
NOP=240
BUF2=774
BUF1=776

```

.SBTTL TRAP CATCHER

```

.=0
;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS

```

```

.=174
DISPREG: .WORD 0 ;: SOFTWARE DISPLAY REGISTER
SWREG: .WORD 0 ;: SOFTWARE SWITCH REGISTER

```

```

.SBTTL STARTING ADDRESS(ES)
JMP @#KSTART ;: JUMP TO STARTING ADDRESS OF PROGRAM
;*****

```

```

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

```

```

334
335
336
337
338
339
340
341 001100 001100
342 001100 000000
343 001102 000
344 001103 000
345 001104 000000
346 001106 000000
347 001110 000000
348 001112 000000
349 001114 000
350 001115 001
351 001116 000000
352 001120 000000
353 001122 000000
354 001124 000000
355 001126 000000
356 001130 000000
357 001132 000000
358 001134 000000
359 001136 177570
360 001140 177570
361 001142 177560
362 001144 177562
363 001146 177564
364 001150 177566
365 001152 000
366 001153 002
367 001154 012
368 001155 000
369 001156 000000
370
371
372 001160 000000
373 001162 000000
374 001164 000000
375 001166 000000
376 001170 000000
377 001172 000000
378 001174 000000
379 001176 000000
380 001200 000000
381 001202 000000
382 001204 000000
383 001206 000000
384 001210 000000
385 001212 000000
386 001214 000000
387 001216 000000
388 001220 000000
389 001222 000000

```

.SBTTL COMMON TAGS

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

SCMTAG: . =1100

```

$PASS: .WORD 0
$STSTM: .BYTE 00
$ERFLG: .BYTE 00
$ICNT: .WORD 00
$LPADR: .WORD 00
$LPERR: .WORD 00
$ERTTL: .WORD 00
$ITEMB: .BYTE 0
$ERMAX: .BYTE 1
$ERRPC: .WORD 0
$GDADR: .WORD 0
$BDADR: .WORD 0
$GDDAT: .WORD 0
$BDDAT: .WORD 0
$SWR: .WORD DSWR
DISPLAY: .WORD DDISP
$TKS: 177560
$TKB: 177562
$TPS: 177564
$TPB: 177566
$NULL: .BYTE 0
$FILLS: .BYTE 2
$FILLC: .BYTE 12
$STPFLG: .BYTE 0
$REGAD: .WORD 0
$REG0: .WORD 0
$REG1: .WORD 0
$REG2: .WORD 0
$REG3: .WORD 0
$REG4: .WORD 0
$REG5: .WORD 0
$REG6: .WORD 0
$REG7: .WORD 0
$TMP0: .WORD 0
$TMP1: .WORD 0
$TMP2: .WORD 0
$TMP3: .WORD 0
$TMP4: .WORD 0
$TMP5: .WORD 0
$TMP6: .WORD 0
$TMP7: .WORD 0
$TIMES: 0
$ESCAPE: 0

```

```

:: START OF COMMON TAGS
:: CONTAINS PASS COUNT
:: CONTAINS THE TEST NUMBER
:: CONTAINS ERROR FLAG
:: CONTAINS SUBTEST ITERATION COUNT
:: CONTAINS SCOPE LOOP ADDRESS
:: CONTAINS SCOPE RETURN FOR ERRORS
:: CONTAINS TOTAL ERRORS DETECTED
:: CONTAINS ITEM CONTROL BYTE
:: CONTAINS MAX. ERRORS PER TEST
:: CONTAINS PC OF LAST ERROR INSTRUCTION
:: CONTAINS ADDRESS OF 'GOOD' DATA
:: CONTAINS ADDRESS OF 'BAD' DATA
:: CONTAINS 'GOOD' DATA
:: CONTAINS 'BAD' DATA
:: RESERVED--NOT TO BE USED

:: ADDRESS OF SWITCH REGISTER
:: ADDRESS OF DISPLAY REGISTER
:: TTY KBD STATUS
:: TTY KBD BUFFER
:: TTY PRINTER STATUS REG. ADDRESS
:: TTY PRINTER BUFFER REG. ADDRESS
:: CONTAINS NULL CHARACTER FOR FILLS
:: CONTAINS # OF FILLER CHARACTERS REQUIRED
:: INSERT FILL CHARS. AFTER A "LINE FEED"
:: "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
:: CONTAINS THE ADDRESS FROM
:: WHICH ($REG0) WAS OBTAINED
:: CONTAINS (($REGAD)+0)
:: CONTAINS (($REGAD)+2)
:: CONTAINS (($REGAD)+4)
:: CONTAINS (($REGAD)+6)
:: CONTAINS (($REGAD)+10)
:: CONTAINS (($REGAD)+12)
:: CONTAINS (($REGAD)+14)
:: CONTAINS (($REGAD)+16)
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: MAX. NUMBER OF ITERATIONS
:: ESCAPE ON ERROR ADDRESS

```

390 001224 177607 000377
391 001230 077
392 001231 015
393 001232 000012

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

394
395 001234 000000
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411 001236
412 001236 010424
413 001240 010505
414 001242 010566
415 001244 000000
416
417 001246 010604
418 001250 010636
419 001252 010710
420 001254 000000
421
422 001256 010724
423 001260 011017
424 001262 011070
425 001264 000000
426
427 001266 011104
428 001270 011203
429 001272 011340
430 001274 000000
431
432 001276 011356
433 001300 011427
434 001302 011500
435 001304 000000
436
437 001306 011514
438 001310 011552
439 001312 011624
440 001314 000000
441
442 001316 011640
443 001320 011727
444 001322 012006
445 001324 000000
446
447 001326 012024
448 001330 012100
449 001332 012164

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 \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

```

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

\$ERRTB:

```

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
    
```

450	001334	000000		0		
451						
452	001336	012202			EM11	;"TRAPPED TRYING TO ACCESS LKS REGISTER"
453	001340	012260			DH11	;"(PC) (PS) (SP) TEST#"
454	001342	012316			DT11	;\$ERRPC,\$REG7,\$REG6,\$REG5
455	001344	000000		0		
456						
457						
458						
459		001400				;STARTUP CODE
460	001400	012706	001000		KSTART:	MOV #1000,SP ;INITIALIZE THE STACK SO WE CAN CALL THE TYPEOUT
461	001404	005046				CLR -(SP)
462	001406	013746	000034			MOV 34,-(SP) ;SAVE CURRENT TRAP VECTOR
463	001412	012737	001422	000034		MOV #64\$,34 ;SETUP NEW TRAP VECTOT
464	001420	104400				TRAP ;PUSH OLD PSW AN PCON STACK
465	001422	016666	000002	000006	64\$:	MOV 2(SP),6(SP)
466	001430	012716	001436			MOV #65\$, (SP)
467	001434	000002				RTI ;REPLACE OLD PC WITH NEW
468	001436	012637	000034		65\$:	MOV (SP)+,34 ;RESTORE PSW
469	001442	004737	006546			JSR PC,\$TYPE ;RESTORE OLD TRAP VECTOR
470	001446	010256				STMS ;PRINTOUT STARTUP MESSAGE
471	001450				START:	STMS ;ADDRESS OF MESSAGE "MAINDEC-11-DZKWA-E"
472	001450	012706	001100			MOV #SCMTAG,R6 ;FIRST LOCATION TO BE CLEARED
473	001454	005026				CLR (R6)+ ;CLEAR MEMORY LOCATION
474	001456	022706	001126			CMP #SBDDAT,R6 ;DONE?
475	001462	001374				BNE -6 ;LOOP BACK IF NO
476	001464	012706	001000			MOV #1000,SP ;SETUP THE STACK POINTER
477	001470	012737	006274	000020		MOV #SCOPE,@IOTVEC ;IOT VECTOR FOR SCOPE ROUTINE
478	001476	012737	000340	000022		MOV #340,@IOTVEC+2 ;LEVEL 7
479	001504	012737	007570	000030		MOV #ERROR,@EMTVEC ;EMT VECTOR FOR ERROR ROUTINE
480	001512	012737	000340	000032		MOV #340,@EMTVEC+2 ;LEVEL 7
481	001520	012737	007774	000034		MOV #STRAP,@TRAPVEC ;TRAP VECTOR FOR TRAP CALLS
482	001526	012737	000340	000036		MOV #340,@TRAPVEC+2 ;LEVEL 7
483	001534	012737	010030	000024		MOV #SPWRDN,@PWRVEC ;POWER FAILURE VECTOR
484	001542	012737	000340	000026		MOV #340,@PWRVEC+2 ;LEVEL 7
485	001550	013737	006134	006126		MOV SENDCT,\$EOPCT ;SETUP END-OF-PROGRAM COUNTER
486	001556	005037	001220			CLR \$TIMES ;INITIALIZE NUMBER OF ITERATIONS
487	001562	005037	001222			CLR \$ESCAPE ;CLEAR THE ESCAPE ON ERROR ADDRESS
488	001566	112737	000001	001115		MOV #1,\$ERMAX ;ALLOW ONE ERROR PER TEST
489	001574	012737	001574	001106		MOV #,\$LPADR ;INITIALIZE THE LOOP ADDRESS FOR SCOPE
490	001602	012737	001602	001110		MOV #,\$LPERR ;SETUP THE ERROR LOOP ADDRESS
491	001610	013746	000004			MOV @#4,-(SP) ;SAVE ERROR VECTOR
492	001614	013746	000006			MOV @#6,-(SP)
493	001620	012737	001634	000004		MOV #64\$,4 ;SET UP TIME OUT VECTOR
494	001626	005777	177304			TST @SWR ;TRY TO REFERENCE HARDWARE SWR
495	001632	000407				BR 65\$;BRANCH IF NO TIMEOUT TRAP OCCURS
496	001634	012737	000176	001136	64\$:	MOV #SWREG,SWR ;POINT TO SOFTWARE SWR
497	001642	012737	000174	001140		MOV #DISPRG,DISPLAY ;POINT TO SOFTWARE DISPLAY REG
498	001650	022626				CMP (SP)+,(SP)+ ;RESTORE STACK
499	001652	012637	000006		65\$:	MOV (SP)+,@#6 ;RESTORE ERROR VECTOR
500	001656	012637	000004			MOV (SP)+,@#4
501	001662	005737	000042			TST 42 ;LOADED BY A MONITOR
502	001666	001401				BEQ T0001 ;BR IF NO
503	001670	000005				RESET ;YES--GENERATE AN INIT
504						
505						

```

506
507
508 .SBTTL TEST THAT THE LKS CAN BE REFERENCED WITHOUT A BUS ERROR
509 :LKS ACCESS TEST
510 001672 000004 T0001: SCOPE
511 001674 012737 001730 000004 MOV #E0001, @#4 ;PREPARE FOR ADDRESSING THE LKS REGISTER. BAD HARDWARE
512 001702 012737 000340 000006 MOV #340, @#6 ;COULD CAUSE A TRAP TO 4
513 001710 012737 001716 001106 MOV #R0001, $LPADR ;TIGHTEN UP THE SCOPE LOOP A BIT IN CASE OF AN ERROR
514 001716 012706 001000 R0001: MOV #1000, $P ;SETUP THE STACK POINTER IN CASE OF AN ERROR
515 001722 005037 177546 I0001: CLR @#LKS ;JUST REFERENCE LKS. DONT WORRY IF IT DIDNT CLEAR YET
516 001726 000401 BR T0002 ;WE DIDNT TRAP IF WE REACH HERE. GO ON TO NEXT TEST
517 001730 104011 E0001: ERROR 11 ;ERROR:::TRAPED TRYING TO ACCESS THE LKS REGISTER
518
519
520 .SBTTL TEST THAT START CLEARS LINE CLOCK INTERRUPT ENABLE BIT
521 :TEST THAT START CLEARS LINE CLOCK INTERRUPT ENABLE BIT
522 001732 000004 T0002: SCOPE
523 001734 012737 006222 000004 MOV #TRAP0, @#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
524 001742 012737 000340 000006 MOV #340, @#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
525 001750 012737 001756 001106 MOV #R0002, $LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
526 001756 000005 R0002: RESET
527 001760 012737 000200 001124 MOV #200, $GDDAT ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
528 001766 032737 000100 177546 BIT #100, LKS ;TEST THE INTERRUPT ENABLE BIT
529 001774 001401 BEQ T0003
530 001776 104001 E0002: ERROR 1 ;ERROR, CLOCK INTERRUPT ENABLE NOT CLEARED BY INIT
531
532
533 .SBTTL TEST THAT START SETS CLOCK FLAG
534 :TEST THAT START SETS CLOCK FLAG
535 002000 000004 T0003: SCOPE
536 002002 012737 006222 000004 MOV #TRAP0, @#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
537 002010 012737 000340 000006 MOV #340, @#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
538 002016 012737 000200 001124 MOV #200, $GDDAT ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
539 002024 012737 002032 001106 MOV #R0003, $LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
540 002032 000005 R0003: RESET ;SHOULD SET THE CLOCK FLAG
541 002034 105737 177546 TSTB LKS ;FIND OUT IF IT DID
542 002040 100401 BMI T0004 ;GO ON TO THE NEXT TEST IF IT SET THE CLOCK FLAG
543 002042 104001 E0003: ERROR 1 ;ERROR, CLOCK FLAG NOT SET BY INIT
544
545
546
547
548 .SBTTL TEST THAT CLOCK FLAG WILL SET AFTER SUFFICIENT PERIOD OF TIME (20 MS MIN)
549 :TEST THAT CLOCK FLAG WILL SET AFTER SUFFICIENT PERIOD OF TIME (20 MS MIN)
550 002044 000004 T0004: SCOPE
551 002046 012737 006222 000004 MOV #TRAP0, @#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
552 002054 012737 000340 000006 MOV #340, @#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
553 002062 012737 002070 001106 MOV #R0004, $LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
554 002070 012737 000200 001124 R0004: MOV #200, $GDDAT ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
555 002076 005037 177546 CLR LKS ;CLEAR THE CLOCK FLAG
556 002102 005000 CLR R0 ;AND A TIMER LOCATION
557 002104 105737 177546 A0004: TSTB LKS ;IS CLOCK FLAG SET
558 002110 100403 BMI T0005
559 002112 005200 INC R0 ;NO, INCREMENT COUNT 003 WAIT FOR SOMEMORE
560 002114 001373 BNE A0004 ;WAIT SUFFICIENT AMOUNT OF TIME FOR CLOCK
561 002116 104001 E0004: ERROR 1 ;ERROR, CLOCK FLAG FAILED TO SET

