

ICS-11

FIELD TEST PROGRAM
MD-11-DZICA-B

EP DZICA D DL A
COPYRIGHT 1976
FICHE 1 OF 1

NOV 1976
digital
MADE IN USA

This section of the document contains a grid of 14 columns and 18 rows of small, illegible data tables or charts. Each cell in the grid appears to contain a small table or a set of data points, but the text is too small to be read. The grid is located on the left side of the page, with the right side being a large, dark, mostly blank area.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

164000
000000
000001
000000
000000
000174
000174 000000
000176 000000
000200 000137 001704
000210 000210 003366
000214
001100
000011
000012
000015
000200
177776
177774
177772
177570
177570

```

.MCALL .HEADER,.SETUP,.$POWER,.$TYPE,.EQUAT,.SCMTAG,.SERRTYP,.STYPCCT
.MCALL .SCATCH,.$TRAP,.$TRMTRP,.$SETTRAP,.$SREAD,.$SRDDEC,.$ERROR,.$SSCOPE
.MCALL .SSB2D,.$SDB2D,.$SSAVE,.$SRAND
      $SWR=164000

      .HEADER 1/MAINDEC-11-DZICA-B/, 1976, 1/DAN DEKNIS/
      .TITLE MAINDEC-11-DZICA-B
      *COPYRIGHT (C) 1976
      *DIGITAL EQUIPMENT CORP.
      *MAYNARD, MASS. 01754
      *
      *PROGRAM BY DAN DEKNIS
      *
      *THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
      *PACKAGE (MAINDEC-11-DZQAC-CO), MAR 21, 1976.
      *
      $TN=1

      .SCATCH START
      .SBTTL TRAP CATCHER

      =0
      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2.HALT"
      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
      =174
      DISPREG: .WORD 0           ;;SOFTWARE DISPLAY REGISTER
      SWREG:   .WORD 0           ;;SOFTWARE SWITCH REGISTER
      .SBTTL STARTING ADDRESS(ES)
      JMP      @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM

      =210
      JMP      @#START1

      .EQUAT
      .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

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

```


1113 002000
1114 001000
1115 000400
1116 000200
1117 000100
1118 000040
1119 000020
1120 000010
1121 000004
1122 000002
1123 000001

BIT10= 2000
BIT09= 1000
BIT08= 400
BIT07= 200
BIT06= 100
BIT05= 40
BIT04= 20
BIT03= 10
BIT02= 4
BIT01= 2
BIT00= 1
.EQUIV BIT09,BIT9
.EQUIV BIT08,BIT8
.EQUIV BIT07,BIT7
.EQUIV BIT06,BIT6
.EQUIV BIT05,BIT5
.EQUIV BIT04,BIT4
.EQUIV BIT03,BIT3
.EQUIV BIT02,BIT2
.EQUIV BIT01,BIT1
.EQUIV BIT00,BIT0

1124 000004
1125 000010
1126 000014
1127 000014
1128 000014
1129 000020
1130 000024
1131 000030
1132 000034
1133 000060
1134 000064
1135 000240

.*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
.SCMTAG

000214

```

149
150
151 000214
152
153
154
155
156 001100
157 001100
158 001100 000000
159 001102 000
160 001103 000
161 001104 000000
162 001106 000000
163 001110 000000
164 001112 000000
165 001114 000
166 001115 001
167 001116 000000
168 001120 000000
169 001122 000000
170 001124 000000
171 001126 000000
172 001130 000000
173 001132 000000
174 001134 000
175 001135 000
176 001136 000000
177 001140 177570
178 001142 177570
179 001144 177560
180 001146 177562
181 001150 177564
182 001152 177566
183 001154 000
184 001155 002
185 001156 012
186 001157 000
187 001160 000000
188 001162 077
189 001163 015
190 001164 000012
191 001166
192

```

.SBTTL COMMON TAGS

STARS

```

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

```

.=1100

```

$CMTAG:          .WORD          0          ;; START OF COMMON TAGS
$PASS:          .WORD          0          ;; CONTAINS PASS COUNT
$STNM:          .BYTE          0          ;; CONTAINS THE TEST NUMBER
$ERFLG:         .BYTE          0          ;; CONTAINS ERROR FLAG
$ICNT:          .WORD          0          ;; CONTAINS SUBTEST ITERATION COUNT
$LPADR:         .WORD          0          ;; CONTAINS SCOPE LOOP ADDRESS
$LPERR:         .WORD          0          ;; CONTAINS SCOPE RETURN FOR ERRORS
$ERTTL:         .WORD          0          ;; CONTAINS TOTAL ERRORS DETECTED
$ITEMB:         .BYTE          0          ;; CONTAINS ITEM CONTROL BYTE
$ERMAX:         .BYTE          1          ;; CONTAINS MAX. ERRORS PER TEST
$ERRPC:         .WORD          0          ;; CONTAINS PC OF LAST ERROR INSTRUCTION
$GDADR:         .WORD          0          ;; CONTAINS ADDRESS OF 'GOOD' DATA
$BDADR:         .WORD          0          ;; CONTAINS ADDRESS OF 'BAD' DATA
$GDCAT:         .WORD          0          ;; CONTAINS 'GOOD' DATA
$BDDAT:         .WORD          0          ;; CONTAINS 'BAD' DATA
                .WORD          0          ;; RESERVED--NOT TO BE USED
                .WORD          0          ;;
$AUTOB:         .BYTE          0          ;; AUTOMATIC MODE INDICATOR
$INTAG:         .BYTE          0          ;; INTERRUPT MODE INDICATOR
                .WORD          0          ;;
$SWR:           .WORD          DSWR       ;; ADDRESS OF SWITCH REGISTER
$DISPLAY:       .WORD          DDISP     ;; ADDRESS OF DISPLAY REGISTER
$TKS:           177560                ;; TTY KBD STATUS
$TKB:           177562                ;; TTY KBD BUFFER
$TPS:           177564                ;; TTY PRINTER STATUS REG. ADDRESS
$TPB:           177566                ;; TTY PRINTER BUFFER REG. ADDRESS
$NULL:         .BYTE          0          ;; CONTAINS NULL CHARACTER FOR FILLS
$FILLS:        .BYTE          2          ;; CONTAINS # OF FILLER CHARACTERS REQUIRED
$FILLC:        .BYTE          12         ;; INSERT FILL CHARS. AFTER A "LINE FEED"
$TPFLG:        .BYTE          0          ;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
$TIMES:        0                      ;; MAX. NUMBER OF ITERATIONS
$QUES:         .ASCII         /?/      ;; QUESTION MARK
$CRLF:         .ASCII         <15>     ;; CARRIAGE RETURN
$LF:           .ASCII         <12>     ;; LINE FEED
STARS
:*****

```

193
194
195
196
197
199
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248

.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:

;A005 ERRORS

;ERROR 1 ADDRESS ABILITY

```

EM1          ;"COULD NOT SEND-RECIEVE DATA"
DH1          ;ERROR  A005
              ;PC  ADDR
DT1          ;$ERRC,  ADCSR,0
DF0          ;0
    
```

;ERROR 2 "BIT EXERCISER"

```

EM2          ;"SEND-RECIEVE DATA ERROR"
DH2          ;ERROR  A/D  GOOD  BAD
              ;PC  ADDR  DATA  DATA
DT2          ;$ERRPC,  ADCSR,  $GDDAT, $BDDAT,C
DF0          ;0
    
```

;ERROR 3 CONVERT BIT

```

EM3          ;CONVERT BIT FAILED TO SET
DH1          ;ERROR  A/D
DT1          ;PC  ADDR
DF0          ;$ERRPC,  ADCSR,0
    
```

;ERROR 4 CONVERT BIT

```

EM4          ;CONVERT BIT FAILED TO CLEAR
DH1          ;ERROR  A/D
DT1          ;PC  ADDR
DF0          ;$ERRPC,  ADCSR,0
    
```

;ERROR 5 READ C/R BIT

```

EM5          ;CAN'T READ DATA REGISTER
DH1          ;ERROR  A/D
DT1          ;PC  ADDR
DF0          ;$ERRPC,  ADCSR,0
    
```

;ERROR 6 A005 INTERRUPT

```

EM6          ;A005 FAILED TO INTERRUPT
DH1          ;ERROR  A/D
DT1          ;PC  ADDR
    
```

001166

001166 022141
 001170 022536

001172 001500
 001174 001476

001176 022177
 001200 022571

001202 001506
 001204 001476

001206 022231
 001210 022536
 001212 001500
 001214 001476

001216 022265
 001220 022536
 001222 001500
 001224 001476

001226 022323
 001230 022536
 001232 001500
 001234 001476

001236 022356
 001240 022536
 001242 001500

249	001244	001476	DF0	;SERRPC, ADCSR,0
250				
251				;ERROR 7 ADDS ADDR OR GENERIC CODE
252				
253	001246	022411	EM7	;ADDR. OR GENERIC CODE INCORRECT ON INTR.
254	001250	022665	DH3	;ERROR MODULE ICAR
255	001252	001506	DT2	; PC ADDR S/B WAS
256	001254	001476	DF0	;SERRPC, ADCSR, \$GDDAT, \$BDDAT
257				
258				;ERROR 10 RIF BIT ACTION
259				
260	001256	022470	EM10	;RIF DID NOT CLEAR INTR. FLAG ON ADDS
261	001250	022536	DH1	;ERROR A/D
262	001262	001500	DT1	; PC ADDR
263	001264	001476	DF0	;SERRPC, ADCSR
264				
265				;ERROR 11 COUNTER MODULE ADDRESSABILITY
266				
267	001266	023015	EM11	;COUNTER MODULE COULD NOT READ-WRITE BUFFER
268	001270	023077	DH4	;ERROR COUNTER
269	001272	001520	DT3	;PC ADDR
270	001274	001476	DF0	;SERRPC \$BDADR
271				
272				;ERROR 12 COUNTER MODULE "BIT EXERCISE ROUTINE
273				
274	001276	022177	EM2	
275	001300	022571	DH2	
276	001302	001526	DT4	
277	001304	001476	DF0	
278				
279				;ERROR 13 COUNTER MODULE COUNTING
280				
281	001306	023140	EM12	;COUNTER MODULE MISSED A COUNT
282	001310	022571	DH2	
283	001312	001526	DT4	
284	001314	001476	DF0	
285				
286				;ERROR 14 COUNTER MODULE FAILED TO INTERRUPT
287				
288	001316	023204	EM13	;COUNTER FAILED TO INTERRUPT
289	001320	023077	DH4	
290	001322	001520	DT3	
291	001324	001476	DF0	
292				
293				;ERROR 15 COUNTER MODULE ADDR OR GENERIC CODE INCORRECT
294				
295	001326	023251	EM14	
296	001330	022665	DH3	
297	001332	001526	DT4	
298	001334	001476	DF0	
299				
300				;ERROR 16 COUNTER DIDN'T HALT ON OVERFLOW
301				
302	001336	023347	EM15	
303	001340	022571	DH2	
304	001342	001526	DT4	

305	001344	001476	DF0		
306					
307				;ERROR 17 RIF DIDN'T CLEAR INTERRUPT FLAG ON COUNTER	
308					
309	001346	023431	EM16		
310	001350	023077	DH4		
311	001352	001520	DT3		
312	001354	001476	DF0		
313					
314				;ERROR 20 COUNTER MODULE INITIALIZATION PART 1	
315					
316	001356	023506	EM17		;INIT FAILED TO CLEAR COUNTER BIT(S)
317	001350	022571	DH2		
318	001362	001526	DT4		
319	001364	001476	DF0		
320					
321				;ERROR 21 ILLEGAL INTERRUPT POSTED ON ICS BUS	
322					
323	001366	023562	EM20		
324	001370	023077	DH4		
325	001372	001520	DT3		
326	001374	001476	DF0		
327					
328				;ERROR 22 ICS-11 FAILED TO INTERRUPT	
329					
330	001376	023630	EM21		
331	001400	023077	DH4		
332	001402	001520	DT3		
333	001404	001476	DF0		
334					
335				;ERROR 23 ICAR NOT ZERO AFTER FORCED ICS INTERRUPT.	
336					
337	001406	023666	EM22		
338	001410	022665	DH3		
339	001412	001526	DT4		
340	001414	001476	DF0		
341					
342				;ERROR 24 SYS INITIALIZE FAILED TO CLEAR COUNTER INTERRUPT FLAG	
343					
344	001416	023737	EM23		
345	001420	023077	DH4		
346	001422	001520	DT3		
347	001424	001476	DF0		
348					
349				;ERROR 25 COUNTER STARTED COUNTING AFTER SYSTEM INIT.	
350					
351	001426	024017	EM24		
352	001430	022571	DH2		
353	001432	001526	DT4		
354	001434	001476	DF0		
355					
356				;ERROR 26 A005 READ DUAL ADDR ERROR	
357					
358	001436	024100	EM25		
359	001440	024217	DH5		;ERROR A/D DUAL
360	001442	001540	DT5		; PC ADDR ADDR

```

361 001444 001476 DFO
362
363 ;ERROR 27 A005 WRITE DUAL ADDR. ERROR
364
365 001446 024135 EM26
366 001450 024217 DH5
367 001452 001540 DT5
368 001454 001476 DFO
369
370
371 ;ERROR 30 SEND -RECIEVE DATA ERROR
372
373 001456 024165 EM27
374 001460 022571 DH2
375 001462 001526 DT4
376 001464 001476 DFO
377
378
379 ;ERROR 31 SAME AS ERROR 30 ONLY NO HEADER TYPEOUT
380
381 001466 001476 DFO
382 001470 001163 $CRLF
383 001472 001526 DT4
384 001474 001476 DFO
385
386
387 001476 000000 DFO: 0
388 001500 001116 007062 000000 DT1: .WORD $ERRPC, ADCSR, 0
389 001506 001116 007062 001124 DT2: .WORD $ERRPC, ADCSR, $GDDAT, $BDDAT, 0
390 001514 001126 000000
391 001520 001116 001122 000000 DT3: .WORD $ERRPC, $BDADR, 0
392 001526 001116 001122 001124 DT4: .WORD $ERRPC, $BDADR, $GDDAT, $BDDAT, 0
393 001534 001126 000000
394 001540 001116 007062 001122 DT5: .WORD $ERRPC, ADCSR, $BDADR, 0
395 001546 000000
396
397
398 ;*TABLE OF CONSTANTS
399
400 001550 020320 FR110: 20320 ;TTY DELAY TIME
401 001552 006532 FR50: 6532
402 001554 005650 FR40: 5650
403 001556 002724 FR20: 2724
404 001560 002052 FR16: 2052 ;DELAY FOR 16 MILLSEC
405 001562 001024 FR5: 1024
406 001564 000500 FR3: 500 ;DELAY FOR 3 MILLI SEC.
407 001566 000000 FREQ: 0 ;DELAY TIME.
408 001570 000000 FREQ1: 0
409 001572 000000 FREQ2: 0
410 001574 000000 FREQ3: 0
411 001576 000152 FR1: 152 ;DELAY FOR 1 MILLI SEC.
412 001600 000220 FR1120: 220 ;ON AN 11/20 FRI X 3 FOR 11/45.
413 001602 000000 PATRNM: 0 ;PATTERN MODIFIER
414 001604 000000 PATRN: 0 ;PATTERN TO BE SENT TO OUTPUT MODULE
415 001606 000000 PATRNC: 0
416 001610 000000 PATJOY: 0

```

417	001612	171000		ICSMOD: 171000	; STARTING ADDRESS OF ICS MODULES.
418	001614	171776		ICSR: 171776	
419	001616	171774		ICAR: 171774	
420	001620	000234		ICSVT: 234	
421	001622	000236		ICSVT2: 236	
422	001624	000064		TPVCT: 64	
423	001626	000066		TPVCT2: 66	
424	001630	000000		INCFLG: 0	
425	001632	000000		TPBSY: 0	
426	001634	000000		HEADER: 0	
427	001636	000000		TOADR: 0	
428	001640	000000		TODAT: 0	
429	001642	000000		TOGEN: 0	
430	001644	000000		TPBSYP: 0	
431	001646	000001		ST200: 1	; CLEARED ON PROGRAM START AT 200
432	001650	000000		CORSIZ: 0	; END ADDR OF CORE.
433	001652	000000		EXPERT: 0	; 0=NOVICE MODE--1=EXPERT MODE
434	001654	000000		CONNT: 0	; 0=NORMAL; 1=MODS CONNECTED FOR TST 0
435	001656	000000		LPAY: 0	
436	001660	000000		LINEPR: 0	; 0=TTY OUTPUT---1=LINE PRINTER
437	001662	177514		LPCSR: 177514	
438	001664	177516		LPDBR: 177516	
439	001666	000000		TPCSR: 0	
440	001670	000000		TPDBR: 0	
441	001672	000000		TMPFIL: 0	
442	001674	000000		TMPVEC: 0	
443	001676	000000		ICSHGH: 0	
444	001700	000000		CTLLOC: 0	
445	001702	000000		ICSLMT: 0	
446					
447		000002		EXIT= 2	; RTI INSTRUCTION
448		000207		RETURN=207	; RTS PC INSTRUCTION
449		004737		GOSUB=4737	; JSR PC, INSTRUCTION
450		022626		POPSP2=22626	
451					
452	001704			.SETUP (<.\$STRAP,.\$SCOPE,.\$ERROR,.\$POWER)	
453					
454	001704			START: SETUP	
455				.SBTTL INITIALIZE THE COMMON TAGS	
456				;;CLEAR THE COMMON TAGS (\$CMTAG) AREA	
457	001704	012706	001100	MOV \$CMTAG,R6	;;FIRST LOCATION TO BE CLEARED
458	001710	005026		CLR (R6)+	;;CLEAR MEMORY LOCATION
459	001712	022706	001140	CMP \$SWR,R6 ;;DONE?	
460	001716	001374		BNE -6	;;LOOP BACK IF NO
461	001720	012706	001100	MOV \$STACK,SP	;;SETUP THE STACK POINTER
462				;;INITIALIZE A FEW VECTORS	
463	001724	012737	017424	MOV \$SCOPE,@#IOTVEC	;;IOT VECTOR FOR SCOPE ROUTINE
464	001732	012737	000340	MOV #340,@#IOTVEC+2	;;LEVEL 7
465	001740	012737	016576	MOV \$ERROR,@#EMTVEC	;;EMT VECTOR FOR ERROR ROUTINE
466	001746	012737	000340	MOV #340,@#EMTVEC+2	;;LEVEL 7
467	001754	012737	020474	MOV \$STRAP,@#TRAPVEC	;;TRAP VECTOR FOR TRAP CALLS
468	001762	012737	000340	MOV #340,@#TRAPVEC+2	;;LEVEL 7
469	001770	012737	017252	MOV \$SPWRDN,@#PWRVEC	;;POWER FAILURE VECTOR
470	001776	012737	000340	MOV #340,@#PWRVEC+2	;;LEVEL 7
471	002004	005037	001160	CLR \$TIMES	;;INITIALIZE NUMBER OF ITERATIONS
472	002010	012737	002010	MOV #.,\$LPADR	;;INITIALIZE THE LOOP ADDRESS FOR SCOPE

K01

```

473 002016          SWRSU
474          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
475          ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
476 002016 013746 000004      MOV    @#ERRVEC, -(SP)    ;;SAVE ERROR VECTOR
477 002022 012737 002056 000004  MOV    #64$, @#ERRVEC    ;;SET UP ERROR VECTOR
478 002030 012737 177570 001140  MOV    #DSWR, SWR        ;;SETUP FOR A HARDWARE SWICH REGISTER
479 002036 012737 177570 001142  MOV    #DDISP, DISPLAY   ;;AND A HARDWARE DISPLAY REGISTER
480 002044 022777 177777 177066  CMP    #-1, @SWR        ;;TRY TO REFERENCE HARDWARE SWR
481 002052 001012          BNE    66$                ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
482          ;;AND THE HARDWARE SWR IS NOT = -1
483 002054 000403          BR     65$                ;;BRANCH IF NO TIMEOUT
484 002056 012716 002064      64$:  MOV    #65$, (SP)        ;;SET UP FOR TRAP RETURN
485 002062 000002          RTI
486 002064 012737 000176 001140  65$:  MOV    #SWREG, SWR     ;;POINT TO SOFTWARE SWR
487 002072 012737 000174 001142  MOV    #DISPREG, DISPLAY
488 002100 012637 000004      66$:  MOV    (SP)+, @#ERRVEC  ;;RESTORE ERROR VECTOR
489
490 002104 012737 176543 016342  MOV    #176543, $HINUM
491 002112 012737 123456 016344  MOV    #123456, $LONUM
492 002120 005037 001654      CLR    CONNT
493
494          ;;
495          ;;FILL LOCATIONS "214-1000" WITH .+2, IOT
496          ;;
497
498 002124 012700 000214      FILVEC: MOV    #214, R0
499 002130 012701 000216      MOV    #216, R1
500 002134 010120      98$:  MOV    R1, (R0)+
501 002136 012720 000004      MOV    #4, (R0)+
502 002142 022121      CMP    (R1)+, (R1)+
503 002144 022700 001000      CMP    #1000, R0
504 002150 003371      BGT    98$
505 002152 104400      TYPE
506 002154 020574          MHEAD                ;TYPE HEADER
507
508          ;;
509          ;;SINCE ISC-11 IS CAPABLE OF DIFFERENT CONTROL REGISTERS
510          ;;ASK OPERATOR FOR FILE BOX AND INTERRUPT VECTOR ADDRESS
511          ;;
512 002156 012737 002164 001700      MOV    #10$, CTLLC
513 002164 104400 026251      10$:  TYPE, MFILE        ;ASK FOR FILE BOX
514 002170 012737 000001 002206  MOV    #1, 22$
515 002176 000401          BR     +4
516 002200 000771          BR     10$
517 002202 104411          INOCT
518 002204 001672          TMPFIL
519 002206 000001      22$:  1
520 002210 022737 000013 001672  CMP    #13, TMPFIL
521 002216 002003          BGE    11$
522 002220 104400 026336      TYPE, ILLEG        ;ILLEGAL FILE BOX... RE-ASK
523 002224 000757          BR     10$
524 002226 012737 171000 001702  11$:  MOV    #171000, ICSLMT    ;INIT. MODULE ADDRESSES
525 002234 012737 171776 001614  MOV    #171776, ICSR    ;INIT. ICSR
526 002242 005237 001672          INC    TMPFIL
527 002246 005337 001672      19$:  DEC    TMPFIL        ;DEC. FILE
528 002252 001407          BEQ    18$                ;BRANCH IF FOUND

```

```

529 002254 162737 000010 001614 SUB #10,ICSR ;NEXT ICSR
530 002262 062737 000040 001702 ADD #40,ICSLMT ;NEXT GROUP OF MODULE ADDRESSES
531 002270 000766 BR 19$ ;LOOP
532 002272 013737 001614 001616 18$: MOV ICSR,ICAR ;CREATE ICAR
533 002300 162737 000002 001616 SUB #2,ICAR
534 002306 013737 001702 001676 MOV ICSLMT,ICSHGH ;SET UPPER ADDRESS LIMIT
535 002314 062737 000040 001676 ADD #40,ICSHGH
536 002322 012737 002336 000004 MOV #12$,R4
537 002330 017700 177260 MOV @ICSR,R0
538 002334 000404 BR 97$
539 002336 022626 12$: POPSP2
540 002340 104400 026552 TYPE, NONXST
541 002344 000707 BR 10$
542 002346 012737 000006 000004 97$: MOV #6,R4
543 002354 012737 002362 001700 MOV #99$,CTLLOC
544 002362 104400 026305 99$: TYPE, MVECT ;ASK FOR INTERRUPT VECTOR
545 002366 012737 000001 002404 MOV #1,23$
546 002374 000401 ER +4
547 002376 000771 BR 99$
548 002400 104411 INOCT
549 002402 001674 TMPVEC
550 002404 000001 23$: I
551 002406 022737 000776 001674 CMP #776,TMPVEC
552 002414 002404 BLT 16$
553 002416 022737 000234 001674 CMP #234,TMPVEC
554 002424 003403 BLE 17$
555 002426 104400 026336 16$: TYPE, ILLEG ;NOT IN FLOATING AREA
556 002432 000753 BR 99$
557 002434 013737 001674 001620 17$: MOV TMPVEC,ICSVT
558 002442 062737 000002 001674 ADD #2,TMPVEC
559 002450 013737 001674 001622 MOV TMPVEC,ICSVT2 ;STORE VECTOR ADDRESS
560 ;**
561 ;**VERIFY THAT VECTOR GIVEN IS ACTUAL VECTOR BEFORE GOING
562 ;**ANY FARTHER
563 ;
564 ;
565 002456 013737 000020 002624 MOV @#20,93$ ;SAVE LOC. 20
566 002464 012737 002540 000020 MOV #95$,@#20 ;SET UP IOT TRAP VECTOR
567 002472 012777 002606 177120 MOV #94$,@ICSVT ;SET UP INTERRUPT ROUTINE
568 002500 012777 000340 177114 MOV #340,@ICSVT2 ;PS PICKUP
569 002506 012737 000240 177776 MOV #240,@#PS ;SET PRIORITY TO 5
570 002514 012777 000404 177072 MOV #404,@ICSR ;SET MAINT. TO INTERRUPT
571 002522 000240 NOP
572 002524 104400 026431 TYPE, NOINT ;NO INTERRUPT
573 002530 013737 002624 000020 MOV 93$,@#20 ;RESTORE LOC 20
574 002536 000413 BR 96$ ;RE-ASK QUESTION
575 002540 011605 95$: MOV (SP),R5 ;INVALID VECTOR ADDRESS PRINT
576 002542 024545 CMP -(R5),-(R5) ;SUB 4 FROM R5
577 002544 022626 POPSP2 ;POP STACK
578 002546 000240 NOP
579 002550 104400 026474 TYPE, FILINT ;PRINT WHERE VECTOR
580 002554 TYPOCT R5
581 002554 010546 MOV R5,-(SP) ;;SAVE R5 FOR TYPEOUT
582 002556 104401 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
583 002560 104400 026527 TYPE, CKJMP
584 002564 022626 POPSP2