```

MO1

```

562
563
564
565      .SBTTL TEST THAT INTERRUPT ENABLE BIT MAY BE SET
566      :TEST THAT INTERRUPT ENABLE BIT MAY BE SET
567      T0005: SCOPE
568      002120 000004          MOV      #TRAP0,2#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
569      002122 012737 006222 000004      MOV      #340,2#6      ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
570      002130 012737 000340 000006      MOV      #100,$GDDAT   ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
571      002136 012737 000100 001124      MOV      #R0005,$LPADR ;SETUP LOOP BACK ADDRESS IN CASE OF ERROR
572      002144 012737 002164 001106      MOV      PR7,-(SP)    ;;PUT NEW PS ON STACK
573      002152 013746 000340          MOV      #64$,-(SP)   ;;PUT NEW PC ON STACK
574      002156 012746 002164          RTI          ;; POP NEW PC AND PS
575      002162 000002
576      002164 005037 177546      64$:
577      002170 005003          R0005: CLR      LKS
578      002172 105737 177546      A0005: CLR      R3      ;INITIALIZE A COUNTER LOCATION
579      002176 100404          BMI      B0005        ;IS THE CLOCK FLAG SET?
580      002200 005203          INC      R3           ;IF SO, CONTINUE ON WITH THE TEST
581      002202 001373          BNE      A0005        ;IF NOT INCREMENT THE COUNTER LOCATION
582      002204 104001          E0005: ERROR 1      ;AND GO TEST THE CLOCK FLAG AGAIN, UNLESS...
583      002206 000410          BR       T0006       ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
584      002210
585      002210 012737 000100 177546      B0005: MOV      #100,LKS ;CLEAR CLOCK FLAG AND SET INTERRUPT ENABLE
586      002216 032737 000100 177546      BIT      #100,LKS    ;IS INTERRUPT ENABLE SET?
587      002224 001001          BNE      T0006
588      002226 104001          E1005: ERROR 1      ;ERROR INTERRUPT ENABLE NOT SET
589
590
591
592      .SBTTL TEST THAT INTERRUPT ENABLE BIT MAY BE CLEARED
593      :TEST THAT INTERRUPT ENABLE BIT MAY BE CLEARED
594      T0006: SCOPE
595      002230 000004          MOV      #TRAP0,2#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
596      002232 012737 006222 000004      MOV      #340,2#6      ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
597      002240 012737 000340 000006      MOV      #0,$GDDAT     ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
598      002246 012737 000000 001124      MOV      #R0006,$LPADR ;SETUP LOOP BACK ADDRESS IN CASE OF AN ERROR
599      002254 012737 002274 001106      MOV      PR7,-(SP)    ;;PUT NEW PS ON STACK
600      002262 013746 000340          MOV      #64$,-(SP)   ;;PUT NEW PC ON STACK
601      002266 012746 002274          RTI          ;; POP NEW PC AND PS
602      002272 000002
603      002274 005037 177546      64$:
604      002274 005003          R0006: CLR      LKS
605      002300 005003          CLR      R3           ;INITIALIZE A COUNTER LOCATION
606      002302 105737 177546      A0006: TSTB    LKS      ;IS THE CLOCK FLAG SET?
607      002306 100404          BMI      B0006        ;IF SO, CONTINUE ON WITH THE TEST
608      002310 005203          INC      R3           ;IF NOT INCREMENT THE COUNTER LOCATION
609      002312 001373          BNE      A0006        ;AND GO TEST THE CLOCK FLAG AGAIN, UNLESS...
610      002314 104001          E0006: ERROR 1      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
611      002316 000412          BR       T0007
612      002320
613      002320 012737 000100 177546      B0006: MOV      #100,LKS ;CLEAR CLOCK FLAG AND SET INTERRUPT ENABLE
614      002326 005037 177546          CLR      LKS         ;CLEAR INTERRUPT ENABLE
615      002332 032737 000100 177546      BIT      #100,LKS    ;TEST THE INTERRUPT ENABLE BIT
616      002340 001401          BEQ      T0007
617      002342 104001          E10006: ERROR 1     ;ERROR, ERROR INTERRUPT BIT CAN NOT BE CLEARED

```

```

618
619
620 .SBTTL TEST THAT CLOCK INTERRUPTS TO CORRECT VECTOR ADDRESS
621 ;TEST THAT CLOCK INTERRUPTS TO CORRECT VECTOR ADDRESS
622 T0007: SCOPE
623 002344 000004 006222 000004 MOV #TRAP0,2#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
624 002346 012737 000340 000006 MOV #340,2#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
625 002362 012737 000300 001124 MOV #300,$GDDAT ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
626 002370 012737 002412 001106 MOV #R0007,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
627 002376 012737 002474 000100 MOV #D0007,100 ;SET UP VECTOR RETURN POINTER
628 002404 012737 000340 000102 MOV #340,2#102 ;1 INTERRUPT IS ENOUGH
629 002412 012706 001000 R0007: MOV #1000,SP ;GET STACK READY FOR INTERRUPTS
630 002416 005046 CLR -(SP) ;DROP CPU PRIORITY LEVEL
631 002420 012746 002426 MOV #1$,-(SP) ;SET RETURN ADDRESS ON STACK FROM 'RTI'
632 002424 000002 RTI ;RETURN TO NEXT INSTRUCTION
633 002426 005037 177546 1$: CLR LKS
634 002432 005003 CLR R3 ;INITIALIZE A COUNTER LOCATION
635 002434 105737 177546 A0007: TSTB LKS ;IS THE CLOCK FLAG SET?
636 002440 100404 BMI B0007 ;IF SO, CONTINUE ON WITH THE TEST
637 002442 005203 INC R3 ;IF NOT INCREMENT THE COUNTER LOCATION
638 002444 001373 BNE A0007 ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
639 002446 104001 E0007: ERROR 1 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
640 002450 000415 BR T0010
641 002452
642 002452 012737 000100 177546 B0007: MOV #100,LKS ;ENABLE INTERRUPT
643 002460 005000 CLR R0
644 002462 005200 C0007: INC R0
645 002464 000240 NOP ;STALL FOR TIME
646 002466 001375 BNE C0007 ;WAIT FOR INTERRUPT
647 002470 104002 E10007: ERROR 2 ;ERROR, DIDNT GET INTERRUPT
648 002472 000404 BR T0010
649 002474 105737 177546 D0007: TSTB LKS ;ENTER HERE IF INTERRUPTED
650 002500 100401 BMI T0010
651 002502 104001 E20007: ERROR 1 ;ERROR, INTERRUPT NOT CAUSED BY CLOCK
652
653
654
655 .SBTTL TEST THAT CLOCK WILL INTERRUPT WITH PROCESSOR AT PRIORITY 5
656 ;TEST THAT CLOCK WILL INTERRUPT WITH PROCESSOR AT PRIORITY 5
657 ;NOTE: IF THERE IS NO HARDWARE 'PSW' THIS TEST WILL BE SKIPPED
658 T0010: SCOPE
659 002504 000004 006222 000004 MOV #TRAP0,2#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
660 002506 012737 000340 000006 MOV #340,2#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
661 002522 012737 002544 001106 MOV #R0010,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
662 002530 012737 002654 000100 MOV #D0010,100 ;SET UP VECTOR RETURN POINTER
663 002536 012737 000340 000102 MOV #340,2#102 ;NO INTERRUPTS ALLOWED AFTER THE FIRST ONE
664 002544 005037 177546 R0010: CLR LKS
665 002550 013746 000004 MOV 4,-(SP) ;SAVE BUS TIMEOUT VECTOR CONTENTS
666 002554 012737 002572 000004 MOV #1$,4 ;SET A SERVICE 'PC' IN CASE TIMEOUT OCCURS
667 002562 012737 000240 177776 MOV #PR5,PS ;DO WE HAVE A HARDWARE 'PSW'?
668 002570 000404 BR 2$ ;BRANCH IF YES
669 002572 022626 1$: CMP (SP)+,(SP)+ ;RESTORE STACK FROM TIMEOUT
670 002574 012637 000004 MOV (SP)+,4 ;RESTORE BUS TIMEOUT VECTOR CONTENTS
671 002600 000431 BR T0011
672 002602 012637 000004 2$: MOV (SP)+,4 ;RESTORE BUS TIMEOUT VECTOR CONTENTS
673 002606 012706 001000 MOV #1000,SP ;INITIALIZE THE STACK POINTER
    
```

TEST THAT CLOCK WILL INTERRUPT WITH PROCESSOR AT PRIORITY 5

```

674 002612 005003          CLR      R3          ;INITIALIZE A COUNTER LOCATION
675 002614 105737 177546 A0010: TSTB   LKS          ;IS THE CLOCK FLAG SET?
676 002620 100404          BMI     B0010        ;IF SO, CONTINUE ON WITH THE TEST
677 002622 005203          INC     R3           ;IF NOT INCREMENT THE COUNTER LOCATION
678 002624 001373          BNE    A0010        ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS
679 002626 104001          E0010: ERROR 1      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
680 002630 000415          BR     T0011
681 002632          B0010:          ;
682 002632 012737 000100 177546 MOV     #100,LKS     ;ENABLE INTERRUPT
683 002640 005000          CLR     R0
684 002642 005200          C0010: INC     R0
685 002644 000240          NOP
686 002646 001375          BNE    C0010        ;STALL FOR SOME TIME
687 002650 104002          E10010: ERROR 2    ;WAIT FOR INTERRUPT
688 002652 000404          BR     T0011        ;ERROR, INTERRUPT FAILED TO OCCUR
689 002654 105737 177546 D0010: TSTB   LKS          ;ENTER HERE IF INTERRUPTED
690 002660 100401          BMI     T0011
691 002662 104001          E20010: ERROR 1    ;ERROR, INTERRUPT DID NOT CLEAR THE CLOCK FLAG

```

.SBTTL TEST THAT CLOCK WILL NOT INTERRUPT WITH PROCESSOR PRIORITY 6

:TEST THAT CLOCK WILL NOT INTERRUPT WITH PROCESSOR PRIORITY 6

:NOTE: IF THERE IS NO HARDWARE 'PSW' THIS TEST WILL BE SKIPPED

```

692 002664 000004          T0011: SCOPE
693 002666 012737 006222 000004 MOV     #TRAP0,0#4   ;SETUP VECTOR IN CASE OF UNFORSEEN PROBLEMS
694 002674 012737 000340 000006 MOV     #340,0#6     ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
695 002702 012737 002710 001106 MOV     #R0011,$LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
696 002710 005037 177546 R0011: CLR     LKS
697 002714 005003          CLR     R3          ;INITIALIZE A COUNTER LOCATION
698 002716 105737 177546 A0011: TSTB   LKS          ;IS THE CLOCK FLAG SET?
699 002722 100404          BMI     B0011        ;IF SO, CONTINUE ON WITH THE TEST
700 002724 005203          INC     R3           ;IF NOT INCREMENT THE COUNTER LOCATION
701 002726 001373          BNE    A0011        ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS
702 002730 104001          E0011: ERROR 1      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
703 002732 000443          BR     T0012
704 002734          B0011:          ;
705 002734 012706 001000 MOV     #1000,SP     ;INITIALIZE THE STACK POINTER
706 002740 013746 000004 MOV     4,-(SP)       ;SAVE BUS TIMEOUT VECTOR CONTENTS
707 002744 012737 002762 000004 MOV     #1$,4        ;SET SERVICE 'PC' IN CASE TIMEOUT OCCURS
708 002752 012737 000300 177776 MOV     #PR6,PS      ;DO WE HAVE A HARDWARE 'PSW'?
709 002760 000404          BR     2$           ;BRANCH IF YES
710 002762 022626          1$:  CMP     (SP)+,(SP)+ ;RESTORE STACK FROM BUS TIMEOUT
711 002764 012637 000004 MOV     (SP)+,4      ;RESTORE BUS TIMEOUT VECTOR CONTENTS
712 002770 000424          BR     T0012
713 002772 012637 000004          2$:  MOV     (SP)+,4      ;RESTORE BUS TIMEOUT VECTOR CONTENTS
714 002776 012737 003026 000100 MOV     #E1011,100  ;SET UP VECTOR RETURN
715 003004 012737 000100 177546 MOV     #100,LKS     ;ENABLE INTERRUPT
716 003012 005003          CLR     R3          ;INITIALIZE A COUNTER LOCATION
717 003014 105737 177546 C0011: TSTB   LKS          ;IS THE CLOCK FLAG SET?
718 003020 100404          BMI     D0011        ;IF SO, CONTINUE ON WITH THE TEST
719 003022 005203          INC     R3           ;IF NOT INCREMENT THE COUNTER LOCATION
720 003024 001373          BNE    C0011        ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS
721 003026 104001          E1011: ERROR 1      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
722 003030 000715          BR     T0011
723 003032          D0011:          ;

```

```

730 003032 000240      NOP
731 003034 000240      NOP
732 003036 000401      BR      T0012      ;GIVE CLOCK EXTRA TIME TO INTERRUPT
733 003040 104003      E20011: ERROR 3      ;ERROR, CLOCK INTERRUPTED WITHOUT HAVING PRIORITY
734
735
736
737
738
739      .SBTTL TEST THAT RESET SETS CLOCK FLAG
740      :TEST THAT RESET SETS CLOCK FLAG
741      T0012: SCOPE
742      MOV      #200,$GDDAT      ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
743      MOV      #TRAP0,2#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
744      MOV      #340,2#6      ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
745      MOV      #R0012,$LPADR      ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
746      R0012: CLR      LKS      ;CLEAR CLOCK FLAG
747      CLR      R3      ;INITIALIZE A COUNTER LOCATION
748      A0012: TSTB     LKS      ;IS THE CLOCK FLAG SET?
749      BMI      B0012      ;IF SO, CONTINUE ON WITH THE TEST
750      INC      R3      ;IF NOT INCREMENT THE COUNTER LOCATION
751      BNE      A0012      ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
752      E0012: ERROR 1      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
753      BR      T0013
754      B0012: CLR      LKS
755      RESET      ;SHOULD SET CLOCK FLAG
756      TSTB     LKS
757      BMI      T0013
758      E10012: ERROR 1      ;ERROR, RESET DIDN'T SET CLOCK FLAG
759
760
761      .SBTTL TEST LINE CLOCK REPEATABILITY
762      :TEST LINE CLOCK REPEATABILITY
763      :MAKE SURE THAT OVER TWO EQUAL PERIODS OF TIME
764      :THE CLOCK PUTS OUT THE SAME NUMBER OF PULSES
765      T0013: SCOPE
766      R0013: CLR      R0      ;CLEAR 1ST TIME COUNT
767      CLR      R1      ;CLEAR 1ST CLOCK COUNT
768      MOV      PR7,-(SP)      ;PUT NEW PS ON STACK
769      MOV      #64$,-(SP)      ;PUT NEW PC ON STACK
770      RTI      ;POP NEW PC AND PS
771      64$:
772      R1013: MOV      #R1013,$LPADR ;ERROR IN NEXT FEW INSTRUCTIONS CAUSES A SHORT SCOPE LO
773      CLR      LKS
774
775      ;SYNC ON CLOCK FLAG A COUPLE OF TIMES
776      ;INITIALIZE A COUNTER LOCATION
777      A0013: CLR      R3
778      TSTB     LKS      ;IS THE CLOCK FLAG SET?
779      BMI      B0013      ;IF SO, CONTINUE ON WITH THE TEST
780      INC      R3      ;IF NOT INCREMENT THE COUNTER LOCATION
781      BNE      A0013      ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
782      E0013: ERROR 1      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
783      BR      T0014
784      B0013:
785      R2013: MOV      #R2013,$LPADR ;MAKE SCOPE LOOP SHORT IN CASE OF AN ERROR
786      CLR      LKS
787      CLR      R3      ;INITIALIZE A COUNTER LOCATION