```

```

585 002566 005077 177022 96$: CLR @ICSR
586 002572 013737 002624 000020 MOV 93$,@#20
587 002600 000000 HALT
588 002602 000137 002362 JMP 99$
589 002606 013737 002624 000020 94$: MOV 93$,@#20 ; INTERRUPT OK, RESTORE 20
590 002614 022626 POPSP2 ; POP STACK
591 002616 005077 176772 CLR @ICSR
592 002622 000401 BR .+4
593 002624 000000 93$: 0
594
595 ; FILL LOCATIONS "214-1000" WITH .+2. HALT
596 ;
597
598 002626 012737 003366 001700 MOV #START1,CTLLOC
599 002634 012700 000214 FILHLT: MOV #214,R0
600 002640 012701 000216 MOV #216,R1
601 002644 010120 90$: MOV R1,(R0)+
602 002646 012720 000000 MOV #0,(R0)+
603 002652 022121 CMP (R1)+,(R1)+
604 002654 022700 001000 CMP #1000,R0
605 002660 003371 BGT 90$
606 002662 012737 003652 000060 MOV #KEYSRV,@#60
607 002670 052777 000100 176246 BIS #100,@#TKS
608 002676 012737 002726 000004 MOV #9$,@#4
609 002704 005237 001656 INC LPAV
610 002710 005777 176746 TST @LPCSR
611 002714 104400 TYPE
612 002716 001163 $CRLF
613 002720 104400 TYPE
614 002722 025262 MLPV
615 002724 000403 BR 8$
616 002726 005037 001656 9$: CLR LPAV
617 002732 022626 POPSP2
618
619
620 002734 012737 002750 000004 8$: MOV #3$,@#4 ; DETERMINE CORE SIZE.
621 002742 005000 CLR R0
622 002744 005720 2$: TST (0)+
623 002746 000776 BR 2$
624 002750 005740 3$: TST -(0)
625 002752 010037 001650 MOV R0,CORSIZ
626 002756 100003 BPL 4$
627 002760 012737 077776 001650 MOV #077776,CORSIZ
628 002766 012737 000006 000004 4$: MOV #6,@#4
629 002774 013737 001600 001576 MOV FR1120,FR1 ; SET FREQ FOR 11/20
630 003002 012737 000002 000012 MOV #RTI,@#12 ; FIND OUT IF 11/20 OR 11/45
631 003010 000262 SEV ; IF 11/45 THEN DELAY ITERATIONS
632 003012 074101 OR %1,%1 ; MUST BE INCREASED.
633 003014 102427 .VS 5$
634 003016 063737 001600 001576 ADD FR1120,FR1 ; FLOW FELL THUR TO HERE -
635 003024 063737 001600 001576 ADD FR1120,FR1 ; IT MUST BE AN 11/45 OR EQUIV.
636 003032 013746 000004 MOV @#4,-(SP) ; SAVE LOC 4
637 003036 013746 000006 MOV @#6,-(SP) ; SAVE LOC 6
638 003042 012737 003062 000004 MOV #99$,@#4 ; SETUP FOR TRAP
639 003050 005737 177760 TST @#177760 ; CHECK IF 11/70
640 003054 006337 001576 ASL FR1 ; 11/70, DOUBLE

```

641	003060	000401			BR	.+4				;SKIP
642	003062	022626			99\$: POPSP2					;RESTORE STACK
643	003064	012637	000006		MOV	(SP)+,2#6				;RESTORE LOC 6
644	003070	012637	000004		MOV	(SP)+,2#4				;RESTORE LOC 4
645	003074	005037	000012		5\$: CLR	2#12				;RESTORE LOC 12.
646	003100	013737	001576	001564	MOV	FR1,FR3				;NOW WE MUST SET THE REST OF
647	003106	006137	001564		ROL	FR3				;OF THE DELAY TIMES UP.
648	003112	013737	001564	001562	MOV	FR3,FR5				
649	003120	063737	001576	001564	ADD	FR1,FR3				
650	003126	063737	001564	001562	ADD	FR3,FR5				
651	003134	013737	001562	001560	MOV	FR5,FR16				
652	003142	006137	001560		ROL	FR16				
653	003146	063737	001562	001560	ADD	FR5,FR16				
654	003154	013737	001560	001556	MOV	FR16,FR20				
655	003162	063737	001576	001560	ADD	FR1,FR16				
656	003170	063737	001562	001556	ADD	FR5,FR20				
657	003176	013737	001556	001552	MOV	FR20,FR50				
658	003204	006137	001552		ROL	FR50				
659	003210	063737	001562	001552	ADD	FR5,FR50				
660	003216	063737	001562	001552	ADD	FR5,FR50				
661	003224	013737	001552	001550	MOV	FR50,FR110				
662	003232	006137	001550		ROL	FR110				
663	003236	063737	001562	001550	ADD	FR5,FR110				
664	003244	063737	001562	001550	ADD	FR5,FR110				
665	003252	013737	001556	001554	MOV	FR20,FR40				
666	003260	006137	001554		ROL	FR40				
667	003264	005037	001652		CLR	EXPERT				
668										
669										
670	003270	012737	001564	001566	MOV	#FR3, FREQ				
671	003276	005037	001604		CLR	PATRN				
672	003302	005737	001646		TST	ST200				
673	003306	001406			BEQ	7\$				
674	003310	013737	001150	001666	MOV	\$TPS,TPCSR				
675	003316	013737	001152	001670	MOV	\$TPB,TPDBR				
676	003324	012700	013104		7\$: MOV	#INADR, RO				;CLEAR ADDR AREA
677	003330	005037	001646		CLR	ST200				;INDICATE START AT LOC 200.
678	003334	005020			1\$: CLR	(0)+				
679	003336	020027	013252		CMP	RO,#OUTS				
680	003342	001374			BNE	1\$				
681	003344	005037	001574		CLR	FREQ3				
682	003350	005037	001572		CLR	FREQ2				
683	003354	005037	001570		CLR	FREQ1				
684	003360	013737	001564	001566	MOV	FR3,FREQ				
685										
686	003366	012706	001100		START1: MOV	#1100,SP				
687	003372	005077	176216		CLR	2ICSR				
688	003376	005037	001100		CLR	\$PASS				
689	003402	005037	001660		CLR	LINEPR				
690	003406	013737	001566	013042	MOV	FREQ,RTEMP				
691	003414	005737	001646		TST	ST200				;HAS PROG. BEEN EVER STARTED
692	003420	001411			BEQ	3\$;AT ADDR. 200? IF SO CONTINUE.
693	003422	012737	020254	000034	MOV	#\$TYPE,2#34				;SET UP TO PRINT.
694	003430	005037	000036		CLR	2#36				
695	003434	104400			TYPE					
696	003436	021767			MSTERR					

```

697 003440 000137 001704 JMP START ;PROGRAM MUST BE STARTED AT ADDR. 200.
698 003444 013737 001550 001566 3$: MOV FR110, FREQ ;SET DELAY FOR TTY SETTLE TIME.
699 003452 005037 001570 CLR FREQ1
700 003456 013737 001666 001150 MOV TPCSR,$TPS
701 003464 013737 001670 001152 MOV TPOBR,$TPB
702 003472 104417 DELAY ;CAUSE A DELAY.
703 003474 000005 RESET ;ISSUE SYSTEM INITIALIZE.
704 003476 104417 DELAY ;TTY SETTLE DOWN TIME.
705 003500 012737 000340 177776 MOV #340,PS
706 003506 104400 99$: TYPE ;TYPE "TEST NO.?"
707 003510 021522 MTN
708 003512 012737 003630 003534 MOV #TSTNO, 1$
709 003520 012737 000001 003536 MOV #1, 1$+2
710 003526 000401 BR +4
711 003530 000766 BR 99$
712 003532 104411 INOCT ;GET TEST NUMBER
713 003534 003630 1$: TSTNO
714 003536 000001 J
715 003540 005737 003630 TST TSTNO
716 003544 100404 BMI 4$
717 003546 023727 003630 000007 CMP TSTNO, #7. ;LEGAL NUMBER?
718 003554 003403 BLE 2$
719 003556 104400 4$: TYPE ;NO-TYPE "NO SUCH TEST"
720 003560 021545 MTNL
721 003562 000701 BR
722 003564 013737 003630 013042 2$: MOV START1
TSTNO,RTEMP
723 003572 006337 013042 ASL RTEMP
724 003576 062737 003632 013042 ADD #TSTLST, RTEMP
725 003604 017737 007232 013042 MOV #RTEMP,RTEMP
726 003612 012777 000100 175324 MOV #100,$TKS ;ENABLE TTY TO INTERRUPT.
727 003620 005037 177776 CLR PS
728 003624 000177 007212 JMP #RTEMP ;GOTO TEST
729
730 003630 000000 TSTNO: 0 ;TEST NUMBER
731 003632 003706 TSTLST: TST0 ;INPUT AND OUTPUT MODULE EXERCISER
732 003634 005264 TST1 ;INPUT OR OUTPUT MODULE SIMPLE EXER.
733 003636 005530 TST2 ;DAC CALIBRATION TEST
734 003640 005622 TST3 ;DAC INTERACTION TEST
735 003642 005672 TST4 ;COUNTER MODULE TEST
736 003644 007064 TST5 ;A/D LOGIC TEST
737 003646 007744 TST6 ;A/D CALIBRATION TEST
738 003650 010706 TST7 ;A/D REPEATIBILITY TEST
739
740
741 ;*
742 ;*KEYBOARD INTERRUPT HANDLER
743 ;*
744 003652 017737 175270 003704 KEYSRV: MOV #STKB,2$
745 003660 042737 177600 003704 BIC #177600,2$
746 003666 122737 000003 003704 CMPB #3,2$
747 003674 001401 BEQ 1$
748 003676 000002 EXIT
749 003700 000137 003366 1$: JMP START1
750 003704 000000 2$: 0
751
752

```

```

753
754
755
756
757 003706 104400 TSTO: TYPE ;TYPE HEADER.
758 003710 024642 MHTO
759 003712 104412 INAR ;GET INPUT MODULE ADDRS.
760 003714 104413 OUTAR ;GET OUTPUT MODULE ADDRS.
761 003716 104414 FATAR ;GET PATTERN (OR USE DEFAULT).
762 003720 104415 DELAR ;GET DELAY TIME (OR USE DEFAULT).
763 003722 005037 001100 CLR $PASS
764 003726 005037 001634 CLR HEADER
765
766 003732 012777 004406 175660 MOV #TOINT, JICSVT ;INITIAL SETUP.
767 003740 012777 004332 175656 MOV #TCPINT, JTPVCT ;SET UP FOR ICS INTERRUPTS.
768 003746 012777 000340 175646 MOV #340, JICSVT2 ;SET UP FOR TTY INTERRUPTS.
769 003754 012777 000200 175544 MOV #200, JTPVCT2 ;SET CPU PRIORITY TO 6 ON ICS INTR.
770 003762 005037 001632 TPBSY ;SET CPU PRIORITY TO 4 ON TTY INTR.
771 003766 012700 013252 CLR ;SET NO TTY OUTPUT IN PROGRESS
772 003772 005020 1S: CLR #OUTS, RO ;CLEAR TEMP STORAGE AREA OF INPUT
773 003774 020027 013312 CMP (0)+ ;MODULE DATA CHANGE.
774 004000 001374 BNE 1S
775 004002 005737 001654 TST CONNT
776 004006 001002 BNE 2S
777 004010 104400 TYPE ;TYPE HEADER.
778 004012 020756 MTOH
779 004014 005037 001636 2S: CLR TOADR ;CLEAR TYPE OUT CONSTANTS.
780 004020 005037 001640 CLR TOJAT
781 004024 005037 001642 CLR TGEN
782
783
784 004030 104417 TOOUTR: DELAY ;TAKE CARE OF OUTPUT ADDRS.
785 004032 012737 004032 001106 MOV #TOOUTR, $LPADR
786 004040 104416 DELAY2
787 004042 012702 013104 MOV #INADR, R2
788 004046 052777 000004 175540 BIS #4, JICSR ;ALLOW ICS TO INTR.
789 004054 005037 177776 CLR PS ;LET CPU ALLOW INTR.
790 004060 012704 013252 MOV #OUTS, R4
791 004064 012700 013144 TOOUT: MOV #OUTADR, RO ;GET OUTPUT MODULE LIST.
792 004070 012001 1S: MOV (0)+, R1 ;GET FIRST ADDR
793 004072 001405 BEQ TOIN ;IF N ADDR. - EXIT.
794 004074 013711 001604 MOV PATRN, (1) ;SEND PATTERN TO OUTPUT MODULE.
795 004100 020027 013204 CMP RO, #CNTADR
796 004104 001371 BNE 1S
797
798 004106 005737 001654 TOIN: TST CONNT ;TAKE CARE OF INPUT ADDRS.
799 004112 001407 BEQ 2S ;MODULES CONNECTED?
800 004114 012701 013252 MOV #OUTS, R1 ;IF NOT NORMAL CONTINUE.
801 004120 013721 001604 3S: MOV PATRN, (1)+ ;IF SO COS AREA = CURRENT PATTERN.
802 004124 020127 013312 CMP R1, #OUTSE
803 004130 001373 BNE 3S
804 004132 013737 001604 2S: MOV PATRN, PATJOY
805 004140 104417 DELAY ;DELAY TIME
806 004142 000240 NOP
807 004144 005737 001632 TST TPBSY
808 004150 001374 BNE .-6

```

```

809 004152 011201 1S: MOV (2),R1 ;PICK UP FIRST ADDR.
810 004154 001002 BNE +6
811 004156 000137 004636 JMP TOLOP
812 004162 021114 CMP (1),(4) ;DATA CHANGED?
813 004164 001007 BNE TOIN2 ;IF YES TAKE CARE OF IT
814 004166 004166 TOIN1=.
815 004166 005722 TST (2)+
816 004170 005724 TST (4)+ ;UPDATE COD POINTER
817 004172 020227 013144 CMP R2,#OUTADR ;DONE ALL INPUT MODULES?
818 004176 001365 BNE 1S
819 004200 000137 004636 JMP TOLOP
820
821 004204 005737 001654 TOIN2: TST CONNT
822 004210 001425 1S
823 004212 005737 001632 2S: BEQ TPBSY
824 004216 001375 BNE 2S
825 004220 010137 001122 MOV R1,$B0ADR
826 004224 013737 001610 001124 MOV PATJOY,$GDDAT
827 004232 011137 001126 MOV (1),$B0DAT
828 004236 005737 001624 TST HEADER
829 004242 001403 BEQ 3S
830 004244 104031 ERROR 31
831 004246 000137 004636 JMP TOLOP
832
833 004252 104030 3S: ERROR 30
834 004254 005237 001634 INC HEADER
835 004260 000137 004636 JMP TOLOP
836
837 004264 005737 001632 1S: TST TPBSY
838 004270 001375 BNE 1S
839 004272 012737 000340 177776 MOV #340,PS
840 004300 010137 001636 MOV R1,TOADR ;GET ADDR. OF INPUT MODULE
841 004304 011137 001640 MOV (1),TODAT ;GET CHANGE DATA
842 004310 005037 001642 CLR TOGEN ;NO GEN CODE (NO INTERRUPT)
843 004314 005237 001632 INC TPBSY ;SET TTY BUSY.
844 004320 011114 MOV (1),(4) ;RECORD NEW DATA
845 004322 104425 FOCTA ;FORM INFO INTO AN ASCIZ MESSAG.
846 004324 005046 CLR -(6)
847 004326 012746 004166 MOV #TOIN1,-(6)
848
849 004332 105777 174612 TOPINT: TSTB 2STPS ;PRINTER BUSY?
850 004336 100375 BPL TOPINT
851 004340 032777 020000 174572 BIT #020000,2SWR ;INHIBIT TYPEOUT?
852 004346 001403 BEQ 3S ;NO CONTINUE.
853 004350 005037 001632 CLR TPBSY ;YES-STOP TYPEOUT
854 004354 000002 EXIT
855 004356 112377 174570 3S: MOVB (3)+,2STPB ;SEND CHAR.
856 004362 001404 BEQ 1S ;IF END GO TO END
857 004364 052777 000100 174556 BIS #100,2STPS ;MAKE SURE TTY CAN INTR.
858 004372 000002 EXIT ;EXIT
859
860 004374 005077 174550 1S: CLR 2STPS ;TTY SETTLE DOWN TIME
861 004400 005037 001632 CLR TPBSY ;CLEAR BUSY
862
863
864 004404 000002 EXIT

```

865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920

```

      017737 175204 004630 TOINT: MOV 2ICAR,6S ;GET ADDR. OF INTRING MOD.
      096137 004630 RCL 6S ;FORM REAL ADDR.
      042737 177001 004630 BIC #177001,6S
      053737 001612 004630 BIS ICSMOD,6S
      017737 000170 004632 MOV 26S,7S ;FETCH DATA.
      005737 001632 10S: TST TPBSY ;TYPEOUT PENDING?
      001406 BEQ 5S ;IF NOT CONTINUE.
      042777 000004 175136 BIC #4, 2ICSR ;STOP ICS FROM INTR.
      005037 177776 CLR PS ;ALLOW INTERRUPTS FROM TTY.
      000767 BR 10S ;LOOP.
      012737 000300 177776 5S: MOV #300,PS ;LOCK OUT INTR.S
      052777 000005 175114 BIS #5,2ICSR ;RE-ENABLE INTR.S AND SET RIF.
      017737 175112 001642 MOV 2ICAR,TOGEN ;GET GENERIC CODE
      013737 004630 001636 MOV 6S,TOADR
      005777 175116 TST 2TOADR
      013737 004632 001640 MOV 7S,TODAT
      005737 001654 TST CONNT
      001401 BEQ 9S
      000002 EXIT

      000337 001642 9S: SWAB TOGEN ;RIGHT JUSTIFY GEN CODE.
      042737 177760 001642 BIC #177760,TOGEN
      010037 004634 MOV RO,SAVO ;SAVE RO
      012700 013104 MOV #INADR,RO ;FIND OFFSET OF INPUT MODULE
      022037 001636 2S: CMP (0)+,TOADR ;THAT INTERRUPTED.
      001404 BEQ 3S
      020027 013144 CMP RO,#OUTADR ;IF NOT ENTERED INPUT MODULE THAT
      001407 BEQ 4S ;THAN ADDR WILL NOT BE IN TABLE
      000771 BR 2S ;IN THAT CASE DON'T WORRY ABOUT IT.
      162700 013106 3S: SUB #INADR+2,RO ;SUB TO GET OFFSET.
      062700 013252 ADD #OUTS,RO ;ADD STORAGE OF COS TO RECORD
      013710 001640 MOV TODAT,(0) ;NEW DATA FOR THAT MODULE.
      013700 004634 4S: MOV SAVO,RO ;RESTORE RO.
      104425 FOCTA ;NO-FORM ASCIZ STRING.
      005237 001632 INC TPBSY ;SET OUTPUT BUSY
      000642 BR TOPINT ;START OUTPUT

      000002 1S: EXIT

      000000 6S: 0
      000000 7S: 0
      000000 SAVO: 0
      005737 001632 TOLOP: TST TPBSY
      001375 BNE TOLOP
      000004 SCOPE
      104420 CPATR
      000137 004032 JMP TOOUTR

```

;*
;*ROUTINE TO CONVERT 3 OCTAL NUMBERS TO AN ASCIZ STRING

```

921                                     ; * CALL= FOCTA
922
923 004654 012703 026604          ROCTA: MOV      #OUTBF,R3          ;SET UP BUFFER
924 004660 013737 001636 004746  MOV      TOADR,PACK1
925 004666 004737 004752          JSR      PC,PACK          ;PACK ADDR.
926 004672 013737 001640 004746  MOV      TODAT,PACK1
927 004700 004737 004752          JSR      PC,PACK          ;PACK DATA
928 004704 013737 001642 004746  MOV      TOGEN,PACK1
929 004712 001402          BEQ      IS
930 004714 004737 004752          JSR      PC,PACK          ;PACK GEN CODE (IF ANY)
931 004720 112723 000001          IS:  MOVVB  #1,(3)+      ;FILLER CHARACTERS.
932 004724 112723 000001          MOVVB  #1,(3)+
933 004730 112723 000001          MOVVB  #1,(3)+
934 004734 105023          CLRB   (3)+          ;STRING TERMINATOR.
935 004736 105023          CLRB   (3)+
936 004740 012703 026602          MOV      #OUTBF1,R3      ;RESET POINTER
937 004744 000002          EXIT
938
939 004746 000000          PACK1: 0
940 004750 000000          PACK2: 0
941
942 004752 012737 000260 004750  PACK:  MOV      #260,PACK2
943 004760 005737 004746          TST     PACK1
944 004764 100002          BPL     .+6
945 004766 005237 004750          INC     PACK2
946 004772 113723 004750          MOVVB  PACK2,(3)+
947 004776 000337 004746          SWAB   PACK1
948 005002 013737 004746 004750  MOV      PACK1,PACK2
949 005010 006037 004750          ROR     PACK2
950 005014 006037 004750          ROR     PACK2
951 005020 006037 004750          ROR     PACK2
952 005024 006037 004750          ROR     PACK2
953 005030 042737 177770 004750  BIC     #177770,PACK2
954 005036 052737 000260 004750  BIS     #260,PACK2
955 005044 113723 004750          MOVVB  PACK2,(3)+
956 005050 013737 004746 004750  MOV      PACK1,PACK2
957 005056 006037 004750          ROR     PACK2
958 005062 042737 177770 004750  BIC     #177770,PACK2
959 005070 052737 000260 004750  BIS     #260,PACK2
960 005076 113723 004750          MOVVB  PACK2,(3)+
961 005102 000337 004746          SWAB   PACK1
962 005106 013737 004746 004750  MOV      PACK1,PACK2
963 005114 006037 004750          ROR     PACK2
964 005120 006037 004750          ROR     PACK2
965 005124 006037 004750          ROR     PACK2
966 005130 006037 004750          ROR     PACK2
967 005134 006037 004750          ROR     PACK2
968 005140 006037 004750          ROR     PACK2
969 005144 042737 177770 004750  BIC     #177770,PACK2
970 005152 052737 000260 004750  BIS     #260,PACK2
971 005160 113723 004750          MOVVB  PACK2,(3)+
972 005164 013737 004746 004750  MOV      PACK1,PACK2
973 005172 006037 004750          ROR     PACK2
974 005176 006037 004750          ROR     PACK2
975 005202 006037 004750          ROR     PACK2
976 005206 042737 177770 004750  BIC     #177770,PACK2
  
```

```

977 005214 052737 000260 004750 BIS #260,PACK2
978 005222 113723 004750 MOVB PACK2,(3)+
979 005226 042737 177770 004746 BIC #177770,PACK1
980 005234 052737 000260 004746 BIS #260,PACK1
981 005242 113723 004746 MOVB PACK1,(3)+
982 005246 1.2723 000240 MOVB #240,(3)+
983 005252 112723 000240 MOVB #240,(3)+
984 005256 000207 RTS PC
985
986 005260 000137 003366 JMP START1 ;EXIT
987
988 ;*
989 ;*TEST 1 INPUT OR OUTPUT MODULE EXERCISER
990 ;*
991
992 005264 104400 TST1: TYPE ;TYPE HEADER.
993 005266 024721 MHT1
994 005270 005077 173650 CLR @STKS ;DON'T ALLOW TTY INTERRUPTS
995 005274 005737 001652 5$: TST EXPERT ;EXPERT MODE?
996 005300 001002 BNE .+6
997 005302 104400 TYPE ;ASK "INPUT OR OUTPUT MODULE?"
998 005304 021564 MIOO
999 005306 104400 TYPE
1000 005310 021764 MQ
1001 005312 005037 014754 CLR CHAR
1002 005316 105777 173622 1$: TSTB @STKS ;WAIT FOR RESPONSE
1003 005322 100375 BPL 1$
1004 005324 105777 173620 2$: TSTB @STPS ;PRINTER BUSY?
1005 005330 100375 BPL 2$
1006 005332 117777 173610 173612 MOVB @STKB,@STPB ;ECHO CHARACTER.
1007 005340 117737 173602 013042 MOVB @STKB,RTEMP
1008 005346 142737 000240 013042 BICB #240,RTEMP
1009 005354 123727 013042 000015 CMPB RTEMP,#15
1010 005362 001412 BEQ 4$
1011 005364 123727 013042 000003 CMPB RTEMP,#3 ;WAS 'C' TYPED?
1012 005372 001002 BNE .+6
1013 005374 000137 003366 JMP START1 ;IF SO GOTO MONITR.
1014 005400 113737 013042 014754 MOVB RTEMP,CHAR
1015 005406 000743 BR 1$
1016 005410 104400 4$: TYPE
1017 005412 001163 $CRLF
1018 005414 122737 000117 014754 CMPB #'0, CHAR ;DID HE TYPE "0"?
1019 005422 001424 BEQ TST10 ;IF SO-GOTO OUTPUT ROUTINE
1020 005424 122737 000111 014754 CMPB #'I, CHAR ;DID HE TYPE "I"?
1021 005432 001320 BNE 5$ ;IF NOT RETYPE QUESTION
1022
1023 ;*ROUTINE TO HANDLE INPUTTING FROM INPUT MODULE TO DISPLAY
1024
1025 005434 104412 INAR ;GET INPUT MODULE ADDR.
1026 005436 104400 024373 TYPE, MSWD
1027 005442 104400 021402 TYPE, MWK
1028
1029 005446 017700 005432 3$: MOV @INADR,RO ;GET DATA FROM INPUT MODULE
1030 005452 010037 001142 MOV RO,DISPLAY ;PUT IN DISPLAY REGISTER IF 11/45.
1031 005456 104424 INTR ;RESET TO DISPLAY.
1032 005460 032777 000001 173452 BIT #1, @SWR ;IF BIT1 SET THEN GOTO MONITR.

```

```

1033 005466 001767          BEQ      3$          ;NOT SET-THEN LOOP.
1034 005470 000137 003366  JMP      START1
1035
1036          ;*ROUTINE TO HANDLE OUTPUTTING FROM SWITCH REGISTER TO OUTPUT MODULE
1037
1038 005474 104413          TST10:  OUTAR          ;GET OUTPUT MODULE ADDR.
1039 005476 104415          DELAR          ;GET DELAY TIME (DEFAULT=3MS).
1040 005500 005037 177776    CLR      PS
1041 005504 104400 021402    TYPE,   MWK
1042 005510 012777 000100 173426    MOV     #100, 2$TKS ;ALLOW TTY INTERRUPTS
1043 005516 017777 173416 005420 1$:  MOV     2$SWR, 2$OUTADR ;SEND DATA FRO THE SWR TO OUTPUT MOD.
1044 005524 104417          DELAY
1045 005526 000773          BR      1$
1046
1047          ;*
1048          ;*TEST 2 DAC CALIBRATION ROUTINE
1049          ;*
1050
1051 005530 104400          TST2:  TYPE          ;TYPE HEADER.
1052 005532 024777          MHT2
1053 005534 104421          IDAC          ;INPUT DAC ADDR.
1054 005536 005737 013244    TST     DACADR      ;ANY DACS PRESENT
1055 005542 001005          BNE     4$
1056 005544 104400          TYPE          ;MESSAGE "NO DAC ADDR. IN BUFFER.
1057 005546 021034          MNDA
1058 005550 000137 005530    JMP     TST2
1059 005554 000000          2$:  0
1060 005556 104400 021402    4$:  TYPE,   MWK
1061 005562 017737 173352 005554 1$:  MOV     2$SWR, 2$          ;GET VALUE OF SWR
1062 005570 042737 140017 005554    BIC     #140017, 2$
1063 005576 013777 005554 005440 3$:  MOV     2$, 2$DACADR ;SEND VALUE.
1064 005604 062737 040000 005554    ADD     #40000, 2$    ;SET FOR NEXT CH.
1065 005612 103371          BCC     3$          ;DONE ALL CHS?
1066 005614 000762          BR      1$
1067
1068 005616 000137 003366  JMP      START1
1069
1070          ;*
1071          ;*TEST 3 DAC INTERACTION TEST
1072          ;*OUTPUTS RAMP TO 4 CHANNELS OF DAC THAT ARE "OUT OF PHASE"
1073          ;*
1074
1075 005622 104400          TST3:  TYPE          ;TYPE HEADER.
1076 005624 025034          MHT3
1077 005626 104421          IDAC          ;GET DAC ADDRESS
1078 005630 005737 013244    TST     DACADR      ;DAC PRESENT?
1079 005634 001004          BNE     1$
1080 005636 104400          TYPE          ;NO-MESSAGE "NO DAC ADDR. IN BUFFER"
1081 005640 021034          MNDA
1082 005642 000137 005622    JMP     TST3
1083 005646 005000          1$:  CLR     RO
1084 005650 104400 021402    TYPE,   MWK
1085 005654 010077 005364    2$:  MOV     RO, 2$DACADR ;SEND WORD TO DAC
1086 005660 062700 000020    ADD     #20, RO
1087 005664 000773          BR      2$
1088

```

MAINDEC-11-DZICA-B MACY11 27(732) 20-SEP-76 15:53 PAGE 22
 DZICAB.P11 INITIALIZE THE COMMON TAGS

```

1089 005666 000137 003366      JMP      START1
1090
1091                          ;*
1092                          ;*TEST 4 COUNTER MODULE TEST
1093                          ;*TEST 1 TO 16 MODULES
1094                          ;*
1095
1096 005672 104400      TST4:   TYPE           ;TYPE HEADER.
1097 005674 025071      MHT4
1098 005676 104422      CNTAR           ;GET COUNTER MODULE ADDRS.
1099 005700 013737 001554 001566      MOV      FR40,FREQ
1100 005706 104400 021402      TYPE,        MWK
1101
1102 005712 012700 013204      TST4L1: MOV      #CNTADR,RO  ;GET LIST OF MODULES
1103 005716 010037 004746      MOV      RO,PACK1
1104 005722 005710      TST      (0)         ;ANY ADDRS. ENTERED?
1105 005724 001004      BNE     TST4L
1106 005726 104400 022753      TYPE     MNAE       ;"NO ADDRS ENTERED".
1107 005732 000137 005672      JMP      TST4
1108
1109 005736 013700 004746      TST4L:  MOV      PACK1,RO
1110 005742 020027 013244      CMP      RO,#DACADR ;DONE ALL COUNTERS?
1111 005746 001005      BNE     2$
1112 005750 005237 001100      1$:     INC      $PASS
1113 005754 104400      TYPE
1114 005756 021063      MEND
1115 005760 000754      BR      TST4L1
1116 005762 012037 001122      2$:     MOV      (0)+,$BDADR ;GET ADDR. OF COUNTER MODULE.
1117 005766 001770      BEQ     1$          ;NOTE: ZERO IF NO ADDR. ENTERED.
1118 005770 010037 004746      MOV      RO,PACK1
1119
1120                          ;*
1121                          ;*TEST THAT THE COUNTER MODULE UNDER TEST CAN BE ADDRESSED
1122                          ;*
1123
1124 005774 012737 005774 001106      TST4A:  MOV      #TST4A,$LPADR ;SET FOR SCOPE, ITERATIONS.
1125 006002 012777 077777 173112      MOV      #77777,$BDADR ;SEND PATTERN.
1126 006010 017737 173106 001126      MOV      $BDADR,$BDDAT ;GET IT BACK
1127 006016 001001      BNE     TST4AL      ;IF NON-ZERO - WE GOT SOMETHING BACK.
1128
1129 006020 104011      ERROR   11          ;COULD NOT MAKE ANY DATA XFER
1130 006022 000004      TST4AL: SCOPE
1131
1132                          ;*
1133                          ;*TEST THAT COUNTER MODULE BITS 00 THRU 15 CAN BE SET AND CLEARED
1134                          ;*
1135
1136 006024 012737 000001 013042      TST4B:  MOV      #1,RTEMP    ;SET "GOOD DATA"
1137 006032 012737 000010 001160      MOV      #10,$TIMES
1138 006040 012737 006040 001106      1$:     MOV      #15,$LPADR  ;SET FOR ITERATIONS
1139 006046 013737 013042 001124      MOV      RTEMP,$GDDAT
1140 006054 013777 001124 173040      MOV      $GDDAT,$BDADR ;SEND PATTERN TO COUNTER MODULE.
1141 006062 017737 173034 001126      MOV      $BDADR,$BDDAT ;GET IT BACK.
1142 006070 023737 001124 001126      CMP      $GDDAT,$BDDAT ;DATA SENT=DATA RECEIVED?
1143 006076 001402      BEQ     2$
1144

```

```

1145 006100 104012          ERROR 12          ;NO - REPORT ERROR
1146 006102 000412          BR      3$          ;LOOP
1147
1149 006104 043777 001124 173010 2$: BIC      $GDDAT, $SBDADR ;TRY CLEARING BIT.
1149 006112 017737 173004 001126   MOV      $SBDADR, $BDDAT ;DID IT CLEAR?
1150 006120 001403          BEQ      3$          ;YES-LOOP.
1151 006122 005037 001124          CLR      $GDDAT
1152 006126 104012          ERROR 12
1153
1154 006130 000004          3$: SCOPE
1155 006132 006137 013042   ROL      RTEMP          ;SET TO NEXT BIT.
1156 006136 133340          BCC      1$          ;CONTINUE TESTING IF ALL BITS NOT DONE.
1157
1158          ;*
1159          ;*TEST THAT THE COUNTER MODULE CAN COUNT THRU
1160          ;*EACH STATE
1161          ;*
1162
1163 006140 012737 007014 013042 TST4C: MOV      #CNTPAT, RTEMP ;GET ADDR. OF PATTERNS
1164 006146 012737 000010 001160 1$: MOV      #10, $TIMES
1165 006154 012737 006146 001106   MOV      #1$, $SLPADR
1166
1167 006162 017777 004654 172732   MOV      @RTEMP, @SBDADR ;SEND PATTERN.
1168 006170 005037 013044          CLR      RTEMP1
1169 006174 017737 004642 001124   MOV      @RTEMP, $GDDAT
1170 006202 005237 001124          INC      $GDDAT
1171 006206 005237 013044          4$: INC      RTEMP1
1172 006212 001413          BEQ      5$          ;WATCH TO SEE IF COUNTER COUNTS.
1173 006214 017737 172702 001126   MOV      @SBDADR, $BDDAT ;IF NO COUNT BY OVERFLOW-ERROR!
1174 006222 027737 004614 001126   CMP      @RTEMP, $BDDAT
1175 006230 001766          BEQ      4$
1176 006232 023737 001124 001126   CMP      $GDDAT, $BDDAT
1177 006240 001401          BEQ      2$
1178
1179 006242 104013          5$: ERROR 13
1180 006244 000004          2$: SCOPE
1181 006246 062737 000002 013042   ADD      #2, RTEMP
1182 006254 023727 013042 007060   CMP      RTEMP, #CNTPAE
1183 006262 002731          BLT      1$
1184
1185          ;*
1186          ;*TEST TO SEE IF COUNTER INITIALIZES PROPERLY -PART 1
1187          ;*
1188
1189 006264 012737 000010 001160 TST4D: MOV      #10, $TIMES
1190 006272 012737 006272 001106 1$: MOV      #1$, $SLPADR
1191 006300 005077 172616          CLR      @SBDADR
1192 006304 104424          INTR
1193 006306 104417          DELAY
1194 006310 005037 001124          CLR      $GDDAT
1195 006314 017737 172602 001126   MOV      @SBDADR, $BDDAT
1196 006322 001401          BEQ      2$
1197
1198 006324 104025          ERROR 25
1199
1200 006326 000004          2$: SCOPE
  
```

```

1201
1202
1203
1204
1205
1206 006330 012737 000010 001160 TST4E: MOV #10,$TIMES ;SET ITERATION COUNT
1207 006336 012737 006350 001106 MOV #15,$LPADR ;SET LOOP ADDRESS.
1208 006344 005037 001124 CLR $GDDAT
1209 006350 012777 177777 172544 1$: MOV #177777,$BDADR ;SET THE COUNTER.
1210 006356 000240 NOP
1211 006360 104424 INTR ;SYSTEM INITIALIZE.
1212 006362 017737 172534 001126 MOV $BDADR,$DDAT ;READ COUNTER.
1213 006370 001401 BEQ 2$
1214
1215 006372 104020 ERROR 20
1216
1217 006374 000004 2$: SCOPE
1218
1219
1220
1221

```

```

;*
;*TEST THAT THE COUNTER INITIALIZES PROPERLY PART 2
;*

```

```

;*
;*TEST THAT THE COUNTER HAS NO
;*INTERRUPTS POSTED ON THE BUS NOR ANY GENERIC CODE

```

```

1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235 006376 105777 173212 TST4F: TSTB @ICSR ;ANY INTERRUPTS POSTED?
1236 006402 100001 BPL 1$
1237
1238 006404 104021 ERROR 21 ;ILLEGAL INTERRUPT ON FCS BUS
1239
1240 006406 000004 1$: SCOPE
1241
1242 006410 012777 006436 173202 MOV #2$,@ICSVT ;SET FOR INTERRUPT
1243 006416 012777 000404 173170 MOV #404,@ICSR ;SET MAINT + INT ENABLE
1244 006424 005037 177776 CLR PS
1245 006430 000240 NOP
1246
1247 006432 104022 ERROR 22 ;FATAL ERROR - ICS DID NOT INTERRUPT
1248 006434 000424 BR 3$
1249
1250 ;ICS INTERRUPTS TO HERE
1251
1252 006436 017737 173154 001126 2$: MOV @ICAR,$BDDAT ;READ ICAR.
1253 006444 022626 POPSP2 ;READJUST STACK
1254 006446 012777 000001 173140 MOV #1,@ICSR
1255 006454 005037 001124 CLR $GDDAT
1256 006460 005777 172436 TST @SBDDADR
1257 006464 005037 177776 CLR PS
1258 006470 042737 000360 001126 BIC #360,$BDDAT ;IGNORE FILE BOX ADDR.
1259 006476 005737 001126 TST $BDDAT ;NO OTHER ADDR. OR GEN BITS
1260 006502 001401 BEQ 3$ ;SHOULD SHOW UP
1261
1262 006504 104023 ERROR 23
1263
1264 006506 000004 3$: SCOPE
1265
1266 ;*
1267 ;*TEST THAT THE COUNTER MODULE WILL INTERRUPT ON OVERFLOW.
1268 ;*ON INTERRUPT CHECK ADDR AND GENERIC CODE, HALT ON OVERFLOW.
1269 ;*CHECK RIF EFFECT ON MODULE
1270 ;*
1271
1272 006510 012777 006576 173102 TST4G: MOV #TST4GI,@ICSVT ;SET UP FOR ICS INTERRUPT.
1273 006516 012777 000340 173076 MOV #340,@ICSVT2 ;PRIORITY 7 ON INTERRUPT.
1274 006524 012737 006532 001106 MOV #TST4GL,$LPADR ;SET FOR ITERATIONS.
1275
1276 006532 012737 000340 177776 TST4GL: MOV #340,PS ;DON'T ALLOW INTERRUPTS.
1277 006540 012777 000004 173046 MOV #4,@ICSR ;ENABLE ICS TO INTERRUPT WHEN READY.

```

```

1278 006546 012777 177777 172346      MOV      #177777, @SBADR ;SET COUNTER TO ALL ONES.
1279 006554 104417                      DELAY                      ;DELAY 40 MS.
1280 006556 005037 177776                      CLR      PS                ;ALLOW INTERRUPTS.
1281 006562 000240                      NOP
1282 006564 005077 173024                      CLR      @ICSR
1283 006570 104014                      ERROR   14                ;REPORT ERROR - COUNTER MODULE DIDN'T INTR.
1284 006572 104424                      INTR
1285 006574 000457                      BR       TST4GE           ;LOOP.
1286
1287                                ;*RECEIVE ICS INTERRUPT HERE
1288
1289 006576 012706 001100      TST4GI: MOV      #1100, SP      ;RESET THE STACK POINTER.
1290 006602 017737 173010 001126      MOV      @ICAR, $BDDAT    ;GET ADDR & GENERIC CODE.
1291 006610 013737 001122 001124      MOV      $BADR, $GDDAT    ;GET REAL ADDR.
1292 006616 006037 001124                      ROR      $GDDAT           ;FORM ADDR. AS IT WOULD LOOK IN
1293 006622 142737 177777 001125      BICB    #-1, $GDDAT+1    ;ICAR.
1294 006630 005077 172760                      CLR      @ICSR
1295 006634 005037 177776                      CLR      PS
1296 006640 052737 003400 001124      BIS     #003400, $GDDAT   ;ADD GENERIC CODE.
1297 006646 023737 001124 001126      CMP     $GDDAT, $BDDAT   ;CHECK ADDR. + GEN. CODE.
1298 006654 001403                      BEQ
1299
1300 006656 104015                      ERROR   15                ;ADDR OR GENERIC CODE INCORRECT.
1301 006660 104424                      INTR
1302 006662 000424                      BR       TST4GE           ;LOOP.
1303
1304 006664 104417                      !$:  DELAY                      ;DELAY 40 MS.
1305 006666 005037 001124                      CLR      $GDDAT
1306 006672 017737 172224 001126      MOV     @SBADR, $BDDAT
1307 006700 001402                      BEQ     2$
1308
1309 006702 104016                      ERROR   16                ;COUNTER MODULE DIDN'T HALT ON OVERFLOW.
1310 006704 000413                      BR       TST4GE           ;LOOP.
1311
1312 006706 052777 000001 172700 2$:  BIS     #1, @ICSR        ;SET RIF BIT IN ICS-11.
1313 006714 005777 172202                      TST     @SBADR           ;INITIATE RIF ON COUNTER MODULE.
1314 006720 000240                      NOP
1315 006722 105777 172666                      TSTB   @ICSR            ;ANY INTR. PENDING ON ICS BUS?
1316 006726 100002                      BPL     TST4GE           ;NO-THEN LOOP.
1317
1318 006730 104017                      ERROR   17                ;RIF DIDN'T CLEAR INTERRUPT FLAG
1319 006732 104424                      INTR
1320 006734 012777 000001 172652  TST4GE: MOV     #1, @ICSR
1321 006742 005777 172154                      TST     @SBADR
1322 006746 000004                      SCOPE
1323
1324                                ;*
1325                                ;*TEST TO SEE IF RESET CLEARS INTERRUPT FLAG ON COUNTER
1326                                ;*
1327
1328 006750 012737 000010 001160  TST4H: MOV     #10, $TIMES
1329 006756 012777 177777 172136  !$:  MOV     #-1, @SBADR    ;START COUNTER COUNTING
1330 006764 012737 006756 001106      MOV     #15, $LPAADR
1331 006772 104417                      DELAY                      ;DELAY FOR INTR.
1332 006774 104424                      INTR                      ;SYSTEM INITIALIZE
1333 006776 105777 172612                      TSTB   @ICSR            ;INTERRUPT POSTED?
  
```

```

1334 007002 100001          BPL      2$
1335
1336 007004 104024          ERROR   24          ;INIT FAILED TO CLEAR INTR. FLAG
1337 007006 000004          2$:      SCOPE
1338 007010 000137 005736          JMP      TST4L
1339
1340 007014 000000          CNTPAT: 0          ;PATTERNS USED BY TST4C TO
1341 007016 000001          1          ;DETERMINE IF COUNTER MODULE
1342 007020 000003          3          ;CAN COUNT THRU EACH STATE.
1343 007022 000007          7
1344 007024 000017          17
1345 007026 000037          37
1346 007030 000077          77
1347 007032 000177          177
1348 007034 000377          377
1349 007036 000777          777
1350 007040 001777          1777
1351 007042 003777          3777
1352 007044 007777          7777
1353 007046 017777          17777
1354 007050 037777          37777
1355 007052 077777          77777
1356 007054 177777          177777
1357 007056 100000          100000
1358 007060 000000          CNTPAE: 0
1359
1360
1361
1362
1363
1364
1365
1366          ;*
1367          ;*TEST 5 A/D LOGIC TEST
1368          ;*
1369
1370 007062 000000          ADCSR: 0          ;ADDR OF A005 UNDER TEST
1371 007064 104400          TST5:  TYPE          ;TYPE HEADER.
1372 007066 025132          MHT5
1373 007070 104423          ADAR          ;GET A005 ADDR.
1374 007072 012746 007700          ADLOGL:  MOV      #ADLOGE,-(6)
1375 007076 104400 021402          TYPE,      MWK
1376
1377 007102 013737 013246 007062          ADLOG:  MOV      ADADR,ADCSR          ;*GET A005'S ADDR.
1378 007110 052777 000100 172026          BIS      #100,0$TKS
1379 007116 005037 177776          CLR      PS
1380 007122 012737 007142 001106          MOV      #1$,SLPADR
1381 007130 012703 007714          MOV      #ADPATP,R3          ;*GET PATTERN POINTERS.
1382 007134 012737 077770 001124          MOV      #077770,$GDDAT          ;*DATA TO BE SENT
1383 007142 013777 001124 177712          1$:  MOV      $GDDAT,0ADCSR          ;*SEND DATA TO BE SENT
1384 007150 017737 177706 001126          MOV      0ADCSR,$BDDAT          ;*GET DATA BACK.
1385 007156 001001          BNE      2$          ;*IF DATA PRESENT GO AHEAD
1386
1387 007160 104001          ERROR   1          ;*ERROR "COULD NOT SEND /RECEIVE DATA"
1388
1389 007162 000004          2$:  SCOPE