```

```

786 003224 105737 177546      C0013: TSTB   LKS           ; IS THE CLOCK FLAG SET?
787 003230 100404              BMI   D0013      ; IF SO, CONTINUE ON WITH THE TEST
788 003232 005203              INC   R3         ; IF NOT INCREMENT THE COUNTER LOCATION
799 003234 001373              BNE   C0013      ; AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
790 003236 104001      E10013: ERROR 1 ; CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
791 003240 000473              BR    T0014
792 003242              D0013:
793 003242 005037 177546      CLR   LKS
794 003246 105737 177546      F0013: TSTB   LKS           ; IS CLOCK FLAG SET
795 003252 100003              BPL   G0013      ; NO
796 003254 005201              INC   R1         ; +1 TO CLOCK COUNT
797 003256 005037 177546      CLR   LKS
798 003262 005200      G0013: INC   R0         ; CLEAR CLOCK IF SET
799 003264 001370              BNE   F0013      ; +1 TO TIME COUNT
800 003266 005000              CLR   R0         ; REPEAT UNTIL R0=0
801 003270 005002              CLR   R0         ; CLEAR 2ND TIME COUNT
802 003272 012737 003272 001106 R3013: MOV   #R3013,$LPADR ; CLEAR 2ND CLOCK COUNT
803 003300 005037 177546      CLR   LKS        ; INSURE A SHORT SCOPE LOOP
804
805 003304 005003              CLR   R3
806 003306 105737 177546      H0013: TSTB   LKS           ; SYNC ON CLOCK FLAG TWICE
807 003312 100404              BMI   J0013      ; INITIALIZE A COUNTER LOCATION
808 003314 005203              INC   R3         ; IS THE CLOCK FLAG SET?
809 003316 001373              BNE   H0013      ; IF SO, CONTINUE ON WITH THE TEST
810 003320 104001      E20013: ERROR 1 ; IF NOT INCREMENT THE COUNTER LOCATION
811 003322 000442              BR    T0014      ; AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
812 003324              J0013:
813 003324 012737 003324 001106 R4013: MOV   #R4013,$LPADR ; CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
814 003332 005037 177546      CLR   LKS
815 003336 005003              CLR   R3
816 003340 105737 177546      K0013: TSTB   LKS           ; INITIALIZE A COUNTER LOCATION
817 003344 100404              BMI   L0013      ; IS THE CLOCK FLAG SET?
818 003346 005203              INC   R3         ; IF SO, CONTINUE ON WITH THE TEST
819 003350 001373              BNE   K0013      ; IF NOT INCREMENT THE COUNTER LOCATION
820 003352 104001      E30013: ERROR 1 ; AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
821 003354 000425              BR    T0014      ; CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
822 003356              L0013:
823 003356 012737 003140 001106 MOV   #R0013,$LPADR ; MUST LOOP BACK TO BEGINING OF TEST IF EEROR COMES NOW
824 003364 005037 177546      CLR   LKS
825 003370 105737 177546      M0013: TSTB   LKS           ; IS CLOCK FLAG SET
826 003374 100003              BPL   N0013      ; NO
827 003376 005202              INC   R2         ; +1 TO CLOCK COUNT
828 003400 005037 177546      CLR   LKS
829 003404 005200      N0013: INC   R0         ; CLEAR CLOCK IF SET
830 003406 001370              BNE   M0013      ; +1 TO TIME COUNT
831 003410 020102              CMP   R1,R2      ; REPEAT UNTIL R0=0
832 003412 001406              BEQ   T0014      ; IS 1ST CLOCK COUNT EQUAL TO 2ND CLOCK COUNT?
833 003414 010137 001162      E40013: MOV   R1,$REG1 ; YES
834 003420 010237 001164      MOV   R2,$REG2 ; GET R1 READY FOR PRINTOUT
835 003424 104004              ERROR 4 ; GET R2 READY FOR PRINTOUT
836 003426 000240              NOP              ; ERROR, CLOCK FLAG OCCURRED DIFFERENT
837
838
839
840
841

```

.SBTTL LINE CLOCK REGISTER ADDRESSING TEST
:LINE CLOCK REGISTER ADDRESSING TEST

```

842 ;TEST THAT THE "LKS" REGISTER CAN NOT BE ADDRESSED AS ANYTHING BUT 177546
843 ;SET A LOCATION THAT IS CLOSE(DIFFERS BY 1 BIT) TO THE LKS REGISTER
844 ;TO 100. IF THE LKS ALSO CHANGES, THEN SIGNAL AN ERROR
845 003430 000004 T0014: SCOPE
846 003432 005037 001124 CLR $GDDAT
847 003436 012737 003500 001106 MOV #R0014,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
848 003444 012737 157546 001120 MOV #157546,$GDADR ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 13 CLEAR
849 003452 012737 003516 000004 MOV #A0014,4 ;SETUP VECTOR IN CASE IT IS NONEXISTANT
850 003460 012737 000340 000006 MOV #340,6
851 003466 013746 000340 MOV PR7,-(SP) ;;PUT NEW PS ON STACK
852 003472 012746 003500 MOV #64$,-(SP) ;;PUT NEW PC ON STACK
853 003476 000002 RTI ;;POP NEW PC AND PS
854 003500
855 003500 012706 001000 64$:
856 003504 005037 177546 R0014: MOV #1000,SP ;SETUP THE STACK
857 003510 012777 000100 175402 I0014: MOV #100,$SGDADR ;SET THE "CLOSE" ADDRESS TO = 100
858 003516 032737 000100 177546 A0014: BIT #100,LKS ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
859 003524 001401 BEQ B0014
860 003526 104005 E0014: ERROR 5 ;IT AFFECTED "LKS" -- ERROR
861 003530 005037 177546 B0014: CLR LKS
862 003534 012737 003546 000004 MOV #T0015,4
863 003542 005077 175352 CLR $SGDADR ;CLEAR OUT THE "CLOSE" ADDRESS
864
865
866
867 .SBTTL LINE CLOCK REGISTER ADDRESSING TEST
868 :LINE CLOCK REGISTER ADDRESSING TEST
869 003546 000004 T0015: SCOPE
870 003550 012737 003634 000004 MOV #A0015,4 ;SETUP VECTOR IN CASE IT IS NONEXISTANT
871 003556 012737 000340 000006 MOV #340,6
872 003564 005037 001124 CLR $GDDAT
873 003570 012737 003616 001106 MOV #R0015,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
874 003576 012737 177146 001120 MOV #177146,$GDADR ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 8 CLEAR
875 003604 013746 000340 MOV PR7,-(SP) ;;PUT NEW PS ON STACK
876 003610 012746 003616 MOV #64$,-(SP) ;;PUT NEW PC ON STACK
877 003614 000002 RTI ;;POP NEW PC AND PS
878 003616
879 003616 012706 001000 64$:
880 003622 005037 177546 R0015: MOV #1000,SP ;SETUP THE STACK
881 003626 012777 000100 175264 I0015: MOV #100,$SGDADR ;SET THE "CLOSE" ADDRESS TO = 100
882 003634 032737 000100 177546 A0015: BIT #100,LKS ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
883 003642 001401 BEQ B0015
884 003644 104005 E0015: ERROR 5 ;IT AFFECTED "LKS" -- ERROR
885 003646 005037 177546 B0015: CLR LKS
886 003652 012737 003664 000004 MOV #T0016,4
887 003660 005077 175234 CLR $SGDADR ;CLEAR OUT THE "CLOSE" ADDRESS
888
889
890
891 .SBTTL LINE CLOCK REGISTER ADDRESSING TEST
892 :LINE CLOCK REGISTER ADDRESSING TEST
893 003664 000004 T0016: SCOPE
894 003666 012737 003752 000004 MOV #A0016,4 ;SETUP VECTOR IN CASE IT IS NONEXISTANT
895 003674 012737 000340 000006 MOV #340,6
896 003702 005037 001124 CLR $GDDAT
897 003706 012737 003734 001106 MOV #R0016,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR

```

```

898 003714 012737 177446 001120      MOV      #177446,$GDADR ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 6 CLEAR
899 003722 013746 000340              MOV      PR7,-(SP)    ;:PUT NEW PS ON STACK
900 003726 012746 003734              MOV      #64$,-(SP) ;:PUT NEW PC ON STACK
901 003732 000002              RTI                    ;: POP NEW PC AND PS
902 003734
903 003734 012706 001000      64$:
R0016: MOV      #1000,SP ;SETUP THE STACK
904 003740 005037 177546      CLR      LKS
905 003744 012777 000100 175146 I0016: MOV      #100,$GDADR ;SET THE "CLOSE" ADDRESS TO = 100
906 003752 032737 000100 177546 A0016: BIT      #100,LKS ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
907 003760 001401              BEQ      B0016
908 003762 104005      E0016: ERROR 5 ;IT AFFECTED "LKS" -- ERROR
909 003764 005037 177546      B0016: CLR      LKS
910 003770 012737 004002 000004      MOV      #T0017 ,4
911 003776 005077 175116      CLR      $GDADR ;CLEAR OUT THE "CLOSE" ADDRESS
912
913
914
915
916
917 004002 000004      .SBTTL LINE CLOCK REGISTER ADDRESSING TEST
:LINE CLOCK REGISTER ADDRESSING TEST
T0017: SCOPE
918 004004 012737 004070 000004      MOV      #A0017,4 ;SETUP VECTOR IN CASE IT IS NONEXISTANT
919 004012 012737 000340 000006      MOV      #340,6
920 004020 005037 001124              CLR      $GDADR
921 004024 012737 004052 001106      MOV      #R0017,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
922 004032 012737 177556 001120      MOV      #177556,$GDADR ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 3 SET
923 004040 013746 000340              MOV      PR7,-(SP) ;:PUT NEW PS ON STACK
924 004044 012746 004052              MOV      #64$,-(SP) ;:PUT NEW PC ON STACK
925 004050 000002              RTI                    ;: POP NEW PC AND PS
926 004052
927 004052 012706 001000      64$:
R0017: MOV      #1000,SP ;SETUP THE STACK
928 004056 005037 177546      CLR      LKS
929 004062 012777 000100 175030 I0017: MOV      #100,$GDADR ;SET THE "CLOSE" ADDRESS TO = 100
930 004070 032737 000100 177546 A0017: BIT      #100,LKS ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
931 004076 001401              BEQ      B0017
932 004100 104005      E0017: ERROR 5 ;IT AFFECTED "LKS" -- ERROR
933 004102 005037 177546      B0017: CLR      LKS
934 004106 012737 004120 000004      MOV      #T0020 ,4
935 004114 005077 175000      CLR      $GDADR ;CLEAR OUT THE "CLOSE" ADDRESS
936
937
938
939
940
941 004120 000004      .SBTTL LINE CLOCK REGISTER ADDRESSING TEST
:LINE CLOCK REGISTER ADDRESSING TEST
T0020: SCOPE
942 004122 012737 004206 000004      MOV      #A0020,4 ;SETUP VECTOR IN CASE IT IS NONEXISTANT
943 004130 012737 000340 000006      MOV      #340,6
944 004136 005037 001124              CLR      $GDADR
945 004142 012737 004170 001106      MOV      #R0020,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
946 004150 012737 177566 001120      MOV      #177566,$GDADR ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 4 SET
947 004156 013746 000340              MOV      PR7,-(SP) ;:PUT NEW PS ON STACK
948 004162 012746 004170              MOV      #64$,-(SP) ;:PUT NEW PC ON STACK
949 004166 000002              RTI                    ;: POP NEW PC AND PS
950 004170
951 004170 012706 001000      64$:
R0020: MOV      #1000,SP ;SETUP THE STACK
952 004174 005037 177546      CLR      LKS
953 004200 012777 000100 174712 I0020: MOV      #100,$GDADR ;SET THE "CLOSE" ADDRESS TO = 100

```

```

954 004206 032737 000100 177546 A0020: BIT #100,LKS ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
955 004214 001401 BEQ B0020
956 004216 104005 E0020: ERROR 5 ;IT AFFECTED "LKS" -- ERROR
957 004220 005037 177546 B0020: CLR LKS
958 004224 012737 004236 000004 MOV #T0021 ,4
959 004232 005077 174662 CLR @SGDADR ;CLEAR OUT THE "CLOSE" ADDRESS
960
961
962
963 .SBTTL LINE CLOCK REGISTER ADDRESSING TEST
964 :LINE CLOCK REGISTER ADDRESSING TEST
965 T0021: SCOPE
966 004236 000004 MOV #A0021,4 ;SETUP VECTOR IN CASE IT IS NONEXISTANT
967 004240 012737 004324 000004 MOV #340,6
968 004246 012737 000340 000006 CLR $GDDAT
969 004254 005037 001124 MOV #R0021,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
970 004260 012737 004306 001106 MOV #177746,$GDADR ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 7 SET
971 004266 012737 177746 001120 MOV PR7,-(SP) ;;PUT NEW PS ON STACK
972 004274 013746 000340 MOV #64$,-(SP) ;;PUT NEW PC ON STACK
973 004300 012746 004306 RTI ;;POP NEW PC AND PS
974 004304 000002
975 004306 012706 001000 64$:
976 004312 005037 177546 R0021: MOV #1000,SP ;SETUP THE STACK
977 004316 012777 000100 174574 CLR LKS
978 004324 032737 000100 177546 I0021: MOV #100,@SGDADR ;SET THE "CLOSE" ADDRESS TO = 100
979 004332 001401 A0021: BIT #100,LKS ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
980 004334 104005 E0021: ERROR 5 ;IT AFFECTED "LKS" -- ERROR
981 004336 005037 177546 B0021: CLR LKS
982 004342 012737 004354 000004 MOV #T0022 ,4
983 004350 005077 174544 CLR @SGDADR ;CLEAR OUT THE "CLOSE" ADDRESS
984
985
986
987 .SBTTL LINE CLOCK REGISTER HIGH BYTE TEST
988 :LINE CLOCK REGISTER HIGH BYTE TEST
989 :MAKE SURE THE LKS REGISTER LOW BYTE RESPONDS TO THE HIGH BYTES ADDRESS
990 T0022: SCOPE
991 004354 000004 JMP T0023
992 004356 000137 004470 MOV #TRAP0,@#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
993 004362 012737 006222 000004 MOV #340,@#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
994 004370 012737 000340 000006 MOV #100,$GDDAT
995 004376 012737 000100 001124 MOV #R0022,$LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
996 004404 012737 004432 001106 MOV #177547,$GDADR ;HIGH BYTE OF THE LKS REGISTER
997 004412 012737 177547 001120 MOV PR7,-(SP) ;;PUT NEW PS ON STACK
998 004420 013746 000340 MOV #64$,-(SP) ;;PUT NEW PC ON STACK
999 004424 012746 004432 RTI ;;POP NEW PC AND PS
1000 004430 000002
1001 004432 012706 001000 64$:
1002 004436 005037 177546 R0022: MOV #1000,SP ;SETUP THE STACK
1003 004442 112777 000100 174450 CLR LKS
1004 004450 032737 000100 177546 I0022: MOVB #100,@SGDADR ;SET THE HIGH BYTE ADDRESS TO = 100
1005 004456 001001 BIT #100,LKS ;MAKE SURE THAT "LKS" LOW BYTE WAS AFFECTED
1006 004460 104005 E0022: ERROR 5
1007 004462 005037 177546 A0022: CLR LKS ;SHOULD HAVE SET BIT 7. ERROR
1008 004466 000005 RESET
1009