```

INITIALIZE THE COMMON TAGS

```

: *BASIC "BIT BANG" OF A005
1400 007212 001401 001124 001124 ADLOG2: MOV #3, $GDDAT ; *GET PATTERN TO BE SENT.
1401 007214 001402 001124 177664 MOV $GDDAT, $ADCSR ; *SEND PATTERN TO ADCSR.
1402 007216 000004 001124 001126 MOV $ADCSR, $BDDAT ; *GET TO BACK.
1403 007220 002703 000002 001126 CMP $GDDAT, $BDDAT ; *DATA SENT=DATE RECIEVED?
1404 007224 0020327 007736 BEQ ADLOG2 ; *IF SO-CONTINUE
1405 007230 003755 ERROR 2 ; *REPORT ERROR: "CSR READ/WRITE ERROR"
1406 007232 012737 104000 001124 2$: SCOPE ; *LOOP
1407 007240 012737 007240 001106 1$: ADD #2, R3 ; *UPDATE PATTERN POINTERS
1408 007246 013777 001124 177606 CMP R3, $ADPATE ; *DONE ALL PATTERNS?
1409 007254 017737 177602 001126 BLE ADLOG2 ; *IF NOT-CONTINUE NEXT PATTERN.
1410 007262 023737 001124 001126 ; *WILL CONVERTING SET AND THEN CLEAR?
1411 007270 001402 ADLOG3: MOV #104000, $GDDAT ; *SET CONVERT, READ CSR
1412 007272 104003 1$: MOV #15, $LPAOR ; *SEND TO ADCSR
1413 007274 000411 MOV $GDDAT, $ADCSR ; *GET A005 CSR
1414 007276 013737 001560 001566 MOV $ADCSR, $BDDAT ; *DID CONVERT BIT SET?
1415 007304 104417 001560 001566 DELAY FR16, FREQ ; *ERROR: "CONVERT BIT FAILED TO SET"
1416 007306 005777 177550 TST $ADCSR ; *SET TO DELAY 16 MILLISEC.
1417 007312 100002 BPL 1$+2 ; *DID CONVERT BIT CLEAR?
1418 007314 104004 ERROR 4 ; *ERROR "CONVERT BIT FAILED TO CLEAR"
1419 007316 104424 1$: INTR ; *LOOP.
1420 007320 000004 ADLOG4: SCOPE ; *NEXT TEST.
1421 ; *ISSLE SYSTEM INITIALIZE
1422 ; *CAN WE READ THE DBR WITH CSR BIT//CLEAR (READ BIT).
1423 007322 012777 052770 177532 ADLOG5: MOV #52770, $ADCSR ; *LOAD CSR WITH ALL BUT READ CR BIT
1424 007330 022777 052770 177524 CMP #52770, $ADCSR ; *DID WE READ THE CR?
1425 007336 001001 BNE 1$ ; *IF NOT NO ERROR.
1426 007340 104005 ERROR 5 ; *REPORT ERROR "CANNOT READ A005 DATA
1427 ; *REGISTER WITH READ BIT CLEARED."
1428 007342 000004 1$: SCOPE
1429 ; *WILL CONVERT DONE CAUSE INTERRUPT?
1430 007344 012777 000340 172250 ADLOG6: MOV #340, $ICSVT2 ; *SET VECTOR FOR INTER.
1431 007352 012777 007424 172240 MOV $ADLOG7, $ICSVT ; *ALLOW ICS TO INT.
1432 007360 012777 000004 172226 MOV #4, $ICSR ; *SET A005 TO CONVERT.
1433 007366 012777 104000 177466 MOV #104000, $ADCSR

```

```

1446 007374 013737 001560 001566 MOV FR16,FREQ ; ALLOW UP TO 16 MILLI SEC FOR NTO.
1447 007402 005037 177776 CLR PS
1448 007406 104417 DELAY
1449 007410 000240 NOP
1450 007412 104006 ERROR 6 ; *REPORT ERROR "AC05 FAILED TO INTER. AT
1451 007414 005077 172174 ADL6L: CLR JICSR
1452 007420 000004 SCOPE
1453 007422 000445 BR ADLOG9
1454
1455 ; *CHECK INTR. ADDR. IN ICAR+GENERIC CODE
1456
1457 007424 022626 ADLOG7: POPSP2 ; *A005 INTR TO HERE RESET SP.
1458 007426 017737 172164 001126 MOV JICAR,$BDDAT ; *GET ADDR. AND GEN CODE
1459 007434 013737 007062 001124 MOV ADCSR,$GDDAT ; *FORM GOOD ADDR.
1460 007442 042737 177000 001124 BIC #177000,$GDDAT
1461 007450 006237 001124 ASR $GDDAT
1462 007454 052737 003400 001124 BIS #003400,$GDDAT ; *ADD GENERIC CODE
1463 007462 023737 001126 001124 JMP $BDDAT,$GDDAT ; *IS ADDR + GENERIC CODE OK?
1464 007470 001402 BEQ ADLOG8
1465
1466 007472 104007 ERROR 7 ; *REPORT ERROR "A005 ADDR. OR GENERIC
1467
1468 ; *CODE INCORRECT.
1469 007474 003747 BR ADL6L
1470
1471 ; *CHECK TO SEE IF RIF CLEARS INTR FLAG ON A005
1472
1473 007476 052777 000001 172110 ADLOG8: BIS #1,JICSR ; *SET RIF BIT.
1474 007504 005777 177352 TST JADCSR ; *PUT IT TO WORK ON A005.
1475 007510 012777 007530 172102 MOV #15,JICSVT ; *SET INTERRUPT VECTORS.
1476 007516 005037 177776 CLR PS ; *ALLOW INTERRUPTS
1477 007522 104417 DELAY
1478 007524 000240 NOP
1479 007526 000732 BR ADL6L ; *EXIT A/D TESTS.
1480
1481 007530 022626 15: POPSP2 ; *RESET SP.
1482 007532 104010 ERROR 10 ; *REPORT ERROR" RIF DID NOT CLEAR INTR.
1483 ; *FLAG ON A005".
1484 007534 000727 BR ADL6L
1485
1486 ; *
1487 ; *A005 DUAL ADDRESSING TEST
1488 ; *
1489
1490 007536 012737 007626 001106 ADLOG9: MOV #25,$LPADR
1491 007544 005037 001122 CLR $BDADR ; SET FIRST ADDR.
1492 007550 053737 001612 001122 BIS ICSMOD,$BDADR
1493 007556 162737 000002 001122 SUB #2,$BDADR
1494 007564 013737 001612 013042 MOV ICSMOD,RTEMP ; FIX STOP ADDR.
1495 007572 062737 000040 013042 ADD #40,RTEMP
1496 007600 062737 000002 001122 15: ADD #2,$BDADR ; UPDATE ADDR.
1497 007606 023737 001122 013042 CMP $BDADR,RTEMP ; DONE ALL ADDRS?
1498 007614 001427 BEQ 65
1499 007616 023737 001122 007062 CMP $BDADR,ADCSR ; SAME ADDR AS A005 UNDER TEST?
1500 007624 001765 BEQ 15
1501 007626 012777 004100 177226 25: MOV #4100,JADCSR ; SEND DATA TO A/D.

```

```

1502 007634 022777 004100 171260      CMP      #4100,0$BDADR ;DUAL ADDR. READ ERROR?
1503 007642 001001                      BNE      3$
1504
1505 007644 104026                      ERROR    26 ;DUAL ADDR. READ ERROR
1506 007646 000004                      3$:     SCOPE
1507
1508 007650 012777 004200 171244 4$:     MOV      #4200,0$BDADR ;CHECK FOR WRITE ERROR
1509 007656 022777 004200 177176      CMP      #4200,0$ADCSR ;BAD WRITE?
1510 007664 001001                      BNE      5$
1511
1512 007666 104027                      ERROR    27 ;DUAL ADDR. WRITE ERROR
1513
1514 007670 000004                      5$:     SCOPE
1515 007672 000742                      BR       1$
1516 007674 104424                      6$:     INTR
1517 007676 000207                      RTS      PC ;RETURN TO
1518 ;ADLOGE
1519
1520 ;*END OF A/D TESTS
1521
1522 007700 005237 001100      ADLOGE: INC      $PASS
1523 007704 104400      1$:     TYPE
1524 007706 021063                      MEND
1525 007710 000137 007072      JMP      ADLOGL ;*RETURN TO MONITOR.
1526
1527 007714 004000      ADPATP: 004000
1528 007716 004020      004020
1529 007720 004040      004040
1530 007722 004100      004100
1531 007724 004200      004200
1532 007726 004400      004400
1533 007730 005000      005000
1534 007732 006000      006000
1535 007734 004000      004000
1536 007736 004000      ADPATE: 004000
1537
1538 007740 000137 003366      JMP      START1
1539
1540 ;*
1541 ;*TEST 6 A/D CALIBRATION TEST
1542 ;*
1543
1544 007744 104400      TST6:  TYPE ;TYPE HEADER.
1545 007746 025166      MHT6
1546 007750 104423      ADAR ;GET A005 ADDR.
1547 007752 104426      QUBR ;UNI OR BI-POLAR?
1548 ;TSCA A0001 CALIBRATION ROUTINE
1549 ;SWR0=1 RETURN TO MONITR
1550 ;SWR4-6 SELECT CHANNEL
1551 ;SWR7-10 SELECT MUX.
1552 ;SWR11=1 TYPE OUT CONVERSIONS
1553 ;SWR11=0 DISPLAY CONVERSIONS
1554 ;SWR12-14 SELECT GAIN (1 TO 1000)
1555
1556 007754 104400 024373      TYPE,   MSWD
1557 007760 104400 021402      TYPE,   MWK

```

```

1558 007764 052777 000100 171152 TBCAA: BIS #100,JSK5 ;ALLOW TTY INTERRUPT
1559 007772 013737 013246 007062 . MOV ADADR,ADCSR
1560 010000 001004 BNE 15 ;
1561 010002 104400 TYPE
1562 010004 024344 MNAD
1563 010006 000137 003366 JMP START1
1564 010012 032777 000001 171120 15: BIT #1,JSWR ;CHECK THE SWITCH REGISTER
1565 010020 001402 BEQ +6 ;FOR RETURN TO MONITOR
1566 010022 000137 003366 JMP START1
1567 010026 004737 010646 JSR PC,CONVER ;START CONVERSION
1568 010032 017700 177024 MOV ADCSR,%0 ;GET RESULTS
1569 010036 004737 010634 JSR PC,REPET7 ;RIGHT JUSTIFY
1570 010042 032737 004000 001140 BIT #4000,SWR ;DISPLAY OR TYPE DATA?
1571 010050 001012 BNE CALTP ;TYPE IT
1572 010052 000005 RESET
1573 010054 023737 001600 001576 CMP FR1120,FR1
1574 010062 001404 BEQ 25
1575 010064 000005 RESET
1576 010066 000005 RESET
1577 010070 000005 RESET
1578 010072 000005 RESET
1579 010074 25:
1580 010074 000733 CALTP: BR TBCAA
1581 010076 104400 TYPE
1582 010100 001163 SCRLF
1583 010102 004737 010146 JSR %7,CALIT ;CONVERT DATA TO BINARY
1584 010106 013737 001550 001566 MOV FR10,FREQ
1585 010114 104417 DELAY- ;LET TTY SETTLE DOWN.
1586 010116 104400 TYPE
1587 010120 024544 MCALTI
1588 010122 017700 176734 MOV ADCSR,%0 ;GET DATA
1589 010126 004737 010634 JSR PC,REPET7
1590 010132 TYP OCT RO
1591 010132 010046 MOV RO,-(SP) ;;SAVE RO FOR TYPEOUT
1592 010134 104401 TYP OC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1593 010136 104400 TYPE
1594 010140 024523 MCALOT
1595 010142 104417 DELAY
1596 010144 000707 BR TBCAA
1597 010146 005737 013250 CALIT: TST BIPOL
1598 010152 001406 BEQ 35
1599 010154 022700 007777 CMP #7777,RO
1600 010160 001017 BNE 25
1601 010162 104400 TYPE
1602 010164 024463 MPOVFL
1603 010166 000207 RETURN
1604 010170 35:
1605 010170 022700 004000 CMP #4000,%0 ;ROUTINE TO CONVERT BINARY
1606 010174 001003 BNE 15 ;READ FROM A/D BUFFER TO SIX
1607 010176 104400 TYPE ;PLACE DECIMAL VOLTAGE
1608 010200 024434 MMOVFL
1609 010202 000207 RTS PC
1610 010204 022700 003777 15: CMP #3777,%0 ;POSITIVE OR NEGATIVE
1611 010210 001003 BNE 25
1612 010212 104400 TYPE
1613 010214 024463 MPOVFL

```

1614	010216	000207			RTS	PC			
1615	010220	012703	010516	25:	MOV	#A1,	%3		;CLEAR STORAGE AREA
1616	010224	005023			CLR	(3)+			
1617	010226	020327	010532		CMP	%3,	#A1+14		;DETERMINE IF
1618	010232	001374			BNE	.-6			
1619	010234	012703	010516		MOV	#A1,	%3		;POSITIVE OR
1620	010240	012702	002000		MOV	#2000,	%2		
1621	010244	005737	013250		TST	BIPOL			
1622	010250	001401			BEQ	.-4			
1623	010252	006302			ASL	%2			
1624	010254	012701	010540		MOV	#HOLDIT,%1			;NEGATIVE VOLTAGE
1625	010260	005737	013250		TST	BIPOL			
1626	010264	001012			BNE	CALIT1			
1627	010266	032700	004000		BIT	#4000,R0			
1628	010272	001405			BEQ	CALITP			
1629	010274	104400			TYPE				;IF MINUS TYPE "--"
1630	010276	024570			MMINUS				
1631	010300	005100			COM	%0			
1632	010302	005200			INC	%0			
1633	010304	000402			BR	CALIT1			;IF POSITIVE
1634	010306	104400			CALITP: TYPE				;TYPE "+"
1635	010310	024572			MPLUS				
1636	010312	030200			CALIT1: BIT	%2,	%0		
1637	010314	001044			BNE	CALIT2			;DO OCTAL TO
1638	010316	062701	000005		ADD	#5,	%1		
1639	010322	006002			CALITE: ROR	%2			;DECIMAL CONVERSIONS
1640	010324	001372			BNE	CALIT1			
1641	010326	005737	010514		TST	HOLDTM			;BY METHOD OF INCREMENTING
1642	010332	001403			BEQ	.-10			
1643	010334	112737	000001	010516	MOVB	#1,	A1		
1644	010342	005037	010514		CLR	HOLDTM			;DECIMAL COUNTER AND
1645	010346	012703	010516		MOV	#A1,	%3		
1646	010352	052723	000260		BIS	#260,	(3)+		
1647	010356	022703	010532		CMP	#A1+14,	%3		;DECREMENTING OCTAL DATA
1648	010362	001373			BNE	.-10			
1649	010364	012703	010517		MOV	#A1+1,	%3		;UNTIL ZERO
1650	010370	112723	000001		MOVB	#1,	(3)+		
1651	010374	105203			INCB	%3			
1652	010376	112723	000256		MOVB	#256,	(3)+		;IF DECIMAL COUNTER = 12 (OCTAL)
1653	010402	105203			INCB	%3			
1654	010404	112723	000001		MOVB	#1,	(3)+		
1655	010410	022703	010532		CMP	#A1+14,	%3		;CLEAR IT AND INCREMENT
1656	010414	001372			BNE	.-12			;NEXT COUNTER
1657	010416	105013			CLRB	(3)			
1658	010420	104400			TYPE				;TYPE DECIMAL
1659	010422	010516			A1				
1660	010424	000207			RTS	%7			
1661	010426	005037	010514		CALIT2: CLR	HOLDTM			
1662	010432	012703	010532		MOV	#A1+14,	%3		;FIRST USE TABLE HOLDIT
1663	010436	112137	010512		CALIT3: MOVB	(1)+,	SCAN		;TO GET DECIMAL VALUE FOR DATA
1664	010442	063743	010512		ADD	SCAN,	-(3)		;COMPUTED FROM ADDS
1665	010446	005737	010514		TST	HOLDTM			
1666	010452	001404			BEQ	CALIT4			
1667	010454	005037	010514		CLR	HOLDTM			;ADD THESE VALUES TOGETHER
1668	010460	062713	000001		ADD	#1,	(3)		
1669	010464	122713	000012		CALIT4: CMPB	#12,	(3)		;HOLDIT CONTAINS VALUES FOR DATA

1670	010470	101004				BHI	CALITS		
1671	010472	162713	000012			SUB	#12	(3)	;BASED ON GAIN OF A/D
1672	010476	005237	010514			INC	HOLDTM		
1673	010502	022703	010520			CALITS: CMP	#AI+2	%3	;AT TIME DATA WAS TAKEN
1674	010506	001353				BNE	CALIT3		
1675	010510	000704				BR	CALITE		
1676	010512	000000				SCAN:			
1677	010514	000000				HOLDTM:	000000		
1678	010516	000000				AI:	000000		
1679		010540				.+.20			
1680	010540	000	000	000		HOLDIT:	.BYTE 0,0,0,0,5		;TABLE OF VOLTAGE REPRESENTATION
1681	010543	000	005						
1682									
1683	010545	000	000	000		.BYTE	0,0,0,5,2		;OF BINARY INPUT.
1684	010550	005	002						
1685									
1686	010552	000	000	005		.BYTE	0,0,5,2,1		
1687	010555	002	001						
1688									
1689	010557	000	005	002		.BYTE	0,5,2,6,0		
1690	010562	006	000						
1691									
1692	010564	005	002	001		.BYTE	5,2,1,3,0		
1693	010567	003	000						
1694									
1695	010571	003	006	005		.BYTE	3,6,5,1,0		
1696	010574	001	000						
1697									
1698	010576	001	010	007		.BYTE	1,8.,7,0,0		
1699	010601	000	000						
1700									
1701	010603	001	011	003		.BYTE	1,9.,3,0,0		
1702	010606	000	000						
1703									
1704	010610	005	011	001		.BYTE	5,9.,1,0,0		
1705	010613	000	000						
1706									
1707	010615	010	011	000		.BYTE	8.,9.,0,0,0		
1708	010620	000	000						
1709									
1710	010622	011	004	000		.BYTE	9.,4,0,0,0		
1711	010625	000	000						
1712									
1713	010627	004	002	000		.BYTE	4,2,0,0,0		
1714	010632	000	000						
1715									
1716						.EVEN			
1717									
1718	010634	006000				REPET7:	ROR	%0	
1719	010636	006000					ROR	%0	;ROUTINE TO RIGHT
1720	010640	006000					ROR	%0	;JUSTIFY DATA BY
1721	010642	006000					ROR	%0	;ROTATING IT 4 PLACE TO
1722	010644	000207					RTS	PC	;THE RIGHT
1723	010646	013702	001140			CONVER:	MOV	SWR,R2	
1724	010652	042702	000017				BIC	#17,%2	;THISROUTINE SAMPLES SWR
1725	010656	052702	104000				BIS	#104000,%2	;AND SENDS CHANNEL, GAIN

```

1726 010662 010277 176174          MOV      %2,      @ADCSR ; CONVERT AND READ BITS TO ADU01
1727 010666 005777 176170          CONVED: TST      @ADCSR
1728 010672 100775                   BMI      -4          ; TO START CONVERSION
1729 010674 005077 176162          CLR      @ADCSR    ; AND WAIT FOR CONVERSION COMPLETE
1730 010700 000207                   RTS      PC
1731
1732 010702 000137 003366          JMP      START1
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749 010706 104400          TST7:   TYPE                    ; TYPE HEADER
1750 010710 025223          MHT8
1751 010712 104423          ADAR
1752 010714 104426          QUBR
1753 010716 013737 013246 007062      MOV      @ADAR,ADCSR
1754 010724 013737 001550 001566      MOV      @R110,FREQ
1755 010732 104417          DELAY                    ; ALLOW TTY TO SETTLE BEFORE CONT.
1756 010734 004737 007102          JSR      PC,ADLOG       ; BASIC LOGIS CHECK
1757 010740 005737 001652          22$:   TST      EXPERT
1758 010744 001002          BNE      +6
1759 010746 104400          TYPE                    ; ASK FOR GAIN
1760 010750 025361          MGAIN
1761 010752 012737 013006 010774      MOV      #GAIN,10$
1762 010760 012737 000001 010776      MOV      #1,10$+2
1763 010766 000401          BR      99$
1764 010770 000763          BR      22$
1765 010772 104411          99$:   INOCT                    ; GET GAIN
1766 010774 013006          10$:   GAIN
1767 010776 000001          I
1768
1769 011000 012700 013062          20$:   MOV      #GLIST,R0       ; GET LIST OF LEGAL GAINS.
1770 011004 005710          TST      (0)            ; AT END OF LIST?
1771 011006 001003          BNE      21$           ; NO-CONTINUE.
1772 011010 104400          TYPE                    ; GAIN HE TYPED IS KNOWN.
1773 011012 025731          MSG8                    ; TELL HIM.
1774 011014 000751          BR      22$           ; REASK QUESTION.
1775 011016 012001          21$:   MOV      (0)+,R1     ; GET GAIN FROM GAIN LIST.
1776 011020 042701 170000          BIC      #170000,R1    ; MASK OUT REAL GAIN BITS.
1777 011024 020137 013006          CMP      R1,GAIN       ; GAIN HE TYPED MATCH ONE IN GAIN LIST?
1778 011030 001365          BNE      20$           ; NO-CHECK NEXT IN LIST.
1779 011032 014037 013006          MOV      -(0),GAIN     ; YES-REPLACE TYPED GAIN BY REAL GAIN.
1780
1781 011036 005737 001652          3$:   TST      EXPERT
  
```

1782	011042	001002			BNE	.+6	
1783	011044	104400			TYPE		;ASK FOR CHANS:
1784	011046	025376			MCHAN		
1785	011050	005037	013012		CLR	CHANS	
1786	011054	005037	013014		CLR	CHANF	
1787	011060	005037	013016		CLR	CHANSR	
1788	011064	005037	013020		CLR	CHANFR	
1789	011070	013737	001650	013042	MOV	CORSIZ,RTEMP	
1790	011076	162737	026636	013042	SUB	#SENDAD,RTEMP	
1791	011104	005037	013044		CLR	RTEMP1	
1792	011110	005237	013044		INC	RTEMP1	
1793							
1794	011114	162737	001000	013042	SUB	#1000,RTEMP	
1795	011122	100372			BPL	15\$	
1796	011124	005337	013044		DEC	RTEMP1	
1797	011130	013737	013044	013022	MOV	RTEMP1,CHANNO	
1798	011136	013746	013044		MOV	RTEMP1,-(SP)	
1799	011142	004737	016346		GOSUB	,\$\$B20	
1800	011146	062716	000010		ADD	#10,(SP)	
1801	011152	005726			TST	(SP)+	
1802	011154	012737	013016	011176	MOV	#CHANSR,11\$	
1803	011162	012737	000002	011200	MOV	#2,11\$+2	
1804	011170	000401			BR	+4	
1805	011172	000721			BR	3\$	
1806	011174	104411			INOC		;GET CHANS
1807	011176	013016			CHANSR		
1808	011200	000002			2		
1809	011202	005737	013020		TST	CHANFR	
1810							
1811	011206	001003			BNE	.+10	
1812	011210	013737	013016	013020	MOV	CHANSR,CHANFR	
1813	011216	023737	013016	013020	CMP	CHANSR,CHANFR	;CHAN. S+5?
1814	011224	003403			BLE	4\$	
1815	011226	104400			TYPE		
1816	011230	025445			MCHER1		
1817	011232	000701			BR	3\$	
1818	011234	023727	013020	000177	4\$: CMP	CHANFR,#177	;CHAN WITHIN LEGAL BOUNDS"
1819	011242	003403			BLE	30\$	
1820	011244	104400			TYPE		;NO-THEN TELL HIM
1821	011246	025651			MCHANH		
1822	011250	000672			BR	3\$	
1823	011252	005737	001652		30\$: TST	EXPERT	
1824	011256	001002			BNE	.+6	
1825	011260	104400			TYPE		;ASK FOR EXPECTED AVERAGE.
1826	011262	025700			MAVEQ		
1827	011264	012737	013032	011306	MOV	#AVEXP,12\$	
1828	011272	012737	000001	011310	MOV	#1,12\$+2	
1829	011300	000401			BR	+4	
1830	011302	000763			BR	30\$	
1831	011304	104411			INOC		;GET AVERAGE.
1832	011306	013032			AVEXP		
1833	011310	000001			1		
1834	011312	032737	170000	013032	BIT	#170000,AVEXP	;LEGAL AVERAGE"
1835	011320	001403			BEQ	6\$	
1836	011322	104400			TYPE		;NO TELL HIM(OR HER) ASK:
1837	011324	025751			MNTL		;QUESTION AGAIN.