```

```

1010
1011
1012
1013
1014 004470 000004
1015 004472 012737 006222 000004
1016 004500 012737 000340 000006
1017 004506 012737 000200 001124
1018 004514 012737 004522 001106
1019 004522 005037 177546
1020 004526 005003
1021 004530 105737 177546
1022 004534 100404
1023 004536 005203
1024 004540 001373
1025 004542 104001
1026 004544 000410
1027 004546
1028 004546 012737 000200 177546
1029 004554 023737 177546 001124
1030 004562 001401
1031 004564 104001
1032
1033
1034
1035
1036 004566 000004
1037 004570 012737 006222 000004
1038 004576 012737 000340 000006
1039 004604 012737 000200 001124
1040 004612 012737 004632 001106
1041 004620 012737 004672 000100
1042 004626 005037 177546
1043 004632
1044 004632 013746 000340
1045 004636 012746 004644
1046 004642 000002
1047 004644
1048 004644 012737 000300 177546
1049 004652 005227 000000
1050 004656 001375
1051 004660 000005
1052 004662 023737 001124 177546
1053 004670 001401
1054 004672 104001
1055
1056
1057
1058
1059
1060 004674 000004
1061 004676 012737 006222 000004
1062 004704 012737 000340 000006
1063 004712 012737 000300 001124
1064 004720 012737 004742 001106
1065 004726 012737 000340 000102

.SBTTL CLOCK FLAG BIT TEST
:CLOCK FLAG BIT TEST
T0023: SCOPE
MOV #TRAP0, J#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
MOV #340, J#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
MOV #200, $GDDAT
MOV #R0023, $LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
R0023: CLR LKS
CLR R3 ;INITIALIZE A COUNTER LOCATION
A0023: TSTB LKS ;IS THE CLOCK FLAG SET?
BMI B0023 ;IF SO, CONTINUE ON WITH THE TEST
INC R3 ;IF NOT INCREMENT THE COUNTER LOCATION
BNE A0023 ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
E0023: ERROR 1 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
BR T0024
B0023:
T0023: MOV #200, LKS ;MOVE A 1 INTO THE CLOCK FLAG BIT
CMP LKS, $GDDAT ;SHOULD NOT AFFECT THE FLAG BIT
BEQ T0024
E10023: ERROR 1 ;CLOCK FLAG DID NOT CLEAR

.SBTTL INTERRUPT TEST
T0024: SCOPE
MOV #TRAP0, J#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
MOV #340, J#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
MOV #200, $GDDAT ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
MOV #R0024, $LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
MOV #E0024, 100
CLR LKS ;ALLOW CLOCK INTERRUPTS
R0024: MOV PR7, -(SP) ;:PUT NEW PS ON STACK
MOV #64$, -(SP) ;:PUT NEW PC ON STACK
RTI ;: POP NEW PC AND PS
64$:
A0024: MOV #300, LKS
INC #0 ;:WAIT FOR 20+ MS
BNE A0024 ;:LOOP BACK IF NOT DONE WAITING
RESET ;:RESET SHOULD CLEAR INTERRUPT ENABLE
CMP $GDDAT, LKS ;:AND LEAVE THE CLOCK FLAG SET
BEQ T0025 ;:GO ON TO THE NEXT TEST IF IT DID
E0024: ERROR 1 ;:RESET SET INTERRUPT ENABLE OR CLEARED CLOCK FLAG

.SBTTL NO SACK TIMEOUT TEST
:NO SACK TIMEOUT TEST
T0025: SCOPE
MOV #TRAP0, J#4 ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
MOV #340, J#6 ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
MOV #300, $GDDAT ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
MOV #R0025, $LPADR ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
MOV #340, 102 ;NO INTERRUPTS AFTER THE FIRST ONE

```

```

1066 004734 012737 005024 000100      MOV      #C0025,100
1067 004742 005037 177546      R0025:  CLR      LKS
1068 004746 013746 000340      MOV      PR7,-(SP)      ;;PUT NEW PS ON STACK
1069 004752 012746 004760      MOV      #64$,-(SP)    ;;PUT NEW PC ON STACK
1070 004756 000002      RTI                          ;;POP NEW PC AND PS
1071 004760
1072 004760 005003      64$:   CLR      R3      ;INITIALIZE A COUNTER LOCATION
1073 004762 105737 177546      A0025: TSTB     LKS      ;IS THE CLOCK FLAG SET?
1074 004766 100404      BMI     B0025      ;IF SO, CONTINUE ON WITH THE TEST
1075 004770 005203      INC     R3      ;IF NOT INCREMENT THE COUNTER LOCATION
1076 004772 001373      BNE     A0025      ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
1077 004774 104001      E0025: ERROR 1      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS.
1078 004776 000417      BR      T0026
1079 005000      B0025:
1080 005000 005046      CLR     -(SP)      ;DROP CPU PRIORITY LEVEL
1081 005002 012746 005010      MOV     #1$,-(SP)  ;SET RETURN 'PC' FROM THE RTI
1082 005006 000002      RTI                          ;RETURN TO NEXT INSTRUCTION
1083 005010 012737 000100 177546 1$:   MOV     #100,LKS   ;ENABLE CLOCK INTERRUPTS
1084 005016 000001      WAIT                          ;THE ONLY WAY TO LEAVE HERE WITHOUT ERROR IS TO INTERRUPT
1085 005020 104006      E10025: ERROR 6      ;IF IT ERROR IS HERE ITS BECAUSE OF A "NO-SACK" TIMEOUT
1086 005022 000405      BR      T0026
1087 005024 022737 000300 177546 C0025: CMP     #300,LKS
1088 005032 001401      BEQ     T0026
1089 005034 104001      E20025: ERROR 1      ;FIND OUT WHAT THE INTERRUPT DID TO THE CLOCK STATUS REG
1090
1091
1092
1093
1094      .SBTTL  RESET TEST
1095      :RESET TEST
1096 005036 000004      T0026: SCOPE
1097 005040 012737 006222 000004      MOV     #TRAP0,2#4    ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
1098 005046 012737 000340 000006      MOV     #340,2#6     ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
1099 005054 012737 000200 001124      MOV     #200,$GDDAT  ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR.
1100 005062 012737 005116 001106      MOV     #R0026,$LPADR;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
1101 005070 013746 000340      MOV     PR7,-(SP)    ;;PUT NEW PS ON STACK
1102 005074 012746 005102      MOV     #64$,-(SP)  ;;PUT NEW PC ON STACK
1103 005102      RTI                          ;;POP NEW PC AND PS
1104 005102 012737 005154 000100 64$:   MOV     #E0026,100
1105 005110 012737 000140 000102      MOV     #140,102    ;SETUP STATUS FOR AFTER THE INTERRUPT
1106 005116 005037 177546      R0026: CLR     LKS
1107 005122 012706 001000      MOV     #1000,SP    ;SETUP THE STACK
1108 005126 012737 000100 177546      MOV     #100,LKS    ;SET INTERRUPT ENABLE BIT NOW
1109 005134 005227 000000      A0026: INC     #0    ;WAIT FOR CLOCK FLAG TO SET
1110 005140 001375      BNE     A0026
1111 005142 000005      I0026: RESET
1112 005144 023737 177546 001124      CMP     LKS,$GDDAT  ;RESET SHOULD NOT CLEAR THE FLAG
1113 005152 001401      BEQ     T0027      ;FIND OUT IF ID DIT DID OR NOT
1114 005154 104001      E0026: ERROR 1      ;RESET DID NOT INITIALIZE THE LKS WORD CORRECTLY
1115
1116
1117
1118      .SBTTL  CLOCK FLAG BIT TEST
1119      :CLOCK FLAG BIT TEST
1120      :MAKE SURE IT DOESNT CLEAR WHEN YOU TRY TO SET IT VIA A 'MOV' INSTRUCTION
1121 005156 000004      T0027: SCOPE

```

```

1122 005160 012737 006222 000004      MOV      #TRAP0,0#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
1123 005166 012737 000340 000006      MOV      #340,0#6       ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
1124 005174 012737 000200 001124      MOV      #200,$GDDAT    ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
1125 005202 012737 005222 001106      MOV      #R0027,$LPADR  ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
1126 005210 005037 000102          CLR      102
1127 005214 012737 005304 000100      MOV      #T0030 ,100
1128 005222 005037 177546      R0027:  CLR      LKS
1129 005226 013746 000340      MOV      PR7,-(SP)      ;;PUT NEW PS ON STACK
1130 005232 012746 005240      MOV      #64$,-(SP)    ;;PUT NEW PC ON STACK
1131 005236 000002          RTI                    ;;POP NEW PC AND PS
1132 005240          64$:
1133 005240 012706 001000      MOV      #1000,SP      ;SETUP THE STACK
1134 005244 005037 001234      CLR      WORD          ;SETUP A COUNTER LOCATION TO = 0
1135 005250 005237 001234      A0027:  INC      WORD
1136 005254 001375          BNE      A0027         ;WASTE TIME LOOPING UNTIL THE COUNTER REACHES 0
1137 005256 012737 000300 177546      MOV      #300,LKS      ;SET INTERRUPT ENABLE AND TRY TO SET THE CLOCK FLAG
1138 005264 012737 000200 177546      MOV      #200,LKS      ;TRY TO SET IT AGAIN
1139 005272 023737 177546 001124      CMP      LKS,$GDDAT    ;DID THE CLOCK FLAG STAY SET?
1140 005300 001401          BEQ      T0030        ;IF NOT GO ON TO THE NEXT TEST
1141 005302 104001      E0027:  ERROR 1       ;ERROR - MOVED A '1' INTO THE CLOCK FLAG BIT AND IT STAY
1142
1143
1144
1145          .SBTTL  CLOCK FLAG AFTER INTERRUPT TEST
1146          :SEE IF AN INTERRUPT CLEARS THE CLOCK FLAG
1147          T0030:  SCOPE
1148 005304 000004          MOV      #TRAP0,0#4    ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
1149 005306 012737 006222 000004      MOV      #340,0#6     ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
1150 005314 012737 000340 000006      MOV      #340,0#6     ;NO CLOCK INTERRUPTS BEFORE WE ARE READY
1151 005322 005037 177546          CLR      LKS          ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
1152 005326 012737 000100 001124      MOV      #100,$GDDAT  ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
1153 005334 012737 005342 001106      MOV      #R0030,$LPADR ;SETUP CLOCK INTERRUPT VECTOR
1154 005342 012737 005410 000100      R0030:  MOV      #A0030,100 ;PRIORITY LEVEL WILL ALLOW FURTHER INTERRUPTS
1155 005350 005037 000102          CLR      102         ;DROP CPU PRIORITY LEVEL
1156 005354 005046          CLR      -(SP)       ;SET RETURN 'PC' FROM THE RTI
1157 005362 000002          RTI                    ;RETURN TO NEXT INSTRUCTION
1158 005364 012706 001000      1$:  MOV      #1000,SP    ;SETUP THE STACK
1159 005370 005037 177546          CLR      LKS
1160 005374 105037 001234          CLR      WORD
1161 005400 012737 000100 177546      MOV      #100,LKS     ;CLEAR OUT A COUNTER LOCATION
1162 005406 000001          WAIT                  ;ENABLE CLOCK INTERRUPTS NOW
1163 005410 012737 005442 000100      A0030:  MOV      #E0030,100 ;WAIT FOR AN INTERRUPT
1164 005416 005046          CLR      -(SP)       ;ERROR IF WE INTERRUPT AGAIN
1165 005420 012746 005426          MOV      #B0030,-(SP) ;DROP CPU PRIORITY LEVEL
1166 005424 000002          RTI                    ;SET RETURN 'PC' FROM THE RTI
1167 005426 105237 001234      B0030:  INCB      WORD   ;RETURN TO NEXT INSTRUCTION
1168 005432 001375          BNE      B0030       ;DO A NOTHING LOOP FOR A VERY SHORT PERIOD OF TIME
1169 005434 005037 177546          CLR      LKS         ;WE SHOULD INCREMENT TO 0 LONG BEFORE AN INTERRUPT COMES
1170 005440 000401          BR       T0031
1171 005442 104001      E0030:  ERROR 1       ;INTERRUPT DID NOT CLEAR THE CLOCK FLAG
1172
1173
1174
1175          .SBTTL  NO INTERRUPT AT PRIORITY 7 TEST
1176          :TEST THAT CLOCK WILL NOT INTERRUPT WITH PROCESSR AT PRIORITY 7
1177 005444 000004      T0031:  SCOPE

```

```

1178 005446 012737 006222 000004      MOV      #TRAP0, @#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
1179 005454 012737 000340 000006      MOV      #340, @#6       ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
1180 005462 012737 005470 001106      MOV      #R0031, $LPADR  ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
1181 005470 005037 177546      R0031:  CLR      LKS
1182 005474 005003              CLR      R3              ;INITIALIZE A COUNTER LOCATION
1183 005476 105737 177546      A0031:  TSTB   LKS        ;IS THE CLOCK FLAG SET?
1184 005502 100404              BMI     B0031           ;IF SO, CONTINUE ON WITH THE TEST
1185 005504 005203              INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
1186 005506 001373              BNE    A0031           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
1187 005510 104001      E0031:  ERROR 1          ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
1188 005512 000431              BR     T0032
1189 005514              B0031:
1190 005514 012706 001000      MOV     #1000, SP      ;INITIALIZE THE STACK POINTER
1191 005520 013746 000340      MOV     PR7, -(SP)    ;PUT NEW PS ON STACK
1192 005524 012746 005532      MOV     #64$, -(SP)  ;PUT NEW PC ON STACK
1193 005530 000002              RTI     ;POP NEW PC AND PS
1194 005532              64$:
1195 005532 012737 005510 000100      MOV     #E0031, 100   ;SET UP VECTOR RETURN
1196 005540 012737 000100 177546      MOV     #100, LKS     ;ENABLE INTERRUPT
1197 005546 005003              CLR     R3            ;INITIALIZE A COUNTER LOCATION
1198 005550 105737 177546      C0031:  TSTB   LKS        ;IS THE CLOCK FLAG SET?
1199 005554 100404              BMI     D0031           ;IF SO, CONTINUE ON WITH THE TEST
1200 005556 005203              INC     R3            ;IF NOT INCREMENT THE COUNTER LOCATION
1201 005560 001373              BNE    C0031           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
1202 005562 104001      E10031: ERROR 1          ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
1203 005564 000404              BR     T0032
1204 005566              D0031:
1205 005566 000240              NOP
1206 005570 000240              NOP                    ;GIVE CLOCK EXTRA TIME TO INTERRUPT
1207 005572 000401              BR     T0032
1208 005574 104003      E20031: ERROR 3          ;ERROR, CLOCK INTERRUPTED WITHOUT HAVING PRIORITY
1209
1210
1211
1212      .SBTTL  CC PUSH TEST FOR CLOCK INTERRUPTS
1213      ;TEST THAT CLOCK INTERRUPT PUSHES CONDITION CODES ONTO STACK
1214 005576 000004      T0032:  SCOPE
1215 005600 012737 006222 000004      MOV     #TRAP0, @#4    ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
1216 005606 012737 000340 000006      MOV     #340, @#6     ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
1217 005614 012737 005622 001106      MOV     #R0032, $LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
1218 005622 005037 177546      R0032:  CLR     LKS
1219 005626 005003              CLR     R3            ;INITIALIZE A COUNTER LOCATION
1220 005630 105737 177546      A0032:  TSTB   LKS        ;IS THE CLOCK FLAG SET?
1221 005634 100404              BMI     B0032           ;IF SO, CONTINUE ON WITH THE TEST
1222 005636 005203              INC     R3            ;IF NOT INCREMENT THE COUNTER LOCATION
1223 005640 001373              BNE    A0032           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
1224 005642 104001      E0032:  ERROR 1          ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
1225 005644 000433              BR     T0033
1226 005646              B0032:
1227 005646 012706 001000      MOV     #1000, SP      ;INITIALIZE THE STACK POINTER
1228 005652 005037 000776      CLR     BUF1
1229 005656 005037 000774      CLR     BUF2
1230 005662 012737 005712 000100      MOV     #C0032, 100   ;SET UP VECTOR RETURN
1231 005670 012737 000100 177546      MOV     #100, LKS     ;ENABLE INTERRUPT
1232 005676 005046              CLR     -(SP)         ;DROP CPU PRIORITY LEVEL
1233 005700 012746 005706      MOV     #1$, -(SP)    ;SET RETURN 'PC' FROM THE RTI

```

```

1234 005704 000002          RTI          ;RETURN TO NEXT INSTRUCTION
1235 005706 000277          1$:      +SEC!SEV!SEZ!SEN      ;SET ALL CONDITION CODES
1236 005710 000001          WAIT          ;WAIT FOR INTERRUPT
1237 005712 022737 000017 000776 C0032:  CMP      #17,BUF1
1238 005720 001405          BEQ      T0033
1239 005722 012737 000017 001124      MOV      #17,$GDDAT
1240 005730 104007          ERROR 7      ;ERROR DID NOT PUSH CORRECT PS ONTO STACK
1241 005732 000721          BR      T0032
1242
1243
1244
1245          .SBTTL  PC PUSH TEST FOR CLOCK INTERRUPTS
1246          ;TEST THAT CLOCK INTERRUPT PUSHES THE PROGRAM COUNTER ONTO STACK
1247 005734 000004          T0033:  SCOPE
1248 005736 012737 006222 000004      MOV      #TRAP0,2#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
1249 005744 012737 000340 000006      MOV      #340,2#6      ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
1250 005752 012737 005760 001106      MOV      #R0033,$LPADR ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
1251 005760 005037 177546          R0033:  CLR      LKS
1252 005764 005003          CLR      R3      ;INITIALIZE A COUNTER LOCATION
1253 005766 105737 177546          A0033:  TSTB   LKS      ;IS THE CLOCK FLAG SET?
1254 005772 100404          BMI      B0033      ;IF SO, CONTINUE ON WITH THE TEST
1255 005774 005203          INC      R3      ;IF NOT INCREMENT THE COUNTER LOCATION
1256 005776 001373          BNE      A0033      ;AND GO TEST THE CLOCK FLAG AGAIN, UNLESS...
1257 006000 104001          E0033:  ERROR 1      ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
1258 006002 000432          BR      T0034
1259 006004          B0033:
1260 006004 012706 001000      MOV      #1000,SP      ;INITIALIZE THE STACK POINTER
1261 006010 005037 000776      CLR      BUF1
1262 006014 005037 000774      CLR      BUF2
1263 006020 012737 006050 000100      MOV      #C0033,100    ;SET UP VECTOR RETURN
1264 006026 012737 000100 177546      MOV      #100,LKS      ;ENABLE INTERRUPT
1265 006034 005046          CLR      -(SP)      ;DROP CPU PRIORITY LEVEL
1266 006036 012746 006044      MOV      #1$,-(SP)    ;SET RETURN 'PC' FROM THE RTI
1267 006042 000002          RTI          ;RETURN TO NEXT INSTRUCTION
1268 006044 000277          1$:      +SEC!SEV!SEZ!SEN      ;SET ALL CONDITION CODES
1269 006046 000001          WAIT          ;WAIT FOR INTERRUPT
1270 006050 022737 006050 000774 C0033:  CMP      #C0033,BUF2
1271 006056 001404          BEQ      T0034
1272 006060 012737 006050 001124 E10033: MOV      #C0033,$GDDAT
1273 006066 104010          ERROR 10      ;ERROR, DID NOT PUSH CORRECT PC ONTO STACK
1274
1275
1276
1277          .SBTTL  END OF PASS INDICATING
1278 006070 000004          T0034:  SCOPE
1279 006072 005037 177546          CLR      LKS      ;TURN THE CLOCK OFF
1280 006076 000005          RESET      ;TURN EVERYTHING OFF
1281
1282          ;*****
1283
1284          .SBTTL  END OF PASS ROUTINE
1285
1286          ;*INCREMENT THE PASS NUMBER ($PASS)
1287          ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
1288          ;*IF THERES A MONITOR GO TO IT
1289          ;*IF THERE ISN'T JUMP TO T0001

```

```

1290
1291 006100          $EOP:
1292 006100 000004          SCOPE
1293 006102 005037 001102    CLR $STNM          ;; ZERO THE TEST NUMBER
1294 006106 005037 001220    CLR $TIMES        ;; ZERO THE NUMBER OF ITERATIONS
1295 006112 005237 001100    INC $PASS         ;; INCREMENT THE PASS NUMBER
1296 006116 042737 100000 001100 BIC #100000,$PASS ;; DON'T ALLOW A NEG. NUMBER
1297 006124 005327          DEC (PC)+          ;; LOOP?
1298 006126 000001          $EOPCT: .WORD 1
1299 006130 003022          BGT $DOAGN         ;; YES
1300 006132 012737          MOV (PC)+,a(PC)+  ;; RESTORE COUNTER
1301 006134 000001          $ENDCT: .WORD 1
1302 006136 006126          $EOPCT
1303 006140 104400 006202    TYPE $ENDMG      ;; TYPE "END PASS #"
1304 006144 013746 001100    MOV $PASS,-(SP)  ;; SAVE $PASS FOR TYPEOUT
1305 006150 104404          TYPDS           ;; GO TYPE--DECIMAL ASCII WITH SIGN
1306 006152 104400 006217    TYPE , $ENULL   ;; TYPE A NULL CHARACTER
1307 006156          $GET42:
1308
1309 006156 013700 000042    MOV a#42,R0     ;; GET MONITOR ADDRESS
1310 006162 001405          BEQ $DOAGN      ;; BRANCH IF NO MONITOR
1311 006164 000005          RESET        ;; CLEAR THE WORLD
1312 006166 004710          $ENDAD: JSR PC,(R0) ;; GO TO MONITOR
1313 006170 000240          NOP           ;; SAVE ROOM
1314 006172 000240          NOP           ;; FOR
1315 006174 000240          NOP           ;; ACT11
1316 006176          $DOAGN:
1317 006176 000137 001672    JMP a#T0001     ;; RETURN
1318 006202 005015 047105 020104 $ENDMG: .ASCIZ <15><12>/END PASS #/
1319 006210 040520 051523 021440
1320 006216 000
1321 006217 377 377 000 $ENULL: .BYTE -1,-1,0 ;; NULL CHARACTER STRING
1322 006222 005046          TRAPO: CLR -(SP)
1323 006224 004737 006546    JSR PC,$TYPE   ;; PRINTOUT "TRAPPED TO 4 FROM "
1324 006230 010333          TRPMES        ;; ADDRESS OF THE MESSAGE
1325 006232 011646          MOV (SP),-(SP) ;; GET THE ADDRESS WHERE THE TRAP OCCURED
1326 006234 162716 000002    SUB #2,(SP)    ;; MAKE IT RIGHT
1327 006240 104401          TYPDC         ;; TYPE OUT ADDRESS IN OCTAL
1328 006242 104400 001231    TYPE , $CRLF   ;; PRINTOUT A CARRIAGE RETURN-LINE FEED
1329 006246 005046          CLR -(SP)
1330 006250 004737 006546    JSR PC,$TYPE   ;; PRINTOUT RESTARTING MESSAGE
1331 006254 010374          TRPM2S        ;; ADDRESS OF RESTART MESSAGE
1332 006256 000240          NOP
1333 006260 000240          NOP
1334 006262 000240          NOP
1335 006264 000240          NOP
1336 006266 000005          RESET
1337 006270 000137 001450    JMP START
1338 *****
1339
1340 .SBTTL SCOPE HANDLER ROUTINE
1341
1342 ;; *THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
1343 ;; *AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG. (DISPLAY<7:0>)
1344 ;; *AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
1345 ;; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:

```

```

1346 ;*SW14=1 LOOP ON TEST
1347 ;*SW11=1 INHIBIT ITERATIONS
1348 ;*SW09=1 LOOP ON ERROR
1349 ;*SW08=1 LOOP ON TEST IN SWR<7:0>
1350 ;*CALL
1351 ;* SCOPE ;;SCOPE=10T
1352
1353 $SCOPE:
1354 006274 000240 040000 172632 1$: NOP
1355 006276 032777 040000 172632 1$: BIT #BIT14,$SWR ;;LOOP ON PRESENT TEST?
1356 006304 001111 040000 172632 1$: BNE $OVER ;;YES IF SW14=1
1357 ;*****START OF CODE FOR THE XOR TESTER*****
1358 006306 000416 040000 172632 $XTSTR: BR 6$ ;;IF RUNNING ON THE "XOR" TESTER CHANGE
1359 ; THIS INSTRUCTION TO A "NOP" (NOP=240)
1360 006310 013746 000004 000004 MOV $ERRVEC,-(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
1361 006314 012737 006334 000004 MOV #5,$ERRVEC ;;SET FOR TIMEOUT
1362 006322 005737 177060 000004 TST $177060 ;;TIME OUT ON X?
1363 006326 012637 000004 MOV (SP)+,$ERRVEC ;;RESTORE THE ERROR VECTOR
1364 006332 000453 000004 BR $SVLAD ;;GO TO THE NEXT TEST
1365 006334 022626 000004 5$: CMP (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
1366 006336 012637 000004 MOV (SP)+,$ERRVEC ;;RESTORE THE ERROR VECTOR
1367 006342 000423 000004 BR 7$ ;;LOOP ON THE PRESENT TEST.
1368 006344 032777 000400 172564 6$;*****END OF CODE FOR THE XOR TESTER*****
1369 006344 032777 000400 172564 6$: BIT #BIT08,$SWR ;;LOOP ON SPEC. TEST?
1370 006352 001404 172556 001102 6$: BEQ 2$ ;;BR IF NO
1371 006354 127737 172556 001102 6$: CMPB $SWR,$STNM ;;ON THE RIGHT TEST? SWR<7:0>
1372 006362 001462 172556 001102 6$: BEQ $OVER ;;BR IF YES
1373 006364 105737 001103 001103 2$: TSTB $ERFLG ;;HAS AN ERROR OCCURRED?
1374 006370 001421 001103 001103 2$: BEQ 3$ ;;BR IF NO
1375 006372 123737 001115 001103 2$: CMPB $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
1376 006400 101015 001103 001103 2$: BHI 3$ ;;BR IF NO
1377 006402 032777 001000 172526 2$: BIT #BIT09,$SWR ;;LOOP ON ERROR?
1378 006410 001404 001000 172526 2$: BEQ 4$ ;;BR IF NO
1379 006412 013737 001110 001106 7$: MOV $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
1380 006420 000443 001110 001106 7$: BR $OVER
1381 006422 105037 001103 001106 7$: CLRB $ERFLG ;;ZERO THE ERROR FLAG
1382 006426 005037 001220 001106 7$: CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
1383 006432 000415 001220 001106 7$: BR 1$ ;;ESCAPE TO THE NEXT TEST
1384 006434 032777 004000 172474 3$: BIT #BIT11,$SWR ;;INHIBIT ITERATIONS?
1385 006442 001011 004000 172474 3$: BNE 1$ ;;BR IF YES
1386 006444 005737 001100 004000 3$: TST $PASS ;;IF FIRST PASS OF PROGRAM
1387 006450 001406 001100 004000 3$: BEQ 1$ ;;INHIBIT ITERATIONS
1388 006452 005237 001104 004000 3$: INC $ICNT ;;INCREMENT ITERATION COUNT
1389 006456 023737 001220 001104 3$: CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
1390 006464 002021 001220 001104 3$: BGE $OVER ;;BR IF MORE ITERATION REQUIRED
1391 006466 012737 000001 001104 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
1392 006474 013737 006544 001220 1$: MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
1393 006502 105237 001102 001220 $SVLAD: INCB $STNM ;;COUNT TEST NUMBERS
1394 006506 011637 001106 001220 $SVLAD: MOV (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
1395 006512 011637 001110 001220 $SVLAD: MOV (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
1396 006516 005037 001222 001220 $SVLAD: CLR $ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
1397 006522 112737 000001 001115 $OVER: MOVB #1,$ERMAX ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
1398 006530 013777 001102 172402 $OVER: MOV $STNM,$DISPLAY ;;DISPLAY TEST NUMBER
1399 006536 013716 001106 172402 $OVER: MOV $LPADR,(SP) ;;FUDGE RETURN ADDRESS
1400 006542 000002 001106 172402 $OVER: RTI ;;FIXES PS
1401 006544 000010 001106 172402 $OVER: SMXCNT: 10 ;;MAX. NUMBER OF ITERATIONS

```

14200
14201
14202
14203
14204
14205
14206
14207
14208
14209
14210
14211
14212
14213
14214
14215
14216
14217
14218
14219
14220
14221
14222
14223
14224
14225
14226
14227
14228
14229
14230
14231
14232
14233
14234
14235
14236
14237
14238
14239
14240
14241
14242
14243
14244
14245
14246
14247
14248
14249
14250
14251
14252
14253
14254
14255
14256
14257

.SBTTL TYPE ROUTINE

;;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
 ;;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 ;;NOTE1: \$NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 ;;NOTE2: \$FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 ;;NOTE3: \$FILLC CONTAINS THE CHARACTER TO FILL AFTER.

;;CALL:
 ;;1) USING A TRAP INSTRUCTION
 ;; TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 ;;OR
 ;; TYPE
 ;; MESADR

```

006546 105737 001155 $TYPE: TSTB $TFPLG ;; IS THERE A TERMINAL?
006552 100002 BPL 1$ ;; BR IF YES
006554 000000 HALT ;; HALT HERE IF NO TERMINAL
006556 000407 BR 3$ ;; LEAVE
006560 010046 1$: MOV RO, -(SP) ;; SAVE RO
006562 017500 000002 MOV #2(SP), RO ;; GET ADDRESS OF ASCIZ STRING
006566 112046 2$: MOVB (RO)+, -(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
006570 001005 BNE 4$ ;; BR IF IT ISN'T THE TERMINATOR
006572 005726 TST (SP)+ ;; IF TERMINATOR POP IT OFF THE STACK
006574 012600 60$: MOV (SP)+, RO ;; RESTORE RO
006576 062716 3$: ADD #2, (SP) ;; ADJUST RETURN PC
006602 000002 RTI ;; RETURN
006604 122716 4$: CMPB #HT, (SP) ;; BRANCH IF <HT>
006610 001426 BEQ 8$
006612 122716 000200 CMPB #CRLF, (SP) ;; BRANCH IF NOT <CRLF>
006616 001004 BNE 5$
006620 005726 TST (SP)+ ;; POP <CR><LF> EQUIV
006622 104400 TYPE ;; TYPE A CR AND LF
006624 00123: SCRLF
006626 00075: BR 2$ ;; GET NEXT CHARACTER
006630 004737 006712 5$: JSR PC, $TYPEC ;; GO TYPE THIS CHARACTER
006634 123726 001154 6$: CMPB $FILLC, (SP)+ ;; IS IT TIME FOR FILLER CHARS.?
006640 001352 BNE 2$ ;; IF NO GO GET NEXT CHAR.
006642 013746 001152 MOV $NULL, -(SP) ;; GET # OF FILLER CHARS. NEEDED
;; AND THE NULL CHAR.
006646 105366 000001 7$: DECB 1(SP) ;; DOES A NULL NEED TO BE TYPED?
006652 002770 BLT 6$ ;; BR IF NO--GO POP THE NULL OFF OF STACK
006654 004737 006: JSR PC, $TYPEC ;; GO TYPE A NULL
006660 105337 006: DECB $CHARCNT ;; DO NOT COUNT AS A COUNT
006664 000770 BR 7$ ;; LOOP
  