1838	011326	C00751				BR	30\$	
1839	011330	012737	013030	011410	6\$:	MOV	#TOLER,13\$	
1840	011336	012737	000001	011412		MOV	#1,13\$+2	
1841	011344	032737	004000	013032		BIT	#4000,AVEXP	
1842	011352	001406				BEQ	60\$	
1843	011354	005737	013250			TST	BIPOL	
1844	011360	001003				SNE	60\$	
1845	011362	052737	170000	013032		BIS	#170000,AVEXP	
1846	011370	005737	001652		60\$:	TST	EXPERT	
1847	011374	001002				BNE	+.6	
1848	011376	104400				TYPE		;ASK FOR TOLERANCE.
1849	011400	025423				MTOL		
1850	011402	000401				BR	+.4	
1851	011404	000771				BR	60\$	
1852	011406	104411				INOCT		
1853	011410	013030			13\$:	TOLER		
1854	011412	000001				I		
1855	011414	005737	001660			TST	LINEPR	
1856	011420	001411				BEQ	14\$	
1857	011422	013737	001662	001150		MOV	LPCSR,\$TPS	
1858	011430	013737	001664	001152		MOV	LPDBR,\$TPB	
1859	011436	104400	021402			TYPE,	MWK	
1860	011442	000402				BR	+.6	
1861	011444	104400			14\$:	TYPE		;TYPE "REPEAT"
1862	011446	024512				MREP		
1863	011450	013737	013016	013014		MOV	CHANSR,CHANF	
1864	011456	005337	013014			DEC	CHANF	
1865	011462	032777	000001	167450		BIT	#1,\$SWR	
1866	011470	001402				BEQ	+.6	
1867	011472	000137	003366			JMP	START1	
1868	011476	013737	013014	013012	5\$:	MOV	CHANF,CHANS	
1869	011504	005237	013012			INC	CHANS	
1870	011510	063737	013022	013014		ADD	CHANNO,CHANF	
1871	011516	023737	013012	013020		CMP	CHANS,CHANFR	
1872	011524	003347				BGT	14\$	
1873	011526	023737	013014	013020		CMP	CHANF,CHANFR	
1874	011534	003403				BLE	+.10	
1875	011536	013737	013020	013014		MOV	CHANFR,CHANF	
1876	011544	004737	011556			GOSUB	,SAMPR	;TAKE CONVERSIONS
1877	011550	004737	012060			GOSUB	,AVERR	;TAKE AVERAGES
1878	011554	000750				BR	5\$	
1879								
1880						;	*	
1881						;	*ROUTINE TO TAKE 256 SAMPLES OF AL CHANNELS SPECIFIED	
1882						;	*CALL=GOSUB,SAMPR	
1883								
1884	011556	013737	001552	001566	SAMPR:	MOV	FR50,FREQ	;SUB PART TO SET UP DELAY BASEDON
1885	011564	013737	013014	013004		MOV	CHANF,CHAN	
1886	011572	163737	013012	013004		SUB	CHANS,CHAN	;NUMBER OF CHANNELLS WERE SAMPLING.
1887	011600	013737	013004	013026		MOV	CHAN,SAMOFF	
1888	011606	001416				BEQ	2\$	
1889	011610	006337	013026			ASL	SAMOFF	
1890	011614	163737	001562	001566	1\$:	SUB	FR5,FREQ	
1891	011622	005337	013004			DEC	CHAN	
1892	011626	001372				BNE	1\$	
1893	011630	005737	001566			TST	FREQ	

```

1894 011634 100003          BPL      2$
1895 011636 012737 000001 001566    MOV      #1,FREQ
1896 011644 012737 177400 013024 2$:    MOV      #-256,SAMCNT ;SET SAMPLE COUNT.
1897 011652 012701 026636          MOV      #BUFFER,R1 ;SET FOR STORAGE.
1898 011656 013737 013012 013004 3$:    MOV      CHANS,CHAN ;GET STARTING CHANNEL
1899 011664 104417          DELAY
1900 011666 013737 013004 013002 4$:    MOV      CHAN,CHAN1 ;SET TO RIGHT JUSTIFY CHAN.
1901 011674 006337 013002          ASL      CHAN1 ;FIX TO LOAD INTO AID WORD.
1902 011700 006337 013002          ASL      CHAN1
1903 011704 006337 013002          ASL      CHAN1
1904 011710 006337 013002          ASL      CHAN1
1905 011714 004737 011752          GOSUB    ,CONVT ;TAKE CONVERSION.
1906 011720 010021          MOV      RO,(1)+ ;STORE RESULT
1907 011722 005237 013004          INC      CHAN ;READY FOR NEXT CHAN.
1908 011726 023737 013004 013014    CMP      CHAN,CHANF ;DONE ALL CHANNELLS.
1909 011734 003754          BLE      4$
1910 011736 005237 013024          INC      SAMCNT ;DONE 256 SAMPLES?
1911 011742 001345          BNE      3$
1912 011744 012701 026636    MOV      #BUFFER,R1
1913 011750 000207          RETURN
1914
1915 ;*ROUTINE TO FORM A AID WORD FOR CONVERSIONS, START A CONVERSION,
1916 ;*WAIT FOR DONE, AND ENABLE READING OF THE AID DBR.
1917
1918 ;*CALL=GOSUB,CONVT,RESULT FOUND IN RO,RIGHT JUSTIFIED
1919 011752 013737 013006 013010 CONVT: MOV      GAIN,ADWD ;SET GAIN IN WORD
1920 011760 042737 007777 013010    BIC      #7777,ADWD
1921 011766 053737 013002 013010    BIS      CHAN1,ADWD ;SET CHAN.
1922 011774 052737 104000 013010    BIS      #104000,ADWD ;SET READ BIT, CONVERT BIT.
1923 012002 013777 013010 175052    MOV      ADWD,ADCSR ;START CONVERSION.
1924 012010 005777 175046          1$:    TST      ADCSR ;WAIT FOR CONVERSION COMPLETE.
1925 012014 100775          BMI      1$
1926 012016 005077 175040    CLR      ADCSR ;ENABLE READING OF DBR.
1927 012022 017700 175034    MOV      ADCSR,RO ;READ AID RESULTS PUT IN RO
1928 012026 005737 013250    TST      BIPOL
1929 012032 001405          BEQ      2$
1930 012034 006000          ROR      RO
1931 012036 006000          ROR      RO
1932 012040 006000          ROR      RO
1933 012042 006000          ROR      RO
1934 012044 000207          RETURN
1935 012046 006200          2$:    ASR      RO ;RIGHT JUSTIFY, REMEMBERING SIGN.
1936 012050 006200          ASR      RO
1937 012052 006200          ASR      RO
1938 012054 006200          ASR      RO
1939 012056 000207          RETURN ;RETURN
1940
1941 ;*
1942 ;*AVERAGING ROUTINE USED BE TEST 7
1943 ;*AT THIS POINT IN TIME ALL SAMPLES FOR ALL CHANNELS HAVE
1944 ;*BEEN TAKEN AND STORED IN "BUFFER" IN A
1945 ;*SEQUENTIAL INTERLEAVED BUFFER FORM" GIVEN BY THE FOLLOWING FORMULA:
1946 ;*"L=2*N+R1" WHERE L EQUALS THE LOCATION OF A SAMPLE,
1947 ;*N EQUALS THE NUMBER OF CHANNELS SAMPLES WERE TAKEN ON, AND R1
1948 ;*IS THE BUFFER POINTER SET TO "BUFFER" INITIALLY.
1949 ;*CALL=GOSUB,AVERR

```

```

1950 012060 013737 013012 013004 AVERR: MOV CHANS,CHAN ;SET TO FIRST CHAN.
1951 012066 005037 013002 CLR CHANI ;1ST CHAN OFFSET
1952 012072 012737 000400 013024 AVERAL: MOV #256.,SAMCNT
1953 012100 013701 013002 MOV CHANI,R1 ;PUT CHAN OFFSET IN R1.
1954 012104 062701 026636 ADD #BUFFER,R1 ;ADD BUFFER POINTER TO R1.
1955 012110 005037 013034 CLR AVTKN ;SET INITIAL CONDITIONS FOR THIS CHAN.
1956 012114 005037 013042 CLR RTEMP
1957 012120 011137 013036 MOV (1),RLOW
1958 012124 011137 013040 MOV (1),RHIGH
1959 012130 023711 013040 2$: CMP RHIGH,(1) ;FIND REAL HIGH VALUE.
1960 012134 003002 BGT 3$
1961 012136 011137 013040 MOV (1),RHIGH
1962 012142 021137 013036 3$: CMP (1),RLOW ;FIND REAL LOW VALUE.
1963 012146 003002 BGT 4$
1964 012150 011137 013036 MOV (1),RLOW
1965 012154 012137 013044 4$: MOV (1),RTEMP1 ;GET CURRENT SAMPLE.
1966 012160 005737 013250 TST BIPOL
1967 012164 001003 BNE .+10
1968 012166 062737 004000 013044 ADD #4000,RTEMP1 ;ADD CONSTANT TO SAMPLE.
1969 012174 063737 013044 013034 ADD RTEMP1,AVTKN ;"BOOT" ADD ALL SAMPLES.
1970 012202 005537 013042 ADC RTEMP
1971 012206 063701 013026 ADD SAMOFF,R1 ;UPDATE TO LOOK AT NEXT CHAN SAMPLE
1972 012212 005337 013024 DEC SAMCNT ;DONE ALL SAMPLES PER THIS CHAN?
1973 012216 001344 BNE 2$ ;NO-DO NEXT SAMPLE.
1974 012220 013737 013032 013044 MOV AVEXP,RTEMP1 ;YES! SEE IF RHIGH OK:
1975 012226 063737 013030 013044 ADD TOLER,RTEMP1
1976 012234 023737 013044 013040 CMP RTEMP1,RHIGH
1977 012242 002427 BLT ERAV1 ;NO-THEN REPORT ERROR.
1978 012244 163737 013030 013044 SUB TOLER,RTEMP1 ;YES-OK-CHECK LOWEST READING.
1979 012252 163737 013030 013044 SUB TOLER,RTEMP1
1980 012260 023737 013036 013044 CMP RLOW,RTEMP1
1981 012266 002415 BLT ERAV1 ;NO-REPORT ERROR.
1982 012270 005737 013030 TST TOLER ;DOES OPERATOR WISH "FORCED" TYPEOUT?
1983 012274 001415 BEQ ERAV2 ;IF SO-DO IT
1984 012276 062737 000002 013002 AVERRN: ADD #2,CHANI ;SET TO DO NEXT CHAN-BUT
1985 012304 005237 013004 INC CHAN ;IF DONE ALL CHANS-EXIT-
1986 012310 023737 013004 013014 CMP CHAN,CHANF ;OTHERWISE LOOP.
1987 012316 003665 BLE AVERAL
1988 012320 000207 RETURN
1989
1990 ;*ERROR REPORTER
1991
1992 012322 104400 ERAV1: TYPE ;TYPE "REPEATIBILITY ERROR"
1993 012324 026045 MREPER
1994 012326 000402 BR .+6
1995 012330 104400 ERAV2: TYPE ;TYPE "REPEATIBILITY FORCED TYPEOUT"
1996 012332 026007 MREPFT
1997 012334 012702 000007 1$: MOV #7,R2 ;FIND AVERAGE.
1998 012340 006237 013042 10$: ASR RTEMP
1999 012344 006037 013034 ROR AVTKN ;AVERAGE=TOTAL OF SAMPLES-
2000 012350 005302 DEC R2 ;DIVIDED BY 256.
2001 012352 001372 BNE 10$
2002 012354 006237 013042 ASR RTEMP
2003 012360 006037 013034 ROR AVTKN
2004 012364 005537 013034 ADC AVTKN
2005 012370 005737 013250 TST BIPOL

```

2006	012374	001003			BNE	+.10	
2007							
2008	012376	162737	004000	013034	SUB	#4000,AVTKN	;SUBTRACT CONSTANT.
2009	012404	104400			TYPE		;TYPE HEADER 1
2010	012406	026074			MREPT1		
2011	012410				TYPOCT	CHAN	;TYPE CHAN.
2012	012410	013746	013004		MOV	CHAN,-(SP)	;SAVE CHAN FOR TYPEOUT
2013	012414	104401			TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
2014	012416	104400			TYPE		
2015	012420	026004			M2SP		
2016	012422	013702	013006		MOV	GAIN,R2	;TYPE GAIN.
2017	012426	042702	170000		BIC	#170000,R2	
2018	012432				TYPOCT	R2	
2019	012432	010246			MOV	R2,-(SP)	;SAVE R2 FOR TYPEOUT
2020	012434	104401			TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
2021	012436	104400			TYPE		
2022	012440	026004			M2SP		
2023	012442	013702	013036		MOV	RLOW,R2	
2024	012446	042702	170000		BIC	#170000,R2	
2025	012452				TYPOCT	R2	;TYPE LOWEST SAMPLE TAKEN.
2026	012452	010246			MOV	R2,-(SP)	;SAVE R2 FOR TYPEOUT
2027	012454	104401			TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
2028	012456	104400			TYPE		
2029	012460	026004			M2SP		
2030							
2031							
2032							
2033							
2034							
2035							
2036							
2037	012462	013702	013034		MOV	AVTKN,R2	
2038	012466	042702	170000		BIC	#170000,R2	
2039	012472				TYPOCT	R2	;TYPE AVERAGE OF SAMPLES TAKEN.
2040	012472	010246			MOV	R2,-(SP)	;SAVE R2 FOR TYPEOUT
2041	012474	104401			TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
2042	012476	104400			TYPE		
2043	012500	026004			M2SP		
2044	012502	013702	013040		MOV	RHIGH,R2	
2045	012506	042702	170000		BIC	#170000,R2	
2046	012512				TYPOCT	R2	;TYPE HIGHEST SAMPLE TAKEN.
2047	012512	010246			MOV	R2,-(SP)	;SAVE R2 FOR TYPEOUT
2048	012514	104401			TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
2049	012516	104400			TYPE		;TYPE SECOND HEADER.
2050	012520	026146			MREPT2		
2051	012522	013737	013034	013042	MOV	AVTKN,RTEMP	;GET AV.
2052	012530	162737	000005	013042	SUB	#5,RTEMP	;MAKE -5 POINT TO GET LOW POINT.
2053	012536	012737	000400	013024	MOV	#256.,SAMCNT	;SET # OF SAMPLES.
2054	012544	013701	013002		MOV	CHAN1,R1	
2055	012550	062701	026636		ADD	#BUFFER,R1	
2056	012554	005002			CLR	R2	
2057	012556	022137	013042	2\$:	CMP	(1)+,RTEMP	
2058	012562	002001			BGE	3\$	
2059	012564	005202			INC	R2	
2060	012566	063701	013026	3\$:	ADD	SAMOFF,R1	
2061	012572	005337	013024		DEC	SAMCNT	

```

2062 012576 001367      BNE      2$
2063 012600 004737 012744  GOSUB   ,AVTYP      ;TYPE RLOW # OF SAMPLES.
2064 012604 012703 177765      MOV     #-11.,R3      ;SET TO DO 11 TIMES.
2065 012610 012737 000400 013024 4$:  MOV     #256.,SAMCNT ;SET SAMPLE COUNT AT 256.
2066 012616 005002      CLR     R2
2067 012620 013701 013002      MOV     CHAN1,R1      ;GET BUFFER POINTER.
2068 012624 062701 026636      ADD     #BUFFER,R1
2069 012630 022137 013042      5$:  CMP     (1)+,RTMP    ;SAMPLE=COUNT POINTER?
2070 012634 001001      BNE     6$
2071 012636 005202      INC     R2
2072 012640 063701 013026      6$:  ADD     SAMOFF,R1
2073 012644 005337 013024      DEC     SAMCNT        ;CHECKED ALL SAMPLES?
2074 012650 001367      BNE     5$
2075 012652 004737 012744  GOSUB   ,AVTYP      ;TYPE # OF SAMPLES AT THIS POINT.
2076 012656 005237 013042      INC     RTMP          ;MOVE TO NEXT POINT.
2077 012662 005203      INC     R3            ;DONE ALL POINTS?
2078 012664 001351      BNE     4$            ;NO-THEN LOOP
2079 012666 005002      CLR     R2            ;FIND ALL OVERSCALE POINTS.
2080 012670 012737 000400 013024  MOV     #256.,SAMCNT
2081 012676 013701 013002      MOV     CHAN1,R1
2082 012702 062701 026636      ADD     #BUFFER,R1
2083 012706 022137 013042      7$:  CMP     (1)+,RTMP
2084 012712 003401      BLE     8$
2085 012714 005202      INC     R2
2086 012716 063701 013026      8$:  ADD     SAMOFF,R1
2087 012722 005337 013024      DEC     SAMCNT
2088 012726 001367      BNE     7$
2089 012730 004737 012744  GOSUB   ,AVTYP
2090 012734 104400      TYPE
2091 012736 001163      $CRLF
2092 012740 000137 012276      JMP     AVERRN
2093 012744 010246      AVTYP: MOV     R2,-(SP)    ;PUT # ON STACK
2094 012746 004737 016346      GOSUB   $S820        ;GOTO OCTAL-BCD ROUTINE.
2095 012752 062716 000007      ADD     #7,(SP)      ;GET RID OF 1ST.2 DIGITS
2096 012756 012637 012764      MOV     (SP)+,1$    ;TYPE STRING.
2097 012762 104400      TYPE
2098 012764 000000      1$:  0
2099 012766 104400      TYPE                ;TYPE 2 SPACES.
2100 012770 026004      M2SP
2101 012772 000207      RETURN
2102
2103      ;*
2104      ;*POINTERS USED BY TEST 7 REPEATIBILITY TEST
2105      ;*
2106
2107 012774 000000      CHAN7:0
2108 012776 000000      NA07:0
2109 013000 000000      NA17:0
2110 013002 000000      CHAN1: 0
2111 013004 000000      CHAN:  0
2112 013006 000000      GAIN:  0
2113 013010 000000      ADWD:  0
2114 013012 000000      CHANS: 0
2115 013014 000000      CHANF: 0
2116 013016 000000      CHANSR:0
2117 013020 000000      CHANFR:0
;LEFT JUSTIFIED CURRENT CHANNELL
;CURRENT CHANNELL
;CURRENT GAIN
;WORD SENT TO A/D
;STARTING CHANNEL
;LAST CHANNEL

```

```

013020 000000
013022 000000
013024 000000
013026 000000
013028 000000
013030 000000
013032 000000
013034 000000
013036 000000
013038 000000
013040 000000
013042 000000
013044 000000
013046 000000
013048 000000
013050 000000
013052 000000
013054 000000
013056 000
013060 000000
013062 071000
013064 060200
013066 050100
013070 040050
013072 030020
013074 020010
013076 010002
013100 000001
013102 000000
013104 000000
013106 000000
013110 000000
013112 000000
013114 000000
013116 000000
013120 000000
013122 000000
013124 000000
013126 000000
013130 000000
013132 000000
013134 000000
013136 000000
013140 000000
013142 000000
013144 000000
013146 000000
013150 000000
013152 000000
013154 000000
013156 000000
013160 000000

```

050

```

CHANNO: 0
SAMPNT: 0
SAMPFF: 0
TOLER: 0
AVEXP: 0
AVTAK: 0
ALOW: 0
AHIGH: 0
ATEMP: 0
RTEMP: 0
REPMA: 0

```

```

: SAMPLE COUNT
: SAMPLE OFFSET
: TOLERANCE BEFORE ERROR IS REPORTED
: EXPECTED AVERAGE
: AVERAGE OF SAMPLES TAKEN
: LOWEST SAMPLE TAKEN
: HIGHEST SAMPLE TAKEN

```

```

: SECTION RESERVED FOR
: MANUAL REPEATABILITY
: WHERE OPERATOR ENTERS
: IN INFORMATION

```

* GAIN LIST

```

GLIST: 071000 : GAIN OF 1000.
        060200 : GAIN OF 200.
        050100 : GAIN OF 100.
        040050 : GAIN OF 50.
        030020 : GAIN OF 20.
        020010 : GAIN OF 10.
        010002 : GAIN OF 2.
        000001 : GAIN OF 1.
        000000 : ILLEGAL GAIN.

```

* INPUT MODULE ADDRESSES

```

INADR: 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0

```

* OUTPUT MODULE ADDRESSES

```

OUTADR: 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0

```

2174 013162 000000
2175 013164 000000
2176 013166 000000
2177 013170 000000
2178 013172 000000
2179 013174 000000
2180 013176 000000
2181 013200 000000
2182 013202 000000
2183
2184 013204 000000
2185 013206 000000
2186 013210 000000
2187 013212 000000
2188 013214 000000
2189 013216 000000
2190 013220 000000
2191 013222 000000
2192 013224 000000
2193 013226 000000
2194 013230 000000
2195 013232 000000
2196 013234 000000
2197 013236 000000
2198 013240 000000
2199 013242 000000
2200
2201 013244 000000
2202
2203 013246 000000
2204 013250 000000
2205
2206 013252 000000
2207 013254 000000
2208 013256 000000
2209 013260 000000
2210 013262 000000
2211 013264 000000
2212 013266 000000
2213 013270 000000
2214 013272 000000
2215 013274 000000
2216 013276 000000
2217 013300 000000
2218 013302 000000
2219 013304 000000
2220 013306 000000
2221 013310 000000
2222 013312 000000
2223
2224
2225
2226
2227
2228 013314 005737 001652
2229 013320 001002

.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
*COUNTER MODULE ADDRESSES
CNTADR: 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
* DAC MODULE ADDRESS
DACADR: 0
*AQS MODULE ADDRESS
AQADR: 0
BIPOL: 0 ;0=BIPOLAR-1=UNI
*TEMP STORAGE OF INPUT MODULE DATA
OUTS: 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
OUTSE: 0
*
*ROUTINE TO INPUT INPUT MODULE ADDRESS.
*
RINA: TST EXPERT
BNE .+6

```

2230 013322 104400 TYPE ;ASK OPERATOR "INPUT MODULE ADDR(S)?"
2231 013324 020661 MIPA
2232 013326 012737 013104 013350 MOV #INADR, RINA1
2233 013334 012737 000020 013352 MOV #16., RINA1+2
2234 013342 000401 BR .+4
2235 013344 000763 BR RINA
2236 013346 104411 ;CALL TO INPUT OCTAL ROUTINE
2237 013350 013104 RINA1: INOCT ;ADDR TO STORE INPUT ADDRESS AT (VARIES).
2238 013352 000020 INADR ;NUMBER OF ADDRESS ALLOWED.
2239 013354 005737 014754 TST CHAR
2240 013360 001411 BEQ 25
2241 013362 012700 013104 MOV #INADR, RO ;NOW FORM REAL ICS ADDRESS.
2242 013366 053710 001612 15: BIS ICSMOD, (0) ;NORMAL-ICS MOD=171000 UNLESS PATCHED
2243 013372 004737 015416 JSR PC,CKADR
2244 013376 020037 013350 CMP RO,RINA1 ;DONE ALL ADDRS?
2245 013402 101771 BLOS 15 ;IF NOT CONTINUE.
2246 013404 000002 25: EXIT ;IF SO-EXIT
2247
2248
2249 ;*
2250 ;*ROUTINE TO INPUT OUTPUT MODULE ADDRESSES
2251 ;*
2252 013406 005737 001652 ROUTA: TST EXPERT
2253 013412 001002 BNE .+6
2254 013414 104400 TYPE ;ASK OPERATOR "OUTPUT MODULE ADDRESSES"
2255 013416 020717 MOPA
2256 013420 012737 013144 013442 MOV #OUTADR, ROUTA1
2257 013426 012737 000020 013444 MOV #16., ROUTA1+2
2258 013434 000401 BR .+4
2259 013436 000763 BR ROUTA
2260 013440 104411 ;CALL TO INPUT OCTAL ROUTINE
2261 013442 013144 ROUTA1: OUTADR ;ADDR TO STORE OUTPUT ADDRS.
2262 013444 000020
2263 013446 005737 014754 TST CHAR
2264 013452 001411 BEQ 25
2265 013454 012700 013144 MOV #OUTADR, RO ;
2266 013460 053710 001612 15: BIS ICSMOD, (0)
2267 013464 004737 015416 JSR PC,CKADR
2268 013470 020037 013442 CMP RO,ROUTA1
2269 013474 101771 BLOS 15
2270 013476 000002 25: EXIT
2271
2272 ;*
2273 ;*ROUTINE TO INPUT COUNTER MODULE ADDRS
2274 ;*
2275
2276 013500 005737 001652 RCNTA: TST EXPERT
2277 013504 001002 BNE .+6
2278 013506 104400 TYPE ;ASK OPERATOR FOR COUNTER MODULE
2279 013510 021076 MCNT ;ADDRS.
2280 013512 012737 013204 013534 MOV #CNTADR, RCNTA1
2281 013520 012737 000020 013536 MOV #16., RCNTA1+2
2282 013526 000401 BR .+4
2283 013530 000763 BR RCNTA
2284 013532 104411 ;GET THEM
2285 013534 013204 RCNTA1: CNTADR

```

2286	013536	000020			1b.	
2287	013540	012700	013204		MOV	#CNTADR, RO
2288	013544	005737	014754		YST	CHAR
2289	013550	001407			BEQ	ZS
2290	013552	053710	001612	1S:	BIS	ICSMOD, (0)
2291	013556	004737	015416		JSR	PC, CKADR
2292	013562	020037	013534		CMP	RO, RCNTA1

```

2293 013566 003771 BLE 15
2294
2295 013570 000002 2S: EXIT
2296
2297 ;*
2298 ;*ROUTINE TO INPUT DAC ADDR
2299 ;*
2300 013572 005737 001652 RDACA: TST EXPERT
2301 013576 001002 BNE .+6
2302 013600 104400 TYPE
2303 013602 021417 MDAC
2304 013604 012737 013244 013626 MOV #DACADR, RDAC1
2305 013612 012737 000001 013630 MOV #1, RDAC1+2
2306 013620 000401 BR .+4
2307 013622 000763 BR RDACA
2308 013624 104411
2309 013626 013244 RDAC1: INOCT DACADR
2310 013630 000001
2311 013632 005737 014754 TST CHAR
2312 013636 001407 BEQ 15
2313 013640 053737 001612 013244 BIS ICSMOD, DACADR
2314 013646 012700 013244 MOV #DACADR, R0
2315 013652 004737 015416 JSR PC, CKADR
2316 013656 000002 1S: EXIT
2317 ;*
2318 ;*ROUTINE TO INPUT A005 ADDR
2319 ;*
2320 013660 005737 001652 RADA: TST EXPERT
2321 013664 001002 BNE .+6
2322 013666 104400 TYPE
2323 013670 021442 MADU
2324 013672 012737 013246 013714 MOV #A0ADR, RADA1
2325 013700 012737 000001 013716 MOV #1, RADA1+2
2326 013706 000401 BR .+4
2327 013710 000763 BR RADA
2328 013712 104411
2329 013714 013246 RADA1: INOCT ADADR
2330 013716 000001
2331 013720 005737 014754 TST CHAR
2332 013724 001407 BEQ 15
2333 013726 053737 001612 013246 BIS ICSMOD, ADADR
2334 013734 012700 013246 MOV #ADADR, R0
2335 013740 004737 015416 JSR PC, CKADR
2336 013744 000002 1S: EXIT
2337
2338 ;*
2339 ;*INPUT OCTAL ROUTINE
2340 ;*NMW TO CALL=WHERE TO STORE NUMBERS
2341 ;*NMW = NUMBER OF NUMBERS TO ACCEPT.
2342 ;*
2343
2344 013746 104400 INOCTR: TYPE ;TYPE A "?".
2345 013750 021764 MQ
2346 013752 017601 000000 MOV 2(6), R1 ;PICK UP STORAGE ADDRESS.
2347 013756 062716 000002 ADD #2, (6) ;PICK UP # OF WORDS THAT-
2348 013762 017637 000000 014766 MOV 2(6), NINC1 ;IS THE MAX # TO BE INPUTED.
  
```

2349	013770	162716	000002		SUB	#2	(6)		;CLEAR COLON TYPED FLAG.
2350	013774	005037	001630		CLR	INCF LG			;CLEAR ANY CHAR. TYPED FLAG.
2351	014000	005037	013042		CLR	RTEMP			;POINT TO TEMP STORAGE AREA.
2352	014004	012700	013252		MOV	#OUTS,RO			;CLEAR THE AREA.
2353	014010	005020		20\$:	CLR	(0)+			
2354	014012	020027	013312		CMP	RO,#OUTSE			
2355	014016	003774			BLE	20\$			
2356	014020	012700	013252		MOV	#OUTS,RO			;POINT TO AREA AGAIN.
2357	014024	005037	014764		CLR	NINC			;CLEAR CHAR. COUNT
2358	014030	005037	014756	1\$:	CLR	NIN			;CLEAR NUMBER TO BE FORMED
2359	014034	005037	014762		CLR	CHARC			;INPUTED CHAR. COUNT
2360	014040	105777	165100	2\$:	TSTB	2\$TKS			;KEY TYPED?
2361	014044	100375			BPL	2\$;NO - THEN WAIT.
2362	014046	17737	165074	014754	MOV	2\$TKB,CHAR			;YES - READ CHAR.
2363	014054	042737	177600	014754	BIC	#177600,CHAR			;STRIP CHAR PARITY BIT - IF ANY.
2364	014062	023727	014754	000177	CMP	CHAR,#177			;WAS IT A RUBOUT?
2365	014070	001002			BNE	+.6			;NO - CONTINUE
2366	014072	000137	014714		JMP	10\$;YES - TYPE "?" - REINITIALIZE.
2367	014076	105777	165046		TSTB	2\$TPS			;PRINTER BUSY?
2368	014102	100375			BPL	-.4			;YES - THEN WAIT TILL NOT
2369	014104	013777	014754	165040	MOV	CHAR,2\$TPB			;NO - ECHO CHAR.
2370	014112	023727	014754	002015	CMP	CHAR,#15			;CHAR. = <CR>?
2371	014120	001514			BEQ	3\$;YES - THEN TERMINATE INPUT.
2372	014122	023727	014754	000003	CMP	CHAR,#3			;WAS IT A TC?
2373	014130	001002			BNE	+.6			
2374	014132	000177	165542		JMP	2\$CTLLOC			;IF YES GOTO MONITR.
2375	014136	023727	014754	000014	CMP	CHAR,#14			;CHAR = "L"?
2376	C.4144	001002			BNE	+.6			
2377	C.4146	000137	015020		JMP	LPSET			;YES - THEN SET LINEPRINTER MODE.
2378	C.4152	122737	000012	014754	CMPB	#12,CHAR			;CHAR = "J"?
2379	014160	001002			BNE	+.6			
2380	014152	000137	015042		JMP	CONNTR			;YES - THEN SET JOINED MODE.
2381	014166	122737	000004	014754	CMPB	#4,CHAR			;CHAR = "D"?
2382	014174	001002			BNE	+.6			
2383	014176	000137	015166		JMP	RDELA3			;YES - GET SECOUNDRY DELAY.
2384	014202	123727	014754	000005	CMPB	CHAR,#5			;CHAR = "E"?
2385	014210	001002			BNE	+.6			
2386	014212	000137	014770		JMP	EXSET			;YES - SET EXPERT MODE.
2387	014216	123727	014754	000016	CMPB	CHAR,#16			;CHAR = "N"?
2388	014224	001002			BNE	+.6			
2389	014226	000137	015004		JMP	NOSET			;YES - SET NOVICE MODE
2390	014232	023727	014754	000072	CMP	CHAR,#:			;CHAR = COLON?
2391	014240	001522			BEQ	4\$;YES SET FOR "THROUGH" ENTRY.
2392	014242	023727	014754	000054	CMP	CHAR,#,			;CHAR = A COMMA?
2393	014250	001575			BEQ	7\$			
2394	014252	023727	014754	000060	CMP	CHAR,#60			;CHAR TYPED 7 ASCIZ?
2395	014260	002002			BGE	+.6			
2396	014262	000137	014724		JMP	11\$;NO - REPORT ERROR.
2397									
2398	014266	023727	014754	000067	CMP	CHAR,#67			;CHAR TYPED <ASCIZ 7?
2399	014274	003402			BLE	+.6			
2400	014276	000137	014724		JMP	11\$;NO - REPORT ERROR.
2401	014302	005237	013042		INC	RTEMP			;YES - INCREMENT CHAR. COUNT.
2402	014306	006137	014756		ROL	NIN			;LEFT JUSTIFY CURRENT NUMBER.
2403	014312	006137	014756		ROL	NIN			
2404	014316	006137	014756		ROL	NIN			

2405	014322	005237	014762		INC	CHARC			; INCREMENT CHAR COUNT.
2406	014326	042737	000007	014756	BIC	#7	NIN		; STRIP LOWER NUMBER.
2407	014334	042737	000260	014754	BIC	#260,	CHAR		; STRIP CHAR. INPUTED.
2408	014342	053737	014754	014756	BIS	CHAR,	NIN		; ADD TO CURRENT NUMBER.
2409	014350	000633			BR	2\$; LOOP.
2410	014352	005737	001630		3\$: TST	INCFLG			; COME HERE ON <CR>: COLON FLAG SET?
2411	014356	001066			BNE	5\$; YES TAKE CARE OF IT.
2412	014360	023737	014764	014766	CMP	NINC,	NINC1		; DID WE ASSEMBLE MORE NUMBERS THAN ALLOWED?
2413	014366	003146			BGT	8\$; IF SO - REPORT ERROR.
2414	014370	005737	013042		TST	RTEMP			; ANY CHAR. TYPED?
2415	014374	001005			BNE	9\$; YES - FUDGE RETURN.
2416	014376	012737	000017	014764	MOV	#15,	NINC		
2417	014404	005037	014754		CLR	CHAR			
2418	014410	013720	014756		9\$: MOV	NIN,	(0)+		; STORE NUMBER.
2419	014414	013737	014766	013042	MOV	NINC1,	RTEMP		; FIX NUMBER INPUTED COUNT.
2420	014422	163737	014764	014766	SUB	NINC,	NINC1		; FIX EXACT COUNT.
2421	014430	005337	014766		DEC	NINC1			; = COUNT - 1
2422	014434	012700	013252		MOV	#OUTS,RO			; SET TO READ BACK NUMBERS.
2423	014440	012021			21\$: MOV	(0)+,(1)+			; STORE IN CORRECT AREA.
2424	014442	005337	014764		DEC	NINC			; DONE ALL NUMBERS?
2425	014446	001002			BNE	22\$; NO - SKIP NEXT INSTR.
2426	014450	010176	000000		MOV	R1,2(6)			; FIX CALL ON ADDR. OF NUMBERS TO INPUT.
2427	014454	005337	013042		22\$: DEC	RTEMP			; DONE ALL NUMBERS.
2428	014460	001367			BNE	21\$; NO - LOOP.
2429	014462	062716	000002		ADD	#2,(6)			; YES - UPDATE SP.
2430	014466	013776	014766	000000	MOV	NINC1,2(6)			; STORE ON CALL # OF NUMBERS TO INPUT.
2431	014474	062716	000002		ADD	#2,(6)			; FIX SP.
2432	014500	104400	001163		TYPE,	\$CR LF			
2433	014504	000002			EXIT				; EXIT.
2434									
2435	014506	013737	014756	014760	4\$: MOV	NIN,	NIN1		; ENTER HERE WHEN COLON TYPED. STORE CURRENT #.
2436	014514	005737	001630		TST	INCFLG			; CHECK COLON FLAG.
2437	014520	001105			BNE	12\$; IF SET UNKNOWN INPUT (2 COLONS!)
2438	014522	012737	000001	001630	MOV	#1,INCFLG			; SET COLON FLAG.
2439	014530	000137	014030		JMP	1\$; EXIT TO LOOP
2440	014534	162737	000002	014760	5\$: SUB	#2,	NIN1		; COME HERE ON COLON FLAG SET + ANOTHER # FORMED.
2441	014542	062737	000002	014760	6\$: ADD	#2,	NIN1		; UPDATE FORMER NUMBER.
2442	014550	023737	014764	014766	CMP	NINC,	NINC1		; NUMBERS OVERFLOW BUFFER AREA?
2443	014556	003052			BGT	8\$; YES - REPORT ERROR.
2444	014560	013720	014760		MOV	NIN1,	(0)+		; NO - STORE NEW NUMBER.
2445	014564	005237	014764		INC	NINC			; UPDATE NUMBER COUNT.
2446	014570	023737	014756	014760	CMP	NIN,	NIN1		; UPDATED NUMBER = CURRENT NUMBER?
2447	014576	001361			BNE	6\$; NO - LOOP.
2448	014600	005037	001630		CLR	INCFLG			; YES - CLEAR COLON FLAG
2449	014604	005337	014764		DEC	NINC			
2450	014610	162700	000002		SUB	#2,RO			; FIX POINTER (STORAGE).
2451	014614	022737	000015	014754	CMP	#15,	CHAR		; LAST CHAR TYPED A <CR>?
2452	014622	001653			BEQ	3\$; YES - EXIT.
2453	014624	022737	000054	014754	CMP	#',,CHAR			; LAST CHAR TYPED A COMMA?
2454	014632	001404			BEQ	7\$; YES - TAKE CARE OF IT.
2455									
2456	014634	062700	000002		ADD	#2,	RO		; FIX STORAGE POINTER.
2457	014640	000137	014030		JMP	1\$; LOOP.
2458	014644	005737	001630		7\$: TST	INCFLG			; ENTER HERE ON COMMA TYPED. WAS COLON-
2459	014650	001331			BNE	5\$; PREVIOUSLY TYPED? IF SO TAKE CARE OF IT.
2460	014652	023737	014764	014766	CMP	NINC,	NINC1		; BUFFER OVERFLOW?


```

2517
2518
2519
2520 015042 005137 001654 CONNTR: COM CONNT
2521 015046 001402 BEQ .+6
2522 015050 104400 TYPE
2523 015052 021305 MAC
2524 015054 000137 014744 JMP RASK
2525
2526
2527
2528
2529
2530
2531 015060 005737 001652 RDELA: TST EXPERT
2532 015064 001002 BNE .+6
2533 015066 104400 TYPE ;ASK OPERATOR -
2534 015070 021632 MDEL ;"DELAY TIME IN MS (DEFAULT=3MS)?"
2535 015072 005037 001570 CLR FREQ1
2536 015076 013737 001564 001566 MOV FR3, FREQ
2537 015104 104400 TYPE
2538 015106 021764 MQ
2539 015110 104406 RDDEC ;GET NUMBER.
2540 015112 012637 013042 MOV (6)+, RTEMP
2541 015116 001417 BEQ 2$ ;IF ZERRO, USE DEFAULT
2542 015120 005037 001570 CLR FREQ1
2543 015124 005037 001566 CLR FREQ
2544 015130 063737 001576 001566 1$: ADD FR1, FREQ ;SET FREQUENCY (OR DELAY).
2545 015136 005537 001570 ADC FREQ1
2546 015142 005737 001570 TST FREQ1
2547 015146 100404 BMI 3$
2548 015150 005337 013042 DEC RTEMP
2549 015154 001365 BNE 1$
2550 015156 000002 2$: EXIT
2551
2552 015160 104400 3$: TYPE ;TYPE" TIME TOO LONG".
2553 015162 021672 M TTL
2554 015164 000735 BR RDELA
2555
2556
2557
2558
2559 015166 104400 021347 RDELA3: TYPE, MDEL1
2560 015172 104400 RDELA4: TYPE
2561 015174 021352 MDEL2
2562 015176 104406 RDDEC
2563 015200 012637 013042 MOV (6)+, RTEMP
2564 015204 001415 BEQ 2$
2565 015206 005037 001572 CLR FREQ2
2566 015212 063737 001576 001566 1$: ADD FR1, FREQ
2567 015220 005537 001574 ADC FREQ3
2568 015224 005737 001574 TST FREQ3
2569 015230 100405 BMI 3$
2570 015232 005337 013042 DEC RTEMP
2571 015236 001365 BNE 1$
2572 015240 000137 014744 2$: JMP RASK

```

```

2573 015244 104400      3$:  TYPE
2574 015246 021672      MTTL
2575 015250 000750      BR      RDELAY
2576
2577
2579
2579
2580
2581
2582
2583 015252 005737 001652      RQUBR: TST      EXPERT
2584 015256 001002      BNE      .+6
2585 015250 104400      TYPE      ;UNI OR BI-POLAR?
2586 015262 024276      MQURB
2587 015264 104400      TYPE
2588 015266 021764      MQ
2589 015270 005037 014754      CLR      CHAR
2590 015274 105777 163644      1$:  TSTB     @STKS      ;WAIT FOR RESPONSE
2591 015300 100375      BPL
2592 015302 105777 163642      2$:  TSTB     @STPS      ;ECHO.
2593 015306 100375      BPL
2594 015310 017737 163632 013042      MOV      @STKB,RTEMP
2595 015316 013777 013042 163626      MOV      RTEMP,@STPB
2596 015324 042737 000240 013042      BIC      #240,RTEMP
2597 015332 023727 013042 000003      CMP      RTEMP,#3      ;↑C TYPED?
2598 015340 001002      BNE      .+6
2599 015342 000137 003366      JMP      START1
2600 015346 023727 013042 000015      CMP      RTEMP,#15     ;CR TYPED?
2601 015354 001404      BEQ      3$
2602 015356 013737 013042 014754      MOV      RTEMP,CHAR
2603 015364 000743      BR
2604 015366 104400 001163      3$:  TYPE,    SCALF
2605 015372 023727 014754 000125      CMP      CHAR,#'U
2606 015400 001003      BNE      4$
2607 015402 005237 013250      INC      BIPOL
2608 015406 000002      EXIT
2609 015410 005037 013250      4$:  CLR      BIPOL
2610 015414 000002      EXIT
2611
2612
2613
2614
2615
2616
2617
2618 015416 013737 000004 015516      CKADR: MOV      @#4,3$      ;STORE LOCATION 4.
2619 015424 012737 015466 000004      MOV      #2$,@#4      ;SET TIME-OUT LOCATION FOR TRAP IF ANY.
2620 015432 005770 000000      TST      @#4          ;TEST MOD. ADDR. TYPED.
2621 015436 000240      NOP
2622
2623
2624
2625
2626
2627 015440 013737 015516 000004      MOV      3$,@#4      ;RESTORE TIME OUT VECTOR
2628 015446 023710 001702      CMP      IC$LMT,(RQ)

```

;*
;*ROUTINE TO SET UNI OR BIPOLAR A/D
;*CALL = QUBR
;*

;UNI OR BI-POLAR?

;WAIT FOR RESPONSE

;ECHO.

;↑C TYPED?

;CR TYPED?

;*
;*ROUTINE TO CHECK THAT AN ADDRESS TYPED IS A LEGAL ICS-ADDR.
;*TYPE ERROR IF NOT - ENTER WITH ADDR IN (0)
;*CALL = JSR PC,CKADR
;*

;STORE LOCATION 4.
;SET TIME-OUT LOCATION FOR TRAP IF ANY.
;TEST MOD. ADDR. TYPED.

*** MODULE ADDRESS DOES EXIST BUT MAY NOT BE ASSOCIATED WITH THIS
*** FILE BOX. THEREFORE, CHECK ADDRESS FOR WITHIN FILE BOX LIMITS

;RESTORE TIME OUT VECTOR

```

2629 015452 003022          BGT      4$
2630 015454 023710 001676   CMP      ICSHGH, (R0)
2631 015460 003417          BLE      4$
2632 015462 005720          1$:     TST      (0)+
2633 015464 000207          RTS      PC
2634 015466 104400 022115   2$:     TYPE,   MNRFN
2635 015472          TYPOCT  (0)
2636 015472 011046          MOV      (0), -(SP)
2637 015474 104401          TYPOC
2638 015476 062706 000006   ADD      #6, R6
2639 015502 162716 000002   SUB      #2, (6)
2640 015506 013737 015516 000004  MOV      3$, 2#4
2641 015514 000002          EXIT
2642
2643 015516 000000          3$:     .WORD  0
2644
2645
2646 015520 104400 026361          4$:     TYPE,   NRANG1
2647 015524          TYPOCT  (0)
2648 015524 011046          MOV      (0), -(SP)
2649 015526 104401          TYPOC
2650 015530 104400 026374   TYPE,   NRANG2
2651 015534 005726          TST      (R6)+
2652 015536 162716 000002   SUB      #2, (R6)
2653 015542 000002          EXIT
2654
2655
2656
2657
2658
2659
2660
2661 015544 013737 001566 013042  RDELAY: MOV      FREQ,  RTEMP ;GET DELAY TIME
2662 015552 001411          BEQ      2$
2663 015554 013737 001570 015600   MOV      FREQ1, RTEMP2
2664 015562 005337 013042          1$:     DEC      RTEMP ;DELAY
2665 015566 001375          BNE
2666 015570 005337 015600   DEC      RTEMP2
2667 015574 100372          BPL      1$
2668 015576 000002          2$:     EXIT ;RETURN
2669 015600 000000          RTEMP2: 0
2670
2671
2672
2673
2674
2675 015602 013737 001572 013042  RDELA2: MOV      FREQ2, RTEMP
2676 015610 001413          BEQ      2$
2677 015612 013737 001574 015600   MOV      FREQ3, RTEMP2
2678 015620 005037 177776          CLR      PS
2679 015624 005337 013042          1$:     DEC      RTEMP
2680 015630 001375          BNE
2681 015632 005337 015600   DEC      RTEMP2
2682 015636 100372          BPL      1$
2683 015640 000002          2$:     EXIT
2684

```

```

;EXIT - SUBROUTINE.
;TYPE ERROR MESSAGE.
;TYPE ADDR.
;SAVE (0) FOR TYPEOUT
;GO TYPE--OCTAL ASCII(ALL DIGITS)
;FIX RETURN ADDRESS ON STACK
;RESTORE VECTOR ADDRESS
;SO THAT WE RETURN AND REASK
;QUESTION ON MODULE ADDR.
;TEMP STORAGE OF LOC. 4.
;NOT WITHIN FILE
;;SAVE (0) FOR TYPEOUT
;;GO TYPE--OCTAL ASCII(ALL DIGITS)
;POP STACK PAST JSR RETURN
;BACK POINTER TO RE-ASK QUESTION
;RETURN

```

```

;*
;*ROUTINE USED OFR GENERATION A DELAY
;*CALL=DELAT
;*

```

```

;*
;*ROUTINE TO GENERATE SECOUNDARY DELAY
;*CALL =DELAY2
;*

```

```

2685
2686 ;*
2687
2688
2689 ;* SYSTEM INITIALIZE ROUTINE
2690 ;*
2691
2692 RINIT:
2693 015642 013700 001104 MOV SICNT,RO
2694 015646 000005 RESET
2695 015650 032777 000001 163262 BIT #1,@SWR
2696 015656 001402 BEQ 1$
2697 015660 000137 003366 JMP START1
2698
2699 015664 052777 000100 163252 1$: BIS #100, @STKS
2700 015672 000002 EXIT
2701
2702
2703 ;*
2704 ;*ROUTINE USED TO MODIFY PATTERN SENT TO OUTPUT MODULE
2705 ;*FIRST LOCATION MAY BE CHANGED BY RPATA (PATTERN MODIFIER INPUT ROUTINE)
2706 ;*CALL=CPATR
2707 ;*
2708
2709 015674 023727 001602 000012 RCPAT: CMP PATRNM,#12
2710 015702 001442 BEQ RCPATR
2711 015704 000241 CLC ;CLEAR C BIT
2712 015706 005237 001604 RCPATI: INC @PATRN ;MODIFY PATTERN
2713 015712 103412 BCS 1$
2714 015714 022737 005137 015706 CMP #5137,RCPATI ;DOING A COMPLEMENT PATERN?
2715 015722 001406 BEQ 1$ ;IF SO DON'T ADD CARRY IF ANY!
2716 015724 063737 001606 001604 ADD PATRNC,PATRN
2717 015732 005037 001606 CLR PATRNC
2718 015736 000002 EXIT
2719 015740 023727 015706 006037 1$: CMP RCPATI,#6037
2720 015746 001004 BNE 2$
2721 015750 012737 100000 001606 MOV #100000,PATRNC
2722 015756 000002 EXIT
2723 015760 012737 000001 001606 2$: MOV #1,PATRNC
2724 015766 000002 EXIT
2725 ;*MODIFIER LIST
2726 015770 005237 RCPATL: 5237 ;*INCREMENTING PATTERN
2727 015772 005337 5337 ;*DECREMENTING PATTERN
2728 015774 005737 5737 ;*NO CHANGE OF PATTERN
2729 015776 006137 6137 ;*ROTATE LEFT PATTERN
2730 016000 006037 6037 ;*ROTATE RIGHT PATTERN
2731 016002 006237 6237 ;*RANDOM NUMBER GENERATOR
2732 016004 006337 6337 ;*ARITH. SHIFT LEFT PATTERN
2733 016006 005137 5137 ;*COMPLEMENT PATTERN
2734
2735
2736 ;*
2737 ;*ALTERNATE CHANGE PATTERN ROUTINE IF RANOME
2738 ;*NUMBER FOR PATTERN IS SELECTED.
2739 ;*
2740 016010 RCPATR:

```

0.
1
2
3
4
5
6
7

```

2741 016010 004737 016244 JSR PC,$RAND ;GENERATE A RANDOM NUMBER.
2742 016014 013737 016344 001604 MOV $LONUM,PATRN ;PUT NUMBER IN PATTERN.
2743 016022 000002 EXIT
2744
2745
2746 ;*
2747 ;*PATTERN INPUTTER AND MODIFIER ROUTINE
2748 ;*FORM PATTERN MODIFIER, PATTERN.
2749 ;*CALL=PATAR
2750 ;*
2751 016024 005737 001652 RPATA: TST EXPERT
2752 016030 001002 BNE .+6
2753 016032 104400 TYPE ;ASK OPERATOR-"PATTERN MODIFIER AND
2754 ;PATTERN?"
2755 016034 021714 MPPM ;SET TO INPUT PATTERN + MODIFIER
2756 016036 012737 001602 016070 MOV #FATRN, 1$
2757 016044 012737 000002 016072 MOV #2, 1$+2
2758 016052 005037 001602 CLR PATRN ;SET DEFAULTS: 0 PATTERN
2759 016056 005037 001604 CLR PATRN ;PATTERN MOD. = INC.
2760 016062 000401 BR .+4
2761 016064 000757 BR RPATA
2762 016066 104411 INOCT ;GET THEM.
2763 016070 001602 1$: PATRN
2764 016072 000002 2
2765 016074 012737 015770 013042 MOV #RCPATL,RTEMP ;POINT TO BEGINNING OF MODIFIER LIST.
2766 016102 032737 177770 001602 BIT #177770,PATRN ;LEGAL PATTERN MODIFIER?
2767 016110 001403 BEQ 2$ ;IF YES 2$
2768 016112 104400 025326 TYPE, MIVP ;NO TYPE ERROR MESSAGE.
2769 016116 000742 BR RPATA ;REASK QUESTION.
2770 016120 2$:
2771 016120 006337 001602 ASL PATRN
2772 016124 042737 177761 001602 BIC #177761, PATRN ;FIX ADDITIVE.
2773 016132 063737 001602 013042 ADD PATRN, RTEMP ;POINT TO MODIFIER.
2774 016140 017737 174676 015706 MOV @RTEMP, RCPAT+12 ;CHANGE PATTERN MODIFIER ROUTINE.
2775 016146 000002 EXIT ;RETURN.
2776 016150 .SSAVE
2777 .SBTTL SAVE AND RESTORE RO-R5 ROUTINES
2778
2779 016150 STARS
2780 ;*****
2781 ;SAVE RO-R5
2782 ;CALL:
2783 ; SAVREG
2784 ;UPON RETURN FROM $$AVREG THE STACK WILL LOOK LIKE:
2785 ;*
2786 ;*TOP---(+16)
2787 ;* +2---(+18)
2788 ;* +4---R5
2789 ;* +6---R4
2790 ;* +8---R3
2791 ;*+10---R2
2792 ;*+12---R1
2793 ;*+14---R0
2794
2795 016150 $$AVREG:
2796 016150 PUSH <R0,R1,R2,R3,R4,R5>

```

```

016150 010046
016152 010146
016154 010246
016156 010346
016160 010446
016162 010546
016164 016646 000022
016166 016646 000022
016168 016646 000022
016170 016646 000022
016172 016646 000022
016174 000002

```

```

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 22(SP),-(SP)  :: SAVE PS OF MAIN FLOW
MOV 22(SP),-(SP)  :: SAVE PC OF MAIN FLOW
MOV 22(SP),-(SP)  :: SAVE PS OF CALL
MOV 22(SP),-(SP)  :: SAVE PC OF CALL
RTI

```

*RESTORE RO-R5

*CALL:

* RESREG

\$RESREG:

```

016206
016206 012666 000022
016212 012666 000022
016216 012666 000022
016222 012666 000022
016226
016226 012605
016230 012604
016232 012603
016234 012602
016236 012601
016240 012500
016242 000002
016244

```

```

MOV (SP)+,22(SP)  :: RESTORE PC OF CALL
MOV (SP)+,22(SP)  :: RESTORE PS OF CALL
MOV (SP)+,22(SP)  :: RESTORE PC OF MAIN FLOW
MOV (SP)+,22(SP)  :: RESTORE PS OF MAIN FLOW
POP R5,R4,R3,R2,R1,R0
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
RTI

```

\$RAND .SBTTL RANDOM NUMBER GENERATOR ROUTINE

STARS

```

*****
*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
*WITH A RANGE OF 0 TO 2(+33)-1.

```

*CALL:

```

* JSR PC,$RAND      :: CALL THE ROUTINE
* RETURN            :: RETURN HERE THE RANDOM
*                  :: NUMBER WILL BE IN
*                  :: $HINUM,$LONUM

```

```

016244
016244 010046
016246 010146
016250 010246
016252 013700 016344
016256 013701 016342
016262 012702 177771
016266 006300
016270 006101
016272 005202
016274 001374
016276 063700 016344
016302 005501
016304 063701 016342
016310 062700 001057

```

```

$RAND: PUSH (R0,R1,R2)
MOV RO,-(SP)      :: PUSH RO ON STACK
MOV R1,-(SP)      :: PUSH R1 ON STACK
MOV R2,-(SP)      :: PUSH R2 ON STACK
MOV $LONUM,R0     :: SET RO WITH LOW
MOV $HINUM,R1     :: SET R1 WITH HIGH
MOV #-7,R2        :: SET SHIFT COUNT
1$: ASL R0         :: SHIFT RO LEFT AND
ROL R1            :: ROTATE CARRY INTO R1 AND
INC R2            :: CHECK FOR DONE
BNE 1$           :: CONTINUE SHIFT LOOP
ADD $LONUM,R0     :: ADD NUMBER TO MAKE X 129
ADC R1            :: PROPOGATE CARRY
ADD $HINUM,R1     :: ADD NUMBER TO MAKE X 129
ADD #1057,R0      :: ADD LOW CONSTANT

```

```

2853 016314 005501          ADC      R1          ;; PROPGATE CARRY
2854 016316 062701 047401  ADD      #47401,R1   ;; ADD HIGH CONSTANT
2855 016322 010037 016344  MOV      R0,$LONUM  ;; SAVE R0
2856 016326 010137 016342  MOV      R1,$HINUM  ;; SAVE R1
2857 016332          POP      R2,R1,R0
2858 016332 012602          MOV      (SP)+,R2   ;; POP STACK INTO R2
2859 016334 012601          MOV      (SP)+,R1   ;; POP STACK INTO R1
2860 016336 012600          MOV      (SP)+,R0   ;; POP STACK INTO R0
2861 016340 000207          RTS      PC          ;; RETURN
2862 016342 176543          $HINUM: .WORD 176543
2863 016344 123456          $LONUM: .WORD 123456
2864 016346          .SSB2D
2865          .SBTTL SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
2866
2867 016346          STARS
2868          ;;*****
2869          ;;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
2870          ;;*UNSIGNED DECIMAL ASCIZ NUMBER.
2871          ;;*CALL
2872          ;;*   MOV      NUMBER, -(SP)  ;; PUT BINARY NUMBER ON THE STACK
2873          ;;*   JSR      PC, @#$SB2D   ;; CALL
2874          ;;*   RETURN                ;; ADDRESS OF THE 1ST ASCIZ CHAR. IS ON THE STACK
2875
2876
2877 016346 016637 000002 016376 $SB2D:  MOV      2(SP), 1$   ;; SAVE BINARY NUMBER
2878 016354 012746 016376          MOV      #1$, -(SP)  ;; SET POINTER
2879 016360 004737 016402          JSR      PC, @#$DB2D ;; CALL DOUBLE LENGTH CONVERT
2880 016364 062716 000005          ADD      #5, (SP)   ;; ONLY ALLOW FIVE CHARACTERS
2881 016370 012666 000002          MOV      (SP)+, 2(SP) ;; PICKUP POINTER
2882 016374 000207          RTS      PC          ;; RETURN
2883 016376 000000 000000          1$:      .WORD 0,0
2884 016402          .SDB2D
2885          .SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
2886
2887 016402          STARS
2888          ;;*****
2889          ;;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
2890          ;;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
2891          ;;*POSITIVE.
2892          ;;*CALL
2893          ;;*   MOV      #PNTR, -(SP)  ;; POINTER TO LOW WORD OF BINARY NUMBER
2894          ;;*   JSR      PC, @#$DB2D   ;;
2895          ;;*   RETURN                ;; THE FIRST ADDRESS OF ASCII
2896          ;;*                          ;; IS ON THE STACK
2897
2898
2899 016402 104407          $DB2D:  SAVREG          ;; SAVE REGISTERS
2900 016404 016602 000002          MOV      2(SP), R2   ;; PICKUP THE DATA POINTER
2901 016410 012700 016562          MOV      #$DECVL, R0 ;; GET ADDRESS OF "$DECVL" STRING
2902 016414 010066 000002          MOV      R0, 2(SP)  ;; PUT ADDRESS OF ASCII STRING ON STACK
2903 016420 012201          MOV      (R2)+, R1   ;; PICKUP THE BINARY NUMBER
2904 016422 012202          MOV      (R2)+, R2
2905 016424 012737 000012 016500          MOV      #10, 4$    ;; SET UP TO DO 10 CONVERSIONS
2906 016432 012704 016512          MOV      $TNPWR, R4  ;; ADDRESS OF TEN POWER
2907 016436 012705 016514          MOV      $TNPWR+2, R5
2908 016442 005003          1$:      CLR      R3          ;; CLEAR PARTIAL

```

```

2909 016444 161401      2S:   SUB      (R4),R1      ;;SUBTRACT TEN POWER
2910 016446 005602      SBC      R2
2911 016450 161502      SUB      (R5),R2
2912 016452 002402      BLT      3S          ;;BR IF TEN POWER TO LARGE . .
2913 016454 005203      INC      R3          ;;ADD 1 TO PARTIAL
2914 016456 000772      BR       2S          ;;LOOP
2915 016460 062401      3S:   ADD      (R4)+,R1      ;;RESTORE SUBTRACTED VALUE
2916 016462 005502      ADC      R2
2917 016464 062402      ADD      (R4)+,R2
2918 016466 022525      CMP      (R5)+,(R5)+  ;;MOVE TO NEXT TEN POWER
2919 016470 052703      BIS      #'0,R3      ;;CHANGE PARTIAL TO ASCII
2920 016474 110320      MOVW    R3,(R0)+     ;;SAVE IT
2921 016476 005327      DEC      (PC)+       ;;DONE?
2922 016500 000000      4S:   .WORD    0
2923 016502 001357      BNE     1S          ;;BR IF NO
2924 016504 105020      CLRB    (R0)+       ;;TERMINATOR
2925 016506 104410      RESREG          ;;RESTORE REGISTERS
2926 016510 000207      RTS     PC          ;;RETURN
2927 016512 145000      $TNPWR: 145000      ;;1.0E09
2928 016514 035632      35632
2929 016516 160400      160400      ;;1.0E08
2930 016520 002765      2765
2931 016522 113200      113200      ;;1.0E07
2932 016524 000230      230
2933 016526 041100      041100      ;;1.0E06
2934 016530 000017      17
2935 016532 103240      103240      ;;1.0E05
2936 016534 000001      1
2937 016536 023420      23420      ;;1.0E04
2938 016540 000000      0
2939 016542 001750      1750      ;;1.0E03
2940 016544 000000      0
2941 016546 000144      144      ;;1.0E02
2942 016550 000000      0
2943 016552 000012      12      ;;1.0E01
2944 016554 000000      0
2945 016556 000001      1      ;;1.0E00
2946 016560 000000      0
2947 016562 000014      $DECVL: .BLKB 12.      ;;RESERVE STORAGE FOR ASCII STRING

```

000060

```

2953 016576      .SERROR SERRTYP
2954      .SBTTL  ERROR HANDLER ROUTINE
2955
2956 016576      STARS
2957      ;;*****
2958      ;;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2959      ;;SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2960      ;;AND GO TO SERRTYP ON ERROR
2961      ;;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2962      ;;SW15=1      HALT ON ERROR
2963      ;;SW13=1      INHIBIT ERROR TYPEOUTS
2964      ;;CALL

```

2965
2966
2967 016576
2968 016576 105237 001103
2969 016576 001775
2970 016604 013777 001102 162330
2971 016612 005237 001112
2972 016616 011637 001116
2973 016622 162737 000002 001116
2974 016630 117737 162262 001114
2975 016636 032777 020000 162274
2976 016644 001004
2977 016646 004737 016670
2978 016652 104400 001163
2979 016656
2980 016656 005777 162256
2981 016662 100001
2982 016664 000000
2983 016666
2984 016666 000002
2985 016670
2986
2987
2988 016670
2989
2990
2991
2992
2993
2994 016670
2995 016670 104400 001163
2996 016674 010046
2997 016676 005000
2998 016700 153700 001114
2999 016704 001004
3000
3001 016706
3002 016706 013746 001116
3003
3004 016712 104401
3005 016714 000426
3006 016716 005300
3007 016720 006300
3008 016722 006300
3009 016724 006300
3010 016726 062700 001166
3011 016732 012037 016742
3012 016736 001404
3013 016740 104400
3014 016742 000000
3015 016744 104400 001163
3016 016750 012037 016760
3017 016754 001404
3018 016756 104400
3019 016760 000000
3020 016762 104400 001163

```
;* ERPR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER
$ERROR:
7$: INCB $ERFLG ;;SET THE ERROR FLAG
    BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
    MOV $STNM,$DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
    INC $ERTTL ;;INC THE ERROR COUNT
    MOV (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
    SUB #2,$ERRPC
    MOVB $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
    BIT #BIT13,$SWR ;;SKIP TYPEOUT IF SET
    BNE 20$ ;;SKIP TYPEOUTS
    JSR PC,$ERRTYP ;;GO TO USER ERROR ROUTINE
    TYPE ,SCLF
20$:
2$: TST $SWR ;;HALT ON ERROR
    BPL 3$ ;;SKIP IF CONTINUE
    HALT ;;HALT ON ERROR!
3$:
    RTI ;;RETURN
.SBTTL $ERRTYP
      ERROR MESSAGE TYPEOUT ROUTINE
STARS
;*****
;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
;ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
;AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
$ERRTYP:
    TYPE $SCLF ;;"CARRIAGE RETURN" & "LINE FEED"
    MOV RO,-(SP) ;;SAVE RO
    CLR RO ;;PICKUP THE ITEM INDEX
    BISB $ITEMB,RO
    BNE 1$ ;;IF ITEM NUMBER IS ZERO, JUST
    ;;TYPE THE PC OF THE ERROR
    TYPOCT $ERRPC,(ERROR ADDRESS)
    MOV $ERRPC,-(SP) ;;SAVE $ERRPC FOR TYPEOUT
    ;;ERROR ADDRESS
    TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    BR 6$ ;;GET OUT
1$: DEC RO ;;ADJUST THE INDEX SO THAT IT WILL
    ASL RO ;; WORK FOR THE ERROR TABLE
    ASL RO
    ASL RO
    ADD # $ERRTB,RO ;;FORM TABLE POINTER
    MOV (RO)+,$ ;;PICKUP "ERROR MESSAGE" POINTER
    BEQ 3$ ;;SKIP TYPEOUT IF NO POINTER
    TYPE ;;TYPE THE "ERROR MESSAGE"
    .WORD 0 ;;"ERROR MESSAGE" POINTER GOES HERE
2$: TYPE $SCLF ;;"CARRIAGE RETURN" & "LINE FEED"
3$: MOV (RO)+,$4$ ;;PICKUP "DATA HEADER" POINTER
    BEQ 5$ ;;SKIP TYPEOUT IF 0
    TYPE ;;TYPE THE "DATA HEADER"
4$: .WORD 0 ;;"DATA HEADER" POINTER GOES HERE
    TYPE ,SCLF ;;"CARRIAGE RETURN" & "LINE FEED"
```

```

3021 016766 011000          5$:  MOV      (RO),RO          ;; PICKUP "DATA TABLE" POINTER
3022 016770 001004          BNE      7$                    ;; GO TYPE THE DATA
3023 016772 012600          6$:  MOV      (SP)+,RO          ;; RESTORE RO
3024 016774 104400 001163  TYPE      $CRLF              ;; "CARRIAGE RETURN" & "LINE FEED"
3025 017000 000207          RTS      PC                    ;; RETURN
3026 017002          7$:  TYPOCT  2(RO)+          ;; TYPE AN OCTAL NUMBER
3027 017002 013046          MOV      2(RO)+,-(SP)        ;; SAVE 2(RO)+ FOR TYPEOUT
3028 017004 104401          TYPOC                    ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
3029 017006 005710          TST      (RO)                ;; IS THERE ANOTHER NUMBER?
3030 017010 001770          BEQ      6$                    ;; BR IF NO
3031 017012 104400 017020  TYPE      8$                    ;; TYPE TWO(2) SPACES
3032 017016 000771          BR       7$                    ;; LOOP
3033 017020 020040 000      8$:  .ASCIZ  / /                    ;; TWO(2) SPACES
3034 017024 017024          .EVEN
3035 017024          .$TYPOCT
3036          .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
3037
3038 017024          STARS
3039          ;;*****
3040          ;;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3041          ;;OCTAL (ASCII) NUMBER AND TYPE IT.
3042          ;;$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3043          ;;CALL:
3044          ;;      MOV      NUM,-(SP)          ;; NUMBER TO BE TYPED
3045          ;;      TYPOS                    ;; CALL FOR TYPEOUT
3046          ;;      .BYTE  N                    ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3047          ;;      .BYTE  M                    ;; M=1 OR 0
3048          ;;                                  ;; 1=TYPE LEADING ZEROS
3049          ;;                                  ;; 0=SUPPRESS LEADING ZEROS
3050
3051          ;;$STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3052          ;;$TYPOS OR $TYPOC
3053          ;;CALL:
3054          ;;      MOV      NUM,-(SP)          ;; NUMBER TO BE TYPED
3055          ;;      TYPON                    ;; CALL FOR TYPEOUT
3056
3057          ;;$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3058          ;;CALL:
3059          ;;      MOV      NUM,-(SP)          ;; NUMBER TO BE TYPED
3060          ;;      TYPOC                    ;; CALL FOR TYPEOUT
3061
3062 017024 017646 000000 017247  $TYPOS: MOV      2(SP),-(SP)          ;; PICKUP THE MODE
3063 017030 116637 000001          MOV      1(SP), $OFILL        ;; LOAD ZERO FILL SWITCH
3064 017036 112637 017251          MOV      (SP)+, $OMODE+1    ;; NUMBER OF DIGITS TO TYPE
3065 017042 062716 000002          ADD      #2, (SP)          ;; ADJUST RETURN ADDRESS
3066 017046 000406          BR       $TYPON
3067 017050 112737 000001 017247  $TYPOC: MOV      #1, $OFILL        ;; SET THE ZERO FILL SWITCH
3068 017056 112737 000006 017251          MOV      #6, $OMODE+1      ;; SET FOR SIX(6) DIGITS
3069 017064 112737 000005 017246  $TYPON: MOV      #5, $OCNT        ;; SET THE ITERATION COUNT
3070 017072 010346          MOV      R3,-(SP)          ;; SAVE R3
3071 017074 010446          MOV      R4,-(SP)          ;; SAVE R4
3072 017076 010546          MOV      R5,-(SP)          ;; SAVE R5
3073 017100 113704 017251          MOV      $OMODE+1,R4      ;; GET THE NUMBER OF DIGITS TO TYPE
3074 017104 005404          NEG      R4
3075 017106 062704 000006          ADD      #6,R4            ;; SUBTRACT IT FOR MAX. ALLOWED
3076 017112 110437 017250          MOV      R4,$OMODE        ;; SAVE IT FOR USE
    
```

```

3077 017116 113704 017247          MOV      $OFILL,R4      ;; GET THE ZERO FILL SWITCH
3078 017122 016605 000012          MOV      12(SP),R5     ;; PICKUP THE INPUT NUMBER
3079 017126 005003                   CLR      R3            ;; CLEAR THE OUTPUT WORD
3090 017130 006105                   1S:     ROL      R5     ;; ROTATE MSB INTO "C"
3081 017132 000404                   BR       3S            ;; GO DO MSB
3082 017134 006105                   2S:     ROL      R5     ;; FORM THIS DIGIT
3083 017136 006105                   ROL      R5
3084 017140 006105                   ROL      R5
3085 017142 010503                   MOV      R5,R3
3096 017144 006103                   3S:     ROL      R3     ;; GET LSB OF THIS DIGIT
3087 017146 105337 017250          DECB     $OMODE        ;; TYPE THIS DIGIT?
3088 017152 100016                   BPL     7S            ;; BR IF NO
3099 017154 042703 177770          BIC     #177770,R3    ;; GET RID OF JUNK
3090 017160 001002                   BNE     4S            ;; TEST FOR 0
3091 017162 005704                   TST     R4            ;; SUPPRESS THIS 0?
3092 017164 001403                   BEQ     5S            ;; BR IF YES
3093 017166 005204                   4S:     INC      R4     ;; DON'T SUPPRESS ANYMORE 0'S
3094 017170 052703 000060          BIS     #'0,R3       ;; MAKE THIS DIGIT ASCII
3095 017174 052703 000040          5S:     BIS     #' ',R3  ;; MAKE ASCII IF NOT ALREADY
3096 017200 110337 017244          MOV      R3,$S        ;; SAVE FOR TYPING
3097 017204 104400 017244          TYPE    $S           ;; GO TYPE THIS DIGIT
3098 017210 105337 017246          7S:     DECB     $OCNT  ;; COUNT BY 1
3099 017214 003347                   BGT     2S           ;; BR IF MORE TO DO
3100 017216 002402                   BLT     6S           ;; BR IF DONE
3101 017220 005204                   INC     R4           ;; INSURE LAST DIGIT ISN'T A BLANK
3102 017222 000744                   BR      2S           ;; GO DO THE LAST DIGIT
3103 017224 012605                   6S:     MOV      (SP)+,R5  ;; RESTORE R5
3104 017226 012604                   MOV      (SP)+,R4     ;; RESTORE R4
3105 017230 012603                   MOV      (SP)+,R3     ;; RESTORE R3
3106 017232 016666 000002 000004  MOV      2(SP),4(SP)  ;; SET THE STACK FOR RETURNING
3107 017240 012616                   MOV      (SP)+,(SP)
3108 017242 000002                   RTI
3109 017244 000          8S:     .BYTE    0       ;; RETURN
3110 017245 000          .BYTE    0       ;; STORAGE FOR ASCII DIGIT
3111 017246 000          .BYTE    0       ;; TERMINATOR FOR TYPE ROUTINE
3112 017247 000          $OCNT:   .BYTE    0       ;; OCTAL DIGIT COUNTER
3113 017250 000000          $OFILL:  .BYTE    0       ;; ZERO FILL SWITCH
3114 017252                   $OMODE:  .WORD    0       ;; NUMBER OF DIGITS TO TYPE
3115                   .POWER   ,PWRMSG,START
3116                   .SBTTL  POWER DOWN AND UP ROUTINES
3117 017252
3118
3119 *****
3120 017252 012737 017416 000024  $PWRDN: MOV      #SILLUP,@PWRVEC ;; SET FOR FAST UP
3121 017260 012737 000340 000026  MOV      #340,@PWRVEC+2 ;; PRIO:7
3122 017266                   PUSH     <R0,R1,R2,R3,R4,R5>
3123 017266 010046                   MOV      R0,-(SP)     ;; PUSH R0 ON STACK
3124 017270 010146                   MOV      R1,-(SP)     ;; PUSH R1 ON STACK
3125 017272 010246                   MOV      R2,-(SP)     ;; PUSH R2 ON STACK
3126 017274 010346                   MOV      R3,-(SP)     ;; PUSH R3 ON STACK
3127 017276 010446                   MOV      R4,-(SP)     ;; PUSH R4 ON STACK
3128 017300 010546                   MOV      R5,-(SP)     ;; PUSH R5 ON STACK
3129 017302                   PUSH     @SWR
3130 017302 017746 161632                   MOV      @SWR,-(SP)   ;; PUSH @SWR ON STACK
3131 017306 010637 017422                   MOV      SP,$$AVR6    ;; SAVE SP
3132 017312 012737 017324 000024  MOV      #SPWRUP,@PWRVEC ;; SET UP VECTOR