```

;;HORIZONTAL TAB PROCESSOR

```

006666 112716 000040 8$: MOVB #40, (SP) ;; REPLACE TAB WITH SPACE
006672 004737 006712 9$: JSR PC, $TYPEC ;; TYPE A SPACE
006676 132737 000007 006756 BITB #7, $CHARCNT ;; BRANCH IF NOT AT
006704 001372 BNE 9$ ;; TAB STOP
006706 005726 TST (SP)+ ;; POP SPACE OFF STACK
  
```

```

1458 006710 000726          BH      2$      ;;GET NEXT CHARACTER
1459 006712 105777 172230  STYPEC: TSTB   JSTPS   ;;WAIT UNTIL PRINTER IS READY
1460 006716 100375          BPL     STYPEC
1461 006720 116677 000002 172222  MOVB   2(SP),JSTPB  ;;LOAD CHAR TO BE TYPED INTO DATA REG.
1462 006726 122766 000015 000002  CMPB   #15,2(SP)   ;;BRANCH IF
1463 006734 001003          BNE     !$      ;;NOT <CR>
1464 006736 105037 006756  CLRB   $CHARCNT
1465 006742 000406          BR     STYPEX   ;;EXIT
1466 006744 122766 000012 000002  !$:  CMPB   #12,2(SP)  ;;BRANCH IF
1467 006752 002002          BGE     STYPEX   ;;<LF>
1468 006754 105227          INCB   (PC)+    ;;INC SPACE
1469 006756 000000          $CHARCNT: .WORD 0  ;;COUNT
1470 006760 000207          STYPEX: RTS     PC
1471          ;;
1472          THT=11
1473          TCRLF=200
1474
1475          ;*****
1476
1477          .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
1478
1479          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
1480          ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
1481          ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
1482          ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
1483          ;*REPLACED WITH SPACES.
1484          ;*CALL:
1485          ;*   MOV     NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
1486          ;*   TYPDS  ;;GO TO THE ROUTINE
1487
1488          STYPDS:
1489          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
1490          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
1491          MOV     R2,-(SP)          ;;PUSH R2 ON STACK
1492          MOV     R3,-(SP)          ;;PUSH R3 ON STACK
1493          MOV     R5,-(SP)          ;;PUSH R5 ON STACK
1494          MOV     #20200,-(SP)     ;;SET BLANK SWITCH AND SIGN
1495          MOV     20(SP),R5        ;;GET THE INPUT NUMBER
1496          BPL     !$              ;;BR IF INPUT IS POS.
1497          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
1498          MOVB   #'-,1(SP)        ;;MAKE THE ASCII NUMBER NEG.
1499          CLR     R0              ;;ZERO THE CONSTANTS INDEX
1500          MOV     #SDBLK,R3       ;;SETUP THE OUTPUT POINTER
1501          MOVB   #'',(R3)+        ;;SET THE FIRST CHARACTER TO A BLANK
1502          CLR     R2              ;;CLEAR THE BCD NUMBER
1503          MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
1504          SJB    R1,R5            ;;FORM THIS BCD DIGIT
1505          BLT    4$              ;;BR IF DONE
1506          INC    R2              ;;INCREASE THE BCD DIGIT BY 1
1507          BR     3$
1508          4$:  ADD     R1,R5        ;;ADD BACK THE CONSTANT
1509          TST    R2              ;;CHECK IF BCD DIGIT=0
1510          BNE    5$              ;;FALL THROUGH IF 0
1511          TSTB  (SP)             ;;STILL DOING LEADING 0'S?
1512          BMI    7$              ;;BR IF YES
1513          5$:  ASLB   (SP)        ;;MSD?

```

```

1514 007062 103003          BCC      6$          ;;BR IF NO
1515 007064 116663 000001 177777  MOVB   1(SP),-1(R3)  ;;YES--SET THE SIGN
1516 007072 052702 000060 6$:     BIS    #'0,R2    ;;MAKE THE BCD DIGIT ASCII
1517 007076 052702 000040 7$:     BIS    #' ,R2    ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
1518 007102 110223          MOVB   R2,(R3)+     ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
1519 007104 005720          TST    (R0)+       ;;JUST INCREMENTING
1520 007106 020027 000010  CMP    R0,#10     ;;CHECK THE TABLE INDEX
1521 007112 002746          BLT    2$         ;;GO DO THE NEXT DIGIT
1522 007114 003002          SGT    9$         ;;GO TO EXIT
1523 007116 010502          MOV    R5,R2      ;;GET THE LSD
1524 007120 000764          BR     6$         ;;GO CHANGE TO ASCII
1525 007122 105726          8$:     TSTB   (SP)+  ;;WAS THE LSD THE FIRST NON-ZERO?
1526 007124 100003          SPL    9$         ;;BR IF NO
1527 007126 116663 177777 177776 9$:     MOVB   -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
1528 007134 105013          CLRB  (R3)       ;;SET THE TERMINATOR
1529 007136 012605          MOV    (SP)+,R5   ;;POP STACK INTO R5
1530 007140 012603          MOV    (SP)+,R3   ;;POP STACK INTO R3
1531 007142 012602          MOV    (SP)+,R2   ;;POP STACK INTO R2
1532 007144 012601          MOV    (SP)+,R1   ;;POP STACK INTO R1
1533 007146 012600          MOV    (SP)+,R0   ;;POP STACK INTO R0
1534 007150 104400 007176  TYPE   $DBLK      ;;NOW TYPE THE NUMBER
1535 007154 016666 000002 000004  MOV    2(SP),4(SP) ;;ADJUST THE STACK
1536 007162 012616          MOV    (SP)+,(SP)
1537 007164 000002          RTI                    ;;RETURN TO USER
1538 007166 023420          SDTBL: 10000.
1539 007170 001750          1000.
1540 007172 000144          100.
1541 007174 000012          10.
1542 007176 000004          $DBLK: .BLKW 4
1543 *****
1544 *****
1545 .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
1546
1547 ;;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
1548 ;;OCTAL (ASCII) NUMBER AND TYPE IT.
1549 ;;*STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
1550 ;;*CALL:
1551 ;;      MOV    NUM,-(SP)      ;;NUMBER TO BE TYPED
1552 ;;      TYPOS      ;;CALL FOR TYPEOUT
1553 ;;      .BYTE    N          ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
1554 ;;      .BYTE    M          ;;M=1 OR 0
1555 ;;                                ;;1=TYPE LEADING ZEROS
1556 ;;                                ;;0=SUPPRESS LEADING ZEROS
1557
1558 ;;*STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
1559 ;;*STYPOS OR STYPOC
1560 ;;*CALL:
1561 ;;      MOV    NUM,-(SP)      ;;NUMBER TO BE TYPED
1562 ;;      TYPON      ;;CALL FOR TYPEOUT
1563
1564 ;;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
1565 ;;*CALL:
1566 ;;      MOV    NUM,-(SP)      ;;NUMBER TO BE TYPED
1567 ;;      TYPOC      ;;CALL FOR TYPEOUT
1568
1569 007206 017646 000000  STYPOS: MOV    2(SP),-(SP)  ;;PICKUP THE MODE

```

```

1570 007212 116637 000001 007431      MOVB      1(SP),SOFILL      ;;LOAD ZERO FILL SWITCH
1571 007220 112637 007433      MOVB      (SP)+,SOMODE+1  ;;NUMBER OF DIGITS TO TYPE
1572 007224 062716 000002      ADD       #2,(SP)        ;;ADJUST RETURN ADDRESS
1573 007230 000406      BR        $TYPON
1574 007232 112737 000001 007431 $TYPON: MOVB      #1,SOFILL      ;;SET THE ZERO FILL SWITCH
1575 007240 112737 000006 007433      MOVB      #6,SOMODE+1    ;;SET FOR SIX(6) DIGITS
1576 007246 112737 000005 007430 $TYPON: MOVB      #5,SOCNT   ;;SET THE ITERATION COUNT
1577 007254 010346      MOV       R3,-(SP)       ;;SAVE R3
1578 007256 010446      MOV       R4,-(SP)       ;;SAVE R4
1579 007260 010546      MOV       R5,-(SP)       ;;SAVE R5
1580 007262 113704 007433      MOVB      $SOMODE+1,R4   ;;GET THE NUMBER OF DIGITS TO TYPE
1581 007266 005404      NEG       R4
1582 007270 062704 000006      ADD       #6,R4          ;;SUBTRACT IT FOR MAX. ALLOWED
1583 007274 110437 007432      MOVB      R4,$SOMODE     ;;SAVE IT FOR USE
1584 007300 113704 007431      MOVB      $SOFILL,R4     ;;GET THE ZERO FILL SWITCH
1585 007304 016605 000012      MOV       12(SP),R5      ;;PICKUP THE INPUT NUMBER
1586 007310 005003      CLR       R3            ;;CLEAR THE OUTPUT WORD
1587 007312 006105      1$:      ROL       R5            ;;ROTATE MSB INTO "C"
1588 007314 000404      BR        3$
1589 007316 006105      2$:      ROL       R5            ;;GO DO MSB
1590 007320 006105      ROL       R5            ;;FORM THIS DIGIT
1591 007322 006105      ROL       R5
1592 007324 010503      MOV       R5,R3
1593 007326 006103      3$:      ROL       R3            ;;GET LSB OF THIS DIGIT
1594 007330 105337 007432      DECB      $SOMODE        ;;TYPE THIS DIGIT?
1595 007334 100016      BPL       7$            ;;BR IF NO
1596 007336 042703 177770      BIC       #177770,R3     ;;GET RID OF JUNK
1597 007342 001002      BNE       4$            ;;TEST FOR 0
1598 007344 005704      TST      R4            ;;SUPPRESS THIS 0?
1599 007346 001403      BEQ      5$            ;;BR IF YES
1600 007350 005204      4$:      INC       R4            ;;DON'T SUPPRESS ANYMORE 0'S
1601 007352 052703 000060      BIS      #'0,R3        ;;MAKE THIS DIGIT ASCII
1602 007356 052703 000040      5$:      BIS      #' ,R3        ;;MAKE ASCII IF NOT ALREADY
1603 007362 110337 007426      MOVB      R3,8$         ;;SAVE FOR TYPING
1604 007366 104400 007426      TYPE     8$            ;;GO TYPE THIS DIGIT
1605 007372 105337 007430      7$:      DECB      $SOCNT        ;;COUNT BY 1
1606 007376 003347      BGT      2$            ;;BR IF MORE TO DO
1607 007400 002402      BLT      6$            ;;BR IF DONE
1608 007402 005204      INC      R4            ;;INSURE LAST DIGIT ISN'T A BLANK
1609 007404 000744      BR        2$           ;;GO DO THE LAST DIGIT
1610 007406 012605      6$:      MOV      (SP)+,R5       ;;RESTORE R5
1611 007410 012604      MOV      (SP)+,R4       ;;RESTORE R4
1612 007412 012603      MOV      (SP)+,R3       ;;RESTORE R3
1613 007414 016666 000002 000004      MOV      2(SP),4(SP)    ;;SET THE STACK FOR RETURNING
1614 007422 012616      MOV      (SP)+,(SP)
1615 007424 000002      RTI
1616 007426      000      8$:      .BYTE    0            ;;RETURN
1617 007427      000      .BYTE    0            ;;STORAGE FOR ASCII DIGIT
1618 007430      000      $SOCNT: .BYTE    0            ;;TERMINATOR FOR TYPE ROUTINE
1619 007431      000      $SOFILL: .BYTE    0            ;;OCTAL DIGIT COUNTER
1620 007432 000000      $SOMODE: .WORD    0            ;;ZERO FILL SWITCH

```

.SBTTL ERROR MESSAGE TIMEOUT ROUTINE

;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" (\$ITEMB) TO DETERMINE WHICH

```

1626      ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" (SERRTB),
1627      ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
1628
1629      SERRTYP:
1630      007434 104400 001231      TYPE      ,SCLRF      ;; "CARRIAGE RETURN" & "LINE FEED"
1631      007434 010046      MOV      RO,-(SP)      ;; SAVE RO
1632      007442 005000      CLR      RO      ;; PICKUP THE ITEM INDEX
1633      007444 153700 001114      BISB     2*SITEMB,RO
1634      007450 001004      BNE     1$
1635      ;* IF ITEM NUMBER IS ZERO, JUST
1636      007452 013746 001116      MOV      SERRPC,-(SP) ;; TYPE THE PC OF THE ERROR
1637      ;* SAVE SERRPC FOR TYPEOUT
1638      007456 104401      TYP0C   ;* ERROR ADDRESS
1639      007460 000426      BR      6$           ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
1640      007462 005300      1$:     DEC      RO           ;; GET OUT
1641      007464 006300      ASL     RO           ;; ADJUST THE INDEX SO THAT IT WILL
1642      007466 006300      ASL     RO           ;; WORK FOR THE ERROR TABLE
1643      007470 006300      ASL     RO
1644      007472 062700 001236      ADD     #SERRTB,RO   ;; FORM TABLE POINTER
1645      007476 012037 007506      MOV     (RO)+,2$     ;; PICKUP "ERROR MESSAGE" POINTER
1646      007502 001404      BEQ     3$           ;; SKIP TYPEOUT IF NO POINTER
1647      007504 104400      TYPE   ;* TYPE THE "ERROR MESSAGE"
1648      007506 000000      2$:     .WORD 0           ;; "ERROR MESSAGE" POINTER GOES HERE
1649      007510 104400 001231      TYPE   ,SCLRF      ;; "CARRIAGE RETURN" & "LINE FEED"
1650      007514 012037 007524      3$:     MOV     (RO)+,4$     ;; PICKUP "DATA HEADER" POINTER
1651      007520 001404      BEQ     5$           ;; SKIP TYPEOUT IF 0
1652      007522 104400      TYPE   ;* TYPE THE "DATA HEADER"
1653      007524 000000      4$:     .WORD 0           ;; "DATA HEADER" POINTER GOES HERE
1654      007526 104400 001231      TYPE   ,SCLRF      ;; "CARRIAGE RETURN" & "LINE FEED"
1655      007532 011000      5$:     MOV     (RO),RO      ;; PICKUP "DATA TABLE" POINTER
1656      007534 001004      BNE     7$           ;; GO TYPE THE DATA
1657      007536 012600      6$:     MOV     (SP)+,RO     ;; RESTORE RO
1658      007540 104400 001231      TYPE   ,SCLRF      ;; "CARRIAGE RETURN" & "LINE FEED"
1659      007544 000207      RTS     PC           ;; RETURN
1660      007546      7$:
1661      007546 013046      MOV     2(RO)+,-(SP) ;; SAVE 2(RO)+ FOR TYPEOUT
1662      007550 104401      TYP0C   ;* GO TYPE--OCTAL ASCII(ALL DIGITS)
1663      007552 005710      TST     (RO)         ;; IS THERE ANOTHER NUMBER?
1664      007554 001770      BEQ     6$           ;; BR IF NO
1665      007556 104400 007564      TYPE   ,BS          ;; TYPE TWO(2) SPACES
1666      007562 000771      BR      7$           ;; LOOP
1667      007564 020040 000      8$:     .ASCIZ  / /           ;; TWO(2) SPACES
1668      007570      .EVEN
1669      ;*****
1670
1671      .SBTTL  ERROR HANDLER ROUTINE
1672
1673      ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
1674      ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
1675      ;*AND GO TO SERRTYP ON ERROR
1676      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1677      ;*SW15=1      HALT ON ERROR
1678      ;*SW13=1      INHIBIT ERROR TYPEOUTS
1679      ;*SW10=1      BELL ON ERROR
1680      ;*SW09=1      LOOP ON ERROR
1681      ;*CALL

```

```

1682 ;* ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER
1683
1684 $ERROR:
1685 007570 010637 001174 MOV SP,$REG6 ;GET THE CURRENT STACK POINTER VALUE
1686 007574 162737 000004 001174 SUB #4,$REG6 ;RESTORE IT TO ITS "PRE ERROR TRAP" VALUE FOR PR
1687 007602 016637 000002 001176 MOV 2(SP),$REG7 ;GET THE PS OFF OF THE STACK
1688 007610 005037 001172 CLR $REG5 ;PREPARE "$REG5" TO HOLD THE TEST #
1689 007614 113737 001102 001172 MOVB $TSTNM,$REG5 ;TEST # IS HELD IN THE LOW BYTE OF "TSTNM"
1690 007622 010037 001160 MOV RO,$REG0 ;MOST OF THE TIME RO HAS GOOD STUFF IN IT ALSO
1691 007626 105237 001103 7$: INCB $ERFLG ;; SET THE ERROR FLAG
1692 007632 001775 BEQ 7$ ;DON'T LET THE FLAG GO TO ZERO
1693 007634 013777 001102 171276 MOV $TSTNM,$DISPLAY ;DISPLAY TEST NUMBER AND ERROR FLAG
1694 007642 032777 002000 171266 BIT #BIT10,$SWR ;BELL ON ERROR?
1695 007650 001402 BEQ 1$ ;NO - SKIP
1696 007652 104400 001224 TYPE $BELL ;RING BELL
1697 007656 005237 001112 1$: INC $ERTTL ;COUNT THE NUMBER OF ERRORS
1698 007662 011637 001116 MOV (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
1699 007666 162737 000002 001116 SUB #2,$ERRPC
1700 007674 117737 171216 001114 MOVB $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
1701 007702 032777 020000 171226 BIT #BIT13,$SWR ;SKIP TYPEOUT IF SET
1702 007710 001004 BNE 20$ ;SKIP TYPEOUTS
1703 007712 004737 007434 JSR PC,$ERRTYP ;;GO TO USER ERROR ROUTINE
1704 007716 104400 001231 TYPE , $CRLF
1705 007722 20$:
1706 007722 005777 171210 2$: TST $SWR ;;HALT ON ERROR
1707 007726 100006 BPL 3$ ;SKIP IF CONTINUE
1708 007730 000000 HALT ;HALT ON ERROR!
1709 007732 022737 006166 000042 CMP #SENDAD,$#42 ;ACT-11 AUTO-ACCEPT?
1710 007740 001001 BNE 3$ ;BRANCH IF NO
1711 007742 000000 HALT ;YES
1712 007744 032777 001000 171164 3$: BIT #BIT09,$SWR ;LOOP ON ERROR SWITCH SET?
1713 007752 001402 BEQ 4$ ;BR IF NO
1714 007754 013716 001110 MOV $LPERR,(SP) ;FUDGE RETURN FOR LOOPING
1715 007760 005737 001222 4$: TST $ESCAPE ;CHECK FOR AN ESCAPE ADDRESS
1716 007764 001402 BEQ 5$ ;BR IF NONE
1717 007766 013716 001222 MOV $ESCAPE,(SP) ;FUDGE RETURN ADDRESS FOR ESCAPE
1718 007772 5$:
1719 007772 000002 RTI ;;RETURN
1720 ;*****
1721
1722 .SBTTL TRAP DECODER
1723
1724 ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
1725 ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
1726 ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
1727 ;*GO TO THAT ROUTINE.
1728
1729 $TRAP: MOV RO,-(SP) ;;SAVE RO
1730 007776 016600 000002 MOV 2(SP),RO ;;GET TRAP ADDRESS
1731 010002 005740 TST -(RO) ;;BACKUP BY 2
1732 010004 111000 MOVB (RO),RO ;;GET RIGHT BYTE OF TRAP
1733 010006 006300 ASL RO ;;POSITION FOR INDEXING
1734 010010 016000 010016 MOV $TRPAD(RO),RO ;;INDEX TO TABLE
1735 010014 000200 RTS RO ;;GO TO ROUTINE
1736
1737

```

1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793

010016
010016 006546
010020 007232
010022 007206
010024 007246
010026 006762

010030 012737 010172 000024
010036 012737 000340 000026
010044 010046
010046 010146
010050 010246
010052 010346
010054 010446
010056 010546
010060 013746 010420
010064 013746 010422
010070 013746 010210
010074 013746 001672
010100 010637 010176
010104 012737 010116 000024
010112 000000
010114 000776

010116 013706 010176
010122 005037 010176
010126 005237 010176
010132 001375
010134 012605
010136 012604
010140 012603
010142 012602
010144 012601
010146 012600
010150 012737 010030 000024
010156 012737 000340 000026
010164 104400
010166 010200
010170 000002
010172 000000
010174 000776
010176 000000
010200 005015 047520 042527
010206 000122

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

ROUTINE
-----
$TRPAD: $TYPE ;;CALL=TYPE TRAP+0(104400) TTY TYPEOUT ROUTINE
$TYPOC ;;CALL=TYPOC TRAP+1(104401) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
$TYPOS ;;CALL=TYPOS TRAP+2(104402) TYPE OCTAL NUMBER (NO LEADING ZEROS)
$TYPON ;;CALL=TYPON TRAP+3(104403) TYPE OCTAL NUMBER (AS PER LAST CALL)
$TYPDS ;;CALL=TYPDS TRAP+4(104404) TYPE DECIMAL NUMBER (WITH SIGN)
;*****

.SBTTL POWER DOWN AND UP ROUTINES
:POWER DOWN ROUTINE
$PWRDN: MOV $SILLUP, @PWRVEC ;;SET FOR FAST UP
MOV #340, @PWRVEC+2 ;;PRIO:7
MOV RO, -(SP) ;;PUSH RO ON STACK
MOV R1, -(SP) ;;PUSH R1 ON STACK
MOV R2, -(SP) ;;PUSH R2 ON STACK
MOV R3, -(SP) ;;PUSH R3 ON STACK
MOV R4, -(SP) ;;PUSH R4 ON STACK
MOV R5, -(SP) ;;PUSH R5 ON STACK
MOV POWPUS, -(SP) ;;PUSH POWPUS ON STACK
MOV POWPOP, -(SP) ;;PUSH POWPOP ON STACK
MOV POWMES, -(SP) ;;PUSH POWMES ON STACK
MOV T0001, -(SP) ;;PUSH T0001 ON STACK
MOV SP, $SAVR6 ;;SAVE SP
MOV $PWRUP, @PWRVEC ;;SET UP VECTOR
HALT
BR .-2 ;;HANG UP

:POWER UP ROUTINE
$PWRUP: MOV $SAVR6, SP ;;GET SP
CLR $SAVR6 ;;WAIT LOOP FOR THE TTY
IS: INC $SAVR6 ;;WAIT FOR THE INC
BNE IS OF <POWPUS>, <POWPOP>, <POWMES>, <T0001> WORD
MOV (SP)+, R5 ;;POP STACK INTO R5
MOV (SP)+, R4 ;;POP STACK INTO R4
MOV (SP)+, R3 ;;POP STACK INTO R3
MOV (SP)+, R2 ;;POP STACK INTO R2
MOV (SP)+, R1 ;;POP STACK INTO R1
MOV (SP)+, R0 ;;POP STACK INTO R0
MOV $PWRDN, @PWRVEC ;;SET UP THE POWER DOWN VECTOR
MOV #340, @PWRVEC+2 ;;PRIO:7
TYPE $POWER ;;REPORT THE POWER FAILURE
$PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
RTI
$SILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
BR .-2 ;;BEFORE THE POWER DOWN WAS COMPLETE
$SAVR6: 0 ;;PUT THE SP HERE
$POWER: .ASCIZ <15><12>"POWER"
```

```

1794
1795 010210 005015 042522 052123 POWMES: .EVEN
1796 010216 051101 044524 043516 POWMES: .ASCIZ <15> <12> "RESTARTING AFTER A POWER FIALURE" <15> <12> <12>
1797 010224 040440 052106 051105
1798 010232 040440 050040 053517
1799 010240 051105 043040 040511
1800 010246 052514 042522 005015
1801 010254 000012
1802 010256 005015 046412 026504 STMES: .ASCIZ <15><12><12>"MD-11-DZKWA-E LINE FREQUENCY CLOCK TEST"<15><12>
1803 010264 030461 042055 045532
1804 010272 040527 042455 046040
1805 010300 047111 020105 051106
1806 010306 050505 042525 041516
1807 010314 020131 046103 041517
1808 010322 020113 042524 052123
1809 010330 005015 000
1810
1811 010333 124 040522 050120 TRPMES: .ASCIZ "TRAPPED TO LOC 4 FROM LOCATION "
1812 010340 042105 052040 020117
1813 010346 047514 020103 020064
1814 010354 051106 046517 046040
1815 010362 041517 052101 047511
1816 010370 020116 000040
1817 010374 042522 052123 051101 TRPM2S: .ASCIZ "RESTARTING PROGRAM"
1818 010402 044524 043516 050040
1819 010410 047522 051107 046501
1820 010416 000
1821
1822 010420 010420 .EVEN
1823 010422 001234 POWPUS: WORD
1824 010424 020040 020040 020040 POWPOP: WORD
1825 010432 020040 020040 020040 EM1: .ASCIZ " LKS LKS "
1826 010440 020040 020040 020040
1827 010446 020040 020040 020040
1828 010454 020040 020040 020040
1829 010462 020040 045514 020123
1830 010470 020040 020040 045514
1831 010476 020123 020040 020040
1832 010504 000
1833 010505 050 041520 020051 DH1: .ASCIZ "(PC) (PS) (SP) TEST# WAS S/B "
1834 010512 020040 024040 051520
1835 010520 020051 020040 024040
1836 010526 050123 020051 020040
1837 010534 052040 051505 021524
1838 010542 020040 053440 051501
1839 010550 020040 020040 051440
1840 010556 041057 020040 020040
1841 010564 000040
1842
1843 010566 001116 001176 001174 .EVEN
1844 010574 001172 177546 001124 DT1: $ERRPC,$REG7,$REG6,$REG5,LKS,$GDDAT
1845 010602 000000 0
1846 010604 046103 041517 020113 EM2: .ASCIZ "CLOCK FAILED TO INTERRUPT"
1847 010612 040506 046111 042105
1848 010620 052040 020117 047111
1849 010626 042524 051122 050125

```


1906	011312	020040	020040	020040		
1907	011320	042520	044522	042117		
1908	011326	020040	042520	044522		
1909	011334	042117	000			
1910		011340				
1911	011340	001116	001176	001174	.EVEN	
1912	011346	001172	001162	001164	DT4:	\$ERRPC,\$REG7,\$REG6,\$REG5,\$REG1,\$REG2
1913	011354	000000			0	
1914	011356	045514	020123	042522	EM5:	.ASCIZ "LKS REGISTER RESPONDS TO ANOTHER ADDRESS"
1915	011364	044507	052123	051105		
1916	011372	051040	051505	047520		
1917	011400	042116	020123	047524		
1918	011406	040440	047516	044124		
1919	011414	051105	040440	042104		
1920	011422	042522	051523	000		
1921	011427	050	041520	020051	DH5:	.ASCIZ "(PC) (PS) (SP) TEST# ADDRESS"
1922	011434	020040	024040	051520		
1923	011442	020051	020040	024040		
1924	011450	050123	020051	020040		
1925	011456	052040	051505	021524		
1926	011464	020040	040440	042104		
1927	011472	042522	051523	000		
1928		011500			.EVEN	
1929	011500	001116	001176	001174	DT5:	\$ERRPC,\$REG7,\$REG6,\$REG5,\$GDADR
1930	011506	001172	001120			
1931	011512	000000			0	
1932	011514	020101	047516	051440	EM6:	.ASCIZ "A NO SACK TIMEOUT HAS OCCURED"
1933	011522	041501	020113	044524		
1934	011530	042515	052517	020124		
1935	011536	040510	020123	041517		
1936	011544	052503	042522	000104		
1937	011552	050050	024503	020040	DH6:	.ASCIZ "(PC) (PS) (SP) TEST# (LKS) "
1938	011560	020040	050050	024523		
1939	011566	020040	020040	051450		
1940	011574	024520	020040	020040		
1941	011602	042524	052123	020043		
1942	011610	020040	046050	051513		
1943	011616	020051	020040	000		
1944		011624			.EVEN	
1945	011624	001116	001176	001174	DT6:	\$ERRPC,\$REG7,\$REG6,\$REG5,LKS
1946	011632	001172	177546			
1947	011636	000000			0	
1948	011640	051127	047117	020107	EM7:	.ASCIZ "WRONG CONDITION CODES WERE PUT ONTO STACK BY INTERRUPT"
1949	011646	047503	042116	052111		
1950	011654	047511	020116	047503		
1951	011662	042504	020123	042527		
1952	011670	042522	050040	052125		
1953	011676	047440	052116	020117		
1954	011704	052123	041501	020113		
1955	011712	054502	044440	052116		
1956	011720	051105	052522	052120		
1957	011726	000				
1958	011727	050	041520	020051	DH7:	.ASCIZ "(PC) (PS) (SP) TEST# CC WAS CC S/B"
1959	011734	020040	024040	051520		
1960	011742	020051	020040	024040		
1961	011750	050123	020051	020040		

```

1962 011756 052040 051505 021524
1963 011764 020040 041440 020103
1964 011772 040527 020123 041440
1965 012000 020103 027523 000102
1966
1967 012006 001116 001176 001174 .EVEN
1968 012014 001172 000776 001124 DT7: $ERRPC,$REG7,$REG6,$REG5,BUF1,$GDDAT
1969 012022 000000 0
1970 012024 051127 047117 020107 EM10: .ASCIZ "WRONG PC PUT ONTO THE STACK BY AN INTERRUPT"
1971 012032 041520 050040 052125
1972 012040 047440 052116 020117
1973 012046 044124 020105 052123
1974 012054 041501 020113 054502
1975 012062 040440 020116 047111
1976 012070 042524 051122 050125
1977 012076 000124
1978 012100 050050 024503 020040 DH10: .ASCIZ "(PC) (PS) (SP) TEST# 2(SP)WAS 2(SP)S/B "
1979 012106 020040 050050 024523
1980 012114 020040 020040 051450
1981 012122 024520 020040 020040
1982 012130 042524 052123 020043
1983 012136 040040 051450 024520
1984 012144 040527 020123 024100
1985 012152 050123 051451 041057
1986 012160 020040 000
1987
1988 012164 012164 .EVEN
1989 012172 001116 001176 001174 DT10: $ERRPC,$REG7,$REG6,$REG5,BUF2,$GDDAT
1990 012172 001172 000774 001124
1991 012200 000000 0
1992 012202 051124 042531 020104 EM11: .ASCIZ "TRYED TO ACCESS THE LKS REGISTER, AND TRAPPED"
1993 012210 047524 040440 041503
1994 012216 051505 020123 044124
1995 012224 020105 045514 020123
1996 012232 042522 044507 052123
1997 012240 051105 020054 047101
1998 012246 020104 051124 050101
1999 012254 042520 000104
2000 012260 050050 024503 020040 DH11: .ASCIZ "(PC) (PS) (SP) TEST#"
2001 012266 020040 050050 024523
2002 012274 020040 020040 051450
2003 012302 024520 020040 020040
2004 012310 042524 052123 000043
2005
2006 012316 001116 001176 001174 .EVEN
2007 012324 001172 0 DT11: $ERRPC,$REG7,$REG6,$REG5
2008 012326 000000 0
000001 .END

```

A0004	002104	557#	560				
A0005	002172	578#	581				
A0006	002302	605#	608				
A0007	002434	635#	638				
A0010	002614	675#	678				
A0011	002716	704#	707				
A0012	003102	746#	749				
A0013	003172	776#	779				
A0014	003516	849	858#				
A0015	003634	870	882#				
A0016	003752	894	906#				
A0017	004070	918	930#				
A0020	004206	942	954#				
A0021	004324	966	978#				
A0022	004462	1005	1007#				
A0023	004530	1021#	1024				
A0024	004652	1049#	1050				
A0025	004762	1073#	1076				
A0026	005134	1109#	1110				
A0027	005250	1135#	1136				
A0030	005410	1153	1163#				
A0031	005476	1183#	1186				
A0032	005630	1220#	1223				
A0033	005766	1253#	1256				
BIT0	= 000001	291#					
BIT00	= 000001	281#	291				
BIT01	= 000002	280#	290				
BIT02	= 000004	279#	289				
BIT03	= 000010	278#	288				
BIT04	= 000020	277#	287				
BIT05	= 000040	276#	286				
BIT06	= 000100	275#	285				
BIT07	= 000200	274#	284				
BIT08	= 000400	273#	283	1369			
BIT09	= 001000	272#	282	1377	1712		
BIT1	= 000002	290#					
BIT10	= 002000	271#	1694				
BIT11	= 004000	270#	1384				
BIT12	= 010000	269#					
BIT13	= 020000	268#	1701				
BIT14	= 040000	267#	1355				
BIT15	= 100000	266#					
BIT2	= 000004	289#					
BIT3	= 000010	288#					
BIT4	= 000020	287#					
BIT5	= 000040	286#					
BIT6	= 000100	285#					
BIT7	= 000200	284#					
BIT8	= 000400	283#					
BIT9	= 001000	282#					
BPTVEC	= 000014	298#					
BUF1	= 000776	310#	1228*	1237	1261*	1967	
BUF2	= 000774	309#	1229*	1262*	1270	1988	
B0005	002210	579	584#				
B0006	002320	606	611#				
B0007	002452	636	641#				

B0010	002632	676	681#		
B0011	002734	705	710#		
B0012	003120	747	752#		
B0013	003210	777	782#		
B0014	003530	859	861#		
B0015	003646	883	885#		
B0016	003764	907	909#		
B0017	004102	931	933#		
B0020	004220	955	957#		
B0021	004336	979	981#		
B0023	004546	1022	1027#		
B0025	005000	1074	1079#		
B0030	005426	1165	1167#	1168	
B0031	005514	1184	1189#		
B0032	005646	1221	1226#		
B0033	006004	1254	1259#		
C0007	002462	644#	646		
C0010	002642	684#	686		
C0011	003014	723#	726		
C0013	003224	786#	789		
C0025	005024	1066	1087#		
C0031	005550	1198#	1201		
C0032	005712	1230	1237#		
C0033	006050	1263	1270#	1272	
DDISP =	177570	213#	361		
DH1	010505	413	1833#		
DH10	012100	448	1978#		
DH11	012260	453	1999#		
DH2	010636	418	1851#		
DH3	011017	423	1872#		
DH4	011203	428	1894#		
DH5	011427	433	1921#		
DH6	011552	438	1937#		
DH7	011727	443	1958#		
DISPLA	001140	361#	497*	1398*	1693*
DISPRE	000174	319#	497		
DSWR =	177570	212#	360		
DT1	010566	414	1843#		
DT10	012164	449	1988#		
DT11	012316	454	2005#		
DT2	010710	419	1859#		
DT3	011070	424	1880#		
DT4	011340	429	1911#		
DT5	011500	434	1929#		
DT6	011624	439	1945#		
DT7	012006	444	1967#		
D0007	002474	627	649#		
D0010	002654	662	689#		
D0011	003032	724	729#		
D0013	003242	787	792#		
D0031	005566	1199	1204#		
EMTVEC =	000030	301#	479*	480*	
EM1	010424	412	1824#		
EM10	012024	447	1970#		
EM11	012202	452	1991#		
EM2	010604	417	1846#		

.SCMTA	1#	176#	334
.SDB2D	1#		
.SDB20	1#		
.SDIV	1#		
.SEOP	1#	175#	1282
.SERRO	1#	176#	1669
.SERRT	1#	176#	1621
.SMULT	1#		
.SPOWE	1#	176#	1751
.SRAND	1#		
.SRDDE	1#		
.SRDOC	1#		
.SREAD	1#		
.SR2AZ	1#		
.SSAVE	1#		
.SSB2D	1#		
.SSB20	1#		
.SSCOP	1#	175#	1338
.SSIZE	1#		
.SSUPR	1#		
.STRAP	1#	176#	1720
.STYPB	1#		
.STYPD	1#	176#	1475
.STYPE	1#	176#	1402
.STYPO	1#	176#	1543
.S4OCA	1#		

ADD	1430	1508	1572	1582	1644										
ASL	1641	1642	1643	1733											
ASLB	1513														
BCC	1514														
BEQ	502	529	615	832	859	893	907	931	955	979	1030	1053	1088	1113	1140
	1238	1271	1310	1370	1372	1374	1378	1387	1433	1599	1646	1651	1664	1692	1695
	1713	1716													
BGE	1390	1467													
BGT	1299	1522	1606												
BHI	1376														
BIC	1296	1596													
BIS	1516	1517	1601	1602											
BISB	1633														
BIT	528	586	614	858	882	906	930	954	978	1004	1355	1369	1377	1384	1694
	1701	1712													
BITB	1455														
BLT	1446	1505	1521	1607											
BMI	543	558	579	606	636	650	676	690	705	724	747	756	777	787	807
	817	1022	1074	1184	1199	1221	1254	1512							
BNE	475	560	581	587	608	638	646	678	686	707	726	749	779	799	799
	809	819	830	1005	1024	1050	1076	1110	1136	1168	1186	1201	1223	1256	1356
	1385	1427	1435	1442	1456	1463	1510	1597	1634	1656	1702	1710	1777		
BPL	795	826	1421	1460	1496	1526	1595	1707							
BR	495	515	583	610	640	648	668	671	680	688	709	715	718	728	732
	751	781	791	811	821	1026	1078	1086	1170	1188	1203	1207	1225	1241	1258
	1358	1364	1367	1380	1383	1423	1439	1449	1458	1465	1507	1524	1573	1588	1609
CLR	1639	1666	1771	1790											
	461	473	486	487	514	555	556	576	577	603	604	613	630	633	634
	643	664	674	683	702	703	722	744	745	753	766	767	773	775	784
	785	793	797	800	801	803	805	814	815	824	828	846	856	861	863
	872	880	885	887	896	904	909	911	920	928	933	935	944	952	957
	959	968	976	981	983	1002	1007	1019	1020	1042	1067	1072	1080	1106	1126
	1128	1134	1150	1154	1155	1159	1164	1169	1181	1182	1197	1218	1219	1228	1229
	1232	1251	1252	1261	1262	1265	1279	1293	1294	1322	1329	1382	1396	1499	1502
	1586	1632	1688	1775											
CLRB	1160	1381	1464	1528											
CMP	474	498	669	716	831	1029	1052	1087	1112	1139	1237	1270	1365	1389	1520
	1709														
CMPB	1371	1375	1432	1434	1441	1462	1466								
DEC	1297	1640													
DECB	1445	1448	1594	1605											
EMT	206														
HALT	318	1422	1708	1711	1770	1789									
INC	559	580	607	637	644	677	684	706	725	748	778	788	796	798	808
	818	827	829	1023	1049	1075	1109	1135	1185	1200	1222	1255	1295	1388	1506
	1600	1608	1697	1776											
INCB	1167	1393	1468	1691											
IOT	207														
JMP	323	991	1317	1337											
JSR	469	1312	1323	1330	1440	1447	1454	1703							
MOV	460	462	463	465	466	468	472	476	477	478	479	480	481	482	483
	484	485	489	490	491	492	493	496	497	499	500	510	511	512	513
	523	524	525	527	537	538	539	540	551	552	553	554	568	569	570
	571	572	573	585	595	596	597	598	599	600	612	623	624	625	626
	627	628	629	631	642	659	660	661	662	663	665	666	667	670	672
	673	682	699	700	701	711	712	713	714	717	719	720	721	740	741

	742	743	768	769	772	783	802	813	823	833	834	847	848	849	850
	851	852	855	857	862	870	871	873	874	875	876	879	881	886	894
	895	897	898	899	900	903	905	910	918	919	921	922	923	924	927
	929	934	942	943	945	946	947	948	951	953	958	966	967	969	970
	971	972	975	977	982	992	993	994	995	996	997	998	1001	1015	1016
	1017	1018	1028	1037	1038	1039	1040	1041	1044	1045	1048	1061	1062	1063	1064
	1065	1066	1068	1069	1081	1083	1096	1097	1098	1099	1100	1101	1104	1105	1107
	1108	1122	1123	1124	1125	1127	1129	1130	1133	1137	1138	1148	1149	1151	1152
	1153	1156	1158	1161	1163	1165	1178	1179	1180	1190	1191	1192	1195	1196	1215
	1216	1217	1227	1230	1231	1233	1239	1248	1249	1250	1260	1263	1264	1266	1272
	1300	1304	1309	1325	1360	1361	1363	1366	1379	1391	1392	1394	1395	1398	1399
	1424	1425	1429	1443	1489	1490	1491	1492	1493	1494	1495	1500	1503	1523	1529
	1530	1531	1532	1533	1535	1536	1569	1577	1578	1579	1585	1592	1610	1611	1612
	1613	1614	1631	1636	1645	1650	1655	1657	1661	1685	1687	1690	1693	1698	1714
	1717	1729	1730	1734	1756	1757	1758	1759	1760	1761	1762	1763	1764	1765	1766
	1767	1768	1769	1774	1778	1779	1780	1781	1782	1783	1784	1785			
MOV8	488	1003	1397	1426	1453	1461	1498	1501	1515	1518	1527	1570	1571	1574	1575
	1576	1580	1583	1584	1603	1689	1700	1732							
NEG	1497	1581													
NOP	645	685	730	731	836	1205	1206	1313	1314	1315	1332	1333	1334	1335	1354
RESET	503	526	541	754	1008	1051	1111	1280	1311	1336					
ROL	1587	1589	1590	1591	1593										
RTI	467	574	601	632	770	853	877	901	925	949	973	999	1046	1070	1082
	1102	1131	1157	1166	1193	1234	1267	1400	1431	1537	1615	1719	1788		
RTS	1470	1659	1735												
SEC	1235	1268													
SEN	1235	1268													
SEV	1235	1268													
SEZ	1235	1268													
SUB	1326	1504	1686	1699											
TRAP	464	1737	1747	1748	1749	1750									
TST	494	501	1362	1386	1428	1436	1457	1509	1519	1598	1663	1706	1715	1731	
TSTB	542	557	578	605	635	649	675	689	704	723	746	755	776	786	794
	806	816	825	1021	1073	1183	1198	1220	1253	1373	1420	1459	1511	1525	
WAIT	1084	1162	1236	1269											
.ABS	168														
.ASCII	391	392													
.ASCIZ	390	393	1318	1667	1792	1795	1802	1811	1817	1824	1833	1846	1851	1862	1872
	1883	1894	1914	1921	1932	1937	1948	1958	1970	1978	1991	1999			
.BLKW	1542														
.BYTE	344	345	350	351	366	367	368	369	1321	1616	1617	1618	1619		
.ENABL	1	169	174												
.END	2008														
.ENDC	178	183	196	198	199	200	206	292	306	324	325	331	333	335	342
	344	370	380	388	389	390	391	395	396	476	477	479	481	483	485
	486	487	489	491	501	1283	1286	1287	1288	1290	1293	1299	1302	1303	1307
	1308	1317	1318	1321	1322	1339	1345	1350	1355	1357	1368	1371	1372	1373	1375
	1377	1384	1388	1393	1394	1398	1401	1402	1403	1426	1476	1544	1622	1640	1669
	1670	1676	1691	1698	1703	1704	1705	1706	1712	1719	1720	1721	1730	1733	1746
	1747	1748	1749	1750	1751	1752	1768	1778	1788	1795					
.EQUIV	206	207	209	224	225	254	255	256	257	258	259	260	261	262	263
	282	283	284	285	286	287	288	289	290	291					
.EVEN	1668	1794	1821	1842	1858	1879	1910	1928	1944	1966	1987	2004			
.IF	178	179	196	197	198	199	200	204	264	292	321	324	329	331	334
	341	343	370	380	388	389	390	394	395	472	476	477	479	481	483
	485	486	487	489	501	1282	1286	1287	1288	1289	1290	1292	1298	1301	1303

	1307	1308	1317	1318	1338	1344	1349	1354	1355	1367	1369	1370	1371	1373	1374
	1375	1384	1386	1394	1395	1400	1401	1402	1426	1475	1543	1621	1639	1655	1669
	1675	1685	1694	1701	1703	1704	1706	1709	1712	1719	1720	1729	1733	1737	1747
.IFF	1748	1749	1750	1751	1764	1778	1786	1788	1792						
	196	198	199	200	204	325	331	333	335	342	343	370	395	477	1283
	1289	1293	1299	1302	1318	1339	1368	1371	1372	1375	1401	1403	1476	1544	1622
.IFT	1640	1669	1670	1675	1694	1719	1720	1721	1730	1752	1788				
.IFTF	1383	1704													
.IIF	1301	1703													
	178	183	188	193	194	195	196	199	200	318	394	477	479	485	486
	487	489	490	1287	1293	1294	1305	1318	1322	1345	1346	1347	1348	1349	1350
	1382	1383	1398	1401	1402	1475	1637	1662	1676	1677	1678	1679	1680	1709	1720
.IRP	1746	1747	1748	1749	1750										
.LIST	178	395	1354	1489	1529	1685	1758	1764	1778						
	1	170	178	199	306	318	370	372	373	374	375	376	377	378	379
	380	381	382	383	384	385	386	387	388	491	1293	1349	1709	1737	1746
.MACRO	1747	1748	1749	1750	1751										
.MCALL	1	178	200	334	1669	1737									
.NLIST	175	176	177	306	491										
	1	171	178	199	306	318	370	372	373	374	375	376	377	378	379
	380	381	382	383	384	385	386	387	388	491	1293	1349	1709	1737	1746
.PAGE	1747	1748	1749	1750	1751										
.REM	334	394													
.REPT	1														
.SBTTL	318	372	380												
	189	202	312	322	326	336	397	507	520	534	548	565	592	620	655
	695	737	761	840	867	891	915	939	963	987	1012	1035	1058	1093	1118
.TITLE	1145	1175	1212	1245	1277	1284	1340	1404	1477	1545	1623	1671	1722	1738	1753
.WORD	178														
	318	319	320	332	343	346	347	348	349	352	353	354	355	356	357
	358	359	360	361	370	372	373	374	375	376	377	378	379	390	381
	382	383	384	385	386	387	1298	1301	1469	1620	1648	1653	1787		

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

*DZKWA.E, DZKWA.E.SEG/SOL/CRF/DS:ERFZ/EN:ABS=DSKM:SYSMAC.SML, DSKM:DZKWA.E.P11
 RUN-TIME: 41 33 4 SECONDS
 RUN-TIME RATIO: 138/80=1.7
 CORE USED: 29K (57 PAGES)