```

```

3133 017320 000000          HALT
3134 017322 000776          BR      .-2          ;;HANG UP
3135
3136 017324          STARS
3137          ;;*****
3138          :POWER UP ROUTINE
3139 017324 012737 017416 000C24 $PWRUP: MOV    $SILLUP, @#PWRVEC ;;SET FOR FAST DOWN
3140 017332 013706 017422          MOV    $$SAVR6, SP      ;;GET SP
3141 017336 005037 017422          CLR    $$SAVR6        ;;WAIT LOOP FOR THE TTY
3142 017342 005237 017422          IS:   INC    $$SAVR6        ;;WAIT FOR THE INC
3143 017346 001375          BNE    IS              ;;OF WORD
3144 017350          POP    @SWR
3145 017350 012677 161564          MOV    (SP)+, @SWR      ;;POP STACK INTO @SWR
3146 017354          POP    <R5, R4, R3, R2, R1, R0>
3147 017354 012605          MOV    (SP)+, R5      ;;POP STACK INTO R5
3148 017356 012604          MOV    (SP)+, R4      ;;POP STACK INTO R4
3149 017360 012603          MOV    (SP)+, R3      ;;POP STACK INTO R3
3150 017362 012602          MOV    (SP)+, R2      ;;POP STACK INTO R2
3151 017364 012601          MOV    (SP)+, R1      ;;POP STACK INTO R1
3152 017366 012600          MOV    (SP)+, R0      ;;POP STACK INTO R0
3153 017370 012737 017252 000024          MOV    $PWRDN, @#PWRVEC ;;SET UP THE POWER DOWN VECTOR
3154 017376 012737 000340 000026          MOV    #340, @#PWRVEC+2 ;;PRIO:7
3155 017404 104400          TYPE          ;;REPORT THE POWER FAILURE
3156 017406 024574          $PWRMG: .WORD  PWRMSG      ;;POWER FAIL MESSAGE POINTER
3157 017410 012716          MOV    (PC)+, (SP)    ;;RESTART AT START
3158 017412 001704          $PWRAD: .WORD  START      ;;RESTART ADDRESS
3159 017414 000002          RTI
3160 017416 000000          $SILLUP: HALT          ;;THE POWER UP SEQUENCE WAS STARTED
3161 017420 000776          BR      .-2          ;;BEFORE THE POWER DOWN WAS COMPLETE
3162 017422 000000          $$SAVR6: 0           ;;PUT THE SP HERE
3163
3164 017424          .SSCOPE
3165          .SBTTL SCOPE HANDLER ROUTINE
3166
3167 017424          STARS
3168          ;;*****
3169          ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
3170          ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
3171          ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
3172          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3173          ;*SW14=1      LOOP ON TEST
3174          ;*SW11=1      INHIBIT ITERATIONS
3175          ;*CALL
3176          ;*      SCOPE          ;;SCOPE=IOT
3177
3178 017424          $SCOPE:
3179 017424 032777 040000 161506          IS:   BIT    #BIT14, @SWR      ;;LOOP ON PRESENT TEST?
3180 017432 001055          BNE    $OVER          ;;YES IF SW14=1
3181          ;*****START OF CODE FOR THE XOR TESTER*****
3182 017434 000416          $XTSTR: BR    6$      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
3183          ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
3184 017436 013746 000004          MOV    @#ERRVEC, -(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
3185 017442 012737 017462 000004          MOV    $$, @#ERRVEC    ;;SET FOR TIMEOUT
3186 017450 005737 177060          TST    @#177060        ;;TIME OUT ON XOR?
3187 017454 012637 000004          MCV    (SP)+, @#ERRVEC ;;RESTORE THE ERROR VECTOR
3188 017460 000436          BR     $$VLAD          ;;GO TO THE NEXT TEST

```

```

3189 017462 022626          5$:    CMP      (SP)+,(SP)+      ;;CLEAR THE STACK AFTER A TIME OUT
3190 017464 012637 000004    MOV      (SP)+,2*ERRVEC  ;;RESTORE THE ERROR VECTOR
3191 017470 000436          BR       $OVER          ;;LOOP ON THE PRESENT TEST
3192 017472          6$:;*****END OF CODE FOR THE XOR TESTER*****
3193 017472 105737 001103    2$:    TSTB     $ERFLG      ;;HAS AN ERROR OCCURRED?
3194 017476 001404          BEQ      3$            ;;BR IF NO
3195 017500 105037 001103    4$:    CLRB     $ERFLG      ;;ZERO THE ERROR FLAG
3196 017504 005037 001160    CLR      $TIMES        ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
3197 017510 032777 004000 161422 3$:    BIT      #BIT11,$SWR    ;;INHIBIT ITERATIONS?
3198 017516 001011          BNE      1$            ;;BR IF YES
3199 017520 005737 001100    TST      $PASS        ;;IF FIRST PASS OF PROGRAM
3200 017524 001406          BEQ      1$            ;;INHIBIT ITERATIONS
3201 017526 005237 001104    INC      $ICNT        ;;INCREMENT ITERATION COUNT
3202 017532 023737 001160 001104    CMP      $TIMES,$ICNT  ;;CHECK THE NUMBER OF ITERATIONS MADE
3203 017540 002012          BGE      $OVER        ;;BR IF MORE ITERATION REQUIRED
3204 017542 012737 000001 001104 1$:    MOV      #1,$ICNT     ;;REINITIALIZE THE ITERATION COUNTER
3205 017550 013737 017602 001160    MOV      $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
3206 017556 105237 001102    $SVLAD: INCB     $TSTNM   ;;COUNT TEST NUMBERS
3207 017562 011637 001106    MOV      (SP),$LPADR   ;;SAVE SCOPE LOOP ADDRESS
3208 017566 013777 001102 161346 $OVER: MOV      $TSTNM,$DISPLAY ;;DISPLAY TEST NUMBER
3209 017574 013716 001106    MOV      $LPADR,(SP)  ;;FUDGE RETURN ADDRESS
3210 017600 000002          RTI                   ;;FIXES PS
3211 017602 003720    $MXCNT: 2000.        ;;MAX. NUMBER OF ITERATIONS
3212
3213 017604          .SREAD
3214          .SBTTL  TTY INPUT ROUTINE
3215
3216 017604    STARS
3217    ;:*****
3218    .ENABL  LSB
3219
3220    .DSABL  LSB
3221
3222
3223 017604    STARS
3224    ;:*****
3225    ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
3226    ;*CALL:
3227    ;*      RDCHR          ;;INPUT A SINGLE CHARACTER FROM THE TTY
3228    ;*      RETURN HERE   ;;CHARACTER IS ON THE STACK
3229    ;*                    ;;WITH PARITY BIT STRIPPED OFF
3230
3231
3232 017604 011646          $RDCHR: MOV      (SP),-(SP)  ;;PUSH DOWN THE PC
3233 017606 016666 000004 000002    MOV      4(SP),2(SP)  ;;SAVE THE PS
3234 017614 105777 161324    1$:    TSTB     2$TKS      ;;WAIT FOR
3235 017620 100375          BPL      1$            ;;A CHARACTER
3236 017622 117766 161320 000004    MOVB     2$TKB,4(SP)  ;;READ THE TTY
3237 017630 042766 177600 000004    BIC      #1C(17),4(SP) ;;GET RID OF JUNK IF ANY
3238 017636 026627 000004 000023    CMP      4(SP),#23    ;;IS IT A CONTROL-S?
3239 017644 001013          BNE      3$            ;;BRANCH IF NO
3240 017646 105777 161272    2$:    TSTB     2$TKS      ;;WAIT FOR A CHARACTER
3241 017652 100375          BPL      2$            ;;LOOP UNTIL ITS THERE
3242 017654 117746 161266    MOVB     2$TKB,-(SP)  ;;GET CHARACTER
3243 017660 042716 177600    BIC      #1C17,(SP)   ;;MAKE IT 7-BIT ASCII
3244 017664 022627 000021    CMP      (SP)+,#21    ;;IS IT A CONTROL-Q?

```

```

3245 017670 001366          BNE      2$          ;; IF NOT DISCARD IT
3246 017672 000750          BR       1$          ;; YES, RESUME
3247 017674 026627 000004 000140 3$:    CMP      4(SP),#140  ;; IS IT UPPER CASE?
3248 017702 002407          BLT      4$          ;; BRANCH IF YES
3249 017704 026627 000004 000175          CMP      4(SP),#175  ;; IS IT A SPECIAL CHAR?
3250 017712 003003          BGT      4$          ;; BRANCH IF YES
3251 017714 042766 000040 000004          BIC      #40,4(SP)   ;; MAKE IT UPPER CASE
3252 017722 000002          4$:      RTI          ;; GO BACK TO USER
3253 017724
3254
3255
3256
3257
3258
3259
3260
3261 017724 010346          $RDLIN: MOV     R3,-(SP)  ;; SAVE R3
3262 017726 012703 020032 1$:      MOV     #STTYIN,R3  ;; GET ADDRESS
3263 017732 022703 020042 2$:      CMP     #STTYIN+8.,R3  ;; BUFFER FULL?
3264 017736 101405          BLOS     4$          ;; BR IF YES
3265 017740 104404          RDCHR    ;; GO READ ONE CHARACTER FROM THE TTY
3266 017742 112613          MOV     (SP)+,(R3)  ;; GET CHARACTER
3267 017744 122713 000177 10$:    CMP     #177,(R3)  ;; IS IT A RUBOUT
3268 017750 001003          BNE     3$          ;; SKIP IF NOT
3269 017752 104400 001162 4$:      TYPE    $QUES     ;; TYPE A '?'
3270 017756 000763          BR       1$          ;; CLEAR THE BUFFER AND LOOP
3271 017760 111337 020030 3$:      MOV     (R3),9$    ;; ECHO THE CHARACTER
3272 017764 104400 020030          TYPE    9$
3273 017770 122723 000015          CMP     #15,(R3)+  ;; CHECK FOR RETURN
3274 017774 001356          BNE     2$          ;; LOOP IF NOT RETURN
3275 017776 105063 177777          CLRB   -1(R3)     ;; CLEAR RETURN (THE 15)
3276 020002 104400 001164          TYPE    $LF       ;; TYPE A LINE FEED
3277 020006 012603          MOV     (SP)+,R3   ;; RESTORE R3
3278 020010 011646          MOV     (SP),-(SP) ;; ADJUST THE STACK AND PUT ADDRESS OF THE
3279 020012 016666 000004 000002          MOV     4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
3280 020020 012766 020032 000004          MOV     #STTYIN,4(SP)
3281 020026 000002          RTI          ;; RETURN
3282 020030 000          9$:      .BYTE  0          ;; STORAGE FOR ASCII CHAR. TO TYPE
3283 020031 000          .BYTE  0          ;; TERMINATOR
3284 020032 000010          $TTYIN: .BLKB  8.   ;; RESERVE 8 BYTES FOR TTY INPUT
3285 020042 052536 005015 000          $CNTLU: .ASCIZ  /U/<15><12>  ;; CONTROL "U"
3286 020047 136 006507 000012          $CNTLG: .ASCIZ  /G/<15><12>  ;; CONTROL "G"
3287 020054 005015 053523 020122          $MSWR:  .ASCIZ  <15><12>/SWR = /
3288 020062 020075 000
3289 020065 040 047040 053505          $MNEW:  .ASCIZ  / NEW = /
3290 020072 036440 000040
3291 020076
3292
3293
3294 020076          .SRDDEC
3295          .SBTTL READ A DECIMAL NUMBER FROM THE TTY
3296
3297
3298
3299
3300
STARS
*****
*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
*ARE READ A "?" FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS

```

```

3301          ;*POSITIVE 32767 TO NEGATIVE 32768.
3302          ;*CALL:
3303          ;*      RDDEC          ;;READ A DECIMAL NUMBER
3304          ;*      RETURN HERE  ;;NUMBER IS ON TOP OF THE STACK
3305          ;
3306          ;
3307 020076 011646          SRDDEC: MOV      (SP),-(SP)          ;;PROVIDE SPACE FOR
3308 020100 016666 000004 000002 MOV      4(SP),2(SP)          ;;THE INPUT NUMBER
3309 020106          FUSH      (R0,R1,R2)
3310 020106 010046          MOV      RO,-(SP)          ;;PUSH RO ON STACK
3311 020110 010146          MOV      R1,-(SP)          ;;PUSH R1 ON STACK
3312 020112 010246          MOV      R2,-(SP)          ;;PUSH R2 ON STACK
3313 020114 104405          1$:  RDLIN          ;;READ AN ASCIZ LINE
3314 020116 012600          MOV      (SP)+,RO          ;;ADDRESS OF 1ST CHAR.
3315 020120 010037 020244 MOV      RO,6$          ;;SAVE INCASE OF BAD INPUT
3316 020124 005046          CLR      -(SP)          ;;CLEAR DATA WORD
3317 020126 005002          CLR      R2          ;;SIGN SET POSITIVE
3318 020130 122710 000055 CMPB     #'-(,RO)          ;;SEE IF A MINUS SIGN WAS TYPED
3319 020134 001001          BNE     2$          ;;BR IF NO MINUS SIGN
3320 020136 112002          MOVB    (RO)+,R2          ;;SAVE FOR LATER USE
3321 020140 112001          2$:  MOVB    (RO)+,R1          ;;PICKUP THIS CHARACTER
3322 020142 001424          BEQ     3$          ;;GET OUT IF ZERO
3323 020144 122701 000060 CMPB     #'0,R1          ;;MAKE SURE THIS CHARACTER
3324 020150 003032          BGT     5$          ;;IS A DIGIT BETWEEN 0 & 9
3325 020152 122701 000071 CMPB     #'9,R1
3326 020156 002427          BLT     5$
3327 020160 032716 170000 BIT      #1C7777,(SP)          ;;DON'T LET NUMBER GET TO BIG
3328 020164 001024          BNE     5$          ;;BR IF NUMBER WOULD OVERFLOW
3329 020166 006316          ASL     (SP)          ;;*2
3330 020170 011646          MOV     (SP),-(SP)          ;;SAVE FOR LATER
3331 020172 006316          ASL     (SP)          ;;*4
3332 020174 006316          ASL     (SP)          ;;*8
3333 020176 062616          ADD     (SP)+,(SP)          ;;*10
3334 020200 102416          BVS     5$          ;;OVERFLOW ISN'T ALLOWED
3335 020202 162701 000060 SUB      #'0,R1          ;;STRIP AWAY THE ASCII JUNK
3336 020206 060116          ADD     R1,(SP)          ;;ADD IN THIS DIGIT
3337 020210 102412          BVS     5$          ;;OVERFLOW ISN'T ALLOWED
3338 020212 000752          BR      2$          ;;LOOP
3339 020214 005702          3$:  TST     R2          ;;CHECK IF NUMBER IS NEG
3340 020216 001401          BEQ     4$          ;;BR IF NO
3341 020220 005416          NEG     (SP)          ;;YES--NEGATE THE NUMBER
3342 020222 012666 000012 4$:  MOV     (SP)+,12(SP)          ;;SAVE THE RESULT
3343 020226          POP     (R2,R1,RO)
3344 020226 012602          MOV     (SP)+,R2          ;;POP STACK INTO R2
3345 020230 012601          MOV     (SP)+,R1          ;;POP STACK INTO R1
3346 020232 012600          MOV     (SP)+,RO          ;;POP STACK INTO RO
3347 020234 000002          RTI
3348          ;
3349 020236 005726          5$:  TST     (SP)+          ;;CLEAN PARTIAL NUMBER FROM STACK
3350 020240 105010          CLRB   (RO)          ;;SET A TERMINATOR
3351 020242 104400          TYPE          ;;TYPE THE INPUT UP TO BAD CHAR.
3352 020244 000000          6$:  .WORD   0          ;;POINTER GOES HERE
3353 020246 104400 001162 TYPE     $QUES          ;;?" "CR" &"LF"
3354 020252 000720          BR      1$          ;;TRY AGAIN
3355          ;
3356 020254          .STYPE
    
```

```

3357
3358
3359 020254
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371
3372
3373
3374
3375 020254 105737 001157
3376 020260 100002
3377 020262 000000
3378 020264 000407
3379 020266 010046
3380 020270 017600 000002
3381 020274 112046
3382 020276 001005
3383 020300 005726
3384 020302 012600
3385 020304 062716 000002
3386 020310 000002
3387 020312 122716 000011
3388 020316 001430
3389 020320 122716 000200
3390 020324 001006
3391 020326 005726
3392 020330 104400
3393 020332 001163
3394 020334 105037 020470
3395 020340 000755
3396 020342 004737 020424
3397 020346 123726 001156
3398 020352 001350
3399 020354 013746 001154
3400
3401 020360 105366 000001
3402 020364 002770
3403 020366 004737 020424
3404 020372 105337 020470
3405 020376 000770
3406
3407
3408
3409 020400 112716 000040
3410 020404 004737 020424
3411 020410 132737 000007 020470
3412 020416 001372

```

.SBTTL TYPE ROUTINE

STARS

```

*****
;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
;*

```

```

$TYPE: TSTB $TFPLG ;; IS THERE A TERMINAL?
BPL 1$ ;; BR IF YES
HALT ;; HALT HERE IF NO TERMINAL
BR 3$ ;; LEAVE
1$: MOV RO, -(SP) ;; SAVE RO
MOV @2(SP), RO ;; GET ADDRESS OF ASCIZ STRING
2$: MOVB (RO)+, -(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
BNE 4$ ;; BR IF IT ISN'T THE TERMINATOR
TST (SP)+ ;; IF TERMINATOR POP IT OFF THE STACK
60$: MOV (SP)+, RO ;; RESTORE RO
3$: ADD #2, (SP) ;; ADJUST RETURN PC
RTI ;; RETURN
4$: CMPB #HT, (SP) ;; BRANCH IF <HT>
BEQ 8$
CMPB #CRLF, (SP) ;; BRANCH IF NOT <CRLF>
BNE 5$
TST (SP)+ ;; POP <CR><LF> EQUIV
; ; TYPE A CR AND LF
3394: CLRB $CHARCNT ;; CLEAR CHARACTER COUNT
BR 2$ ;; GET NEXT CHARACTER
3396: JSR PC, $TYPEPC ;; GO TYPE THIS CHARACTER
3397: CMPB $FILLC, (SP)+ ;; IS IT TIME FOR FILLER CHARS.?
BNE 2$ ;; IF NO GO GET NEXT CHAR.
MOV $NULL, -(SP) ;; GET # OF FILLER CHARS. NEEDED
; ; AND THE NULL CHAR.
3401: DECB 1(SP) ;; DOES A NULL NEED TO BE TYPED?
BLT 6$ ;; BR IF NO--GO POP THE NULL OFF OF STACK
3403: JSR PC, $TYPEPC ;; GO TYPE A NULL
DECB $CHARCNT ;; DO NOT COUNT AS A COUNT
BR 7$ ;; LOOP

```

;HORIZONTAL TAB PROCESSOR

```

8$: MOVB #' (SP) ;; REPLACE TAB WITH SPACE
9$: JSR PC, $TYPEPC ;; TYPE A SPACE
BITB #7, $CHARCNT ;; BRANCH IF NOT AT
BNE 9$ ;; TAB STOP

```

```

3413 020420 005726          TST      (SP)+          ;; POP SPACE OFF STACK
3414 020422 000724          BR       2$              ;; GET NEXT CHARACTER
3415 020424 105777 160520  $TYPEPC: TSTB   @STPS              ;; WAIT UNTIL PRINTER IS READY
3416 020430 100375          BPL     $TYPEPC
3417 020432 116677 000002 160512  MOVB    2(SP),@STPB      ;; LOAD CHAR TO BE TYPED INTO DATA REG.
3418 020440 122766 000015 000002  CMPB    #CR,2(SP)       ;; IS CHARACTER A CARRIAGE RETURN?
3419 020446 001003          BNE     1$              ;; BRANCH IF NO
3420 020450 105037 020470          CLRB   $CHARCNT        ;; YES--CLEAR CHARACTER COUNT
3421 020454 000406          BR     $TYPEX
3422 020456 122766 000012 000002 1$:  CMPB   #LF,2(SP)       ;; IS CHARACTER A LINE FEED?
3423 020464 001402          BEQ    $TYPEX           ;; BRANCH IF YES
3424 020466 105227          INCB   (PC)+           ;; COUNT THE CHARACTER
3425 020470 000000          $CHARCNT: .WORD      0 ;; CHARACTER COUNT STORAGE
3426 020472 000207          $TYPEX: RTS           PC
3427
3428
3429
3430
3431
3432 020474          .STRAP
3433          .SBTTL TRAP DECODER
3434
3435 020474          STARS
3436          ;;*****
3437          ;;THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
3438          ;;AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3439          ;;OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
3440          ;;GO TO THAT ROUTINE.
3441
3442 020474 010046          $STRAP: MOV     R0, -(SP)      ;; SAVE R0
3443 020476 016600 000002  MOV     2(SP),R0        ;; GET TRAP ADDRESS
3444 020502 005740          TST     -(R0)          ;; BACKUP BY 2
3445 020504 111000          MOVB   (R0),R0        ;; GET RIGHT BYTE OF TRAP
3446 020506 006300          ASL    R0              ;; POSITION FOR INDEXING
3447 020510 016000 020516  MOV     $TRPAD(R0),R0  ;; INDEX TO TABLE
3448 020514 000200          RTS     R0            ;; GO TO ROUTINE
3449
3450 020516          SETTRAP TYPE,$TYPE,↑/TTY TYPEOUT ROUTINE/
3451 020516          $$SET TYPE,$TYPE,\<TRAP+$STRAP>,\$STRP,\<TTY TYPEOUT ROUTINE>
3452          .SBTTL TRAP TABLE
3453
3454          ;;THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3455          ;;BY THE "TRAP" INSTRUCTION.
3456
3457          ; ROUTINE
3458          ; -----
3459          $TRPAD:
3460 020516 020254          $TYPE  ;;CALL=TYPE TRAP+0(104400) TTY TYPEOUT ROUTINE
3461 020520          SETTRAP ↑TYPOC,$TYPOC,↑/TYPE OCTAL NUMBER (WITH LEADING ZEROS)/
3462 020520          $$SET TYPOC,$TYPOC,\<TRAP+$STRP>,\$STRP,\<TYPE OCTAL NUMBER (WITH LEADING ZEROS)>
3463 020520 017050          $TYPOC ;;CALL=TYPOC TRAP+1(104401) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3464 020522          SETTRAP ↑TYPOS,$TYPOS,↑/TYPE OCTAL NUMBER (NO LEADING ZEROS)/
3465 020522          $$SET TYPOS,$TYPOS,\<TRAP+$STRP>,\$STRP,\<TYPE OCTAL NUMBER (NO LEADING ZEROS)>
3466 020522 017024          $TYPOS ;;CALL=TYPOS TRAP+2(104402) TYPE OCTAL NUMBER (NO LEADING ZEROS)
3467 020524          SETTRAP ↑TYPON,$TYPON,↑/TYPE OCTAL NUMBER (AS PER LAST CALL)/
3468 020524          $$SET TYPON,$TYPON,\<TRAP+$STRP>,\$STRP,\<TYPE OCTAL NUMBER (AS PER LAST CALL)>
  
```

Address	Code	Label	Operation	Trap #	Type	Octal Number	Description
3469	020524	017064	\$TYPON	::CALL=TYPON	TRAP+3(104403)		TYPE OCTAL NUMBER (AS PER LAST CALL)
3470							
3471							
3472	020526		SETTRAP	RDCHR,\$RDCHR,↑/TTY TYPEIN CHARACTER ROUTINE/			
3473	020526		\$\$SET	RDCHR,\$RDCHR,\<TRAP+\$TRP>,\$TRP,<TTY TYPEIN CHARACTER ROUTINE>			
3474	020526	017604	\$RDCHR	::CALL=RDCHR	TRAP+4(104404)		TTY TYPEIN CHARACTER ROUTINE
3475	020530		SETTRAP	RDLIN,\$RDLIN,↑/TTY TYPEIN STRING ROUTINE/			
3476	020530		\$\$SET	RDLIN,\$RDLIN,\<TRAP+\$TRP>,\$TRP,<TTY TYPEIN STRING ROUTINE>			
3477	020530	017724	\$RDLIN	::CALL=RDLIN	TRAP+5(104405)		TTY TYPEIN STRING ROUTINE
3478	020532		SETTRAP	RDDEC,\$RDDEC,↑/READ A DECIMAL NUMBER FROM TTY/			
3479	020532		\$\$SET	RDDEC,\$RDDEC,\<TRAP+\$TRP>,\$TRP,<READ A DECIMAL NUMBER FROM TTY>			
3480	020532	020076	\$RDDEC	::CALL=RDDEC	TRAP+6(104406)		READ A DECIMAL NUMBER FROM TTY
3481	020534		SETTRAP	SAVREG,\$SAVREG,↑/SAVE R0-R5 ROUTINE/			
3482	020534		\$\$SET	SAVREG,\$SAVREG,\<TRAP+\$TRP>,\$TRP,<SAVE R0-R5 ROUTINE>			
3483	020534	016150	\$SAVREG	::CALL=SAVREG	TRAP+7(104407)		SAVE R0-R5 ROUTINE
3484	020536		SETTRAP	RESREG,\$RESREG,↑/RESTORE R0-R5 ROUTINE/			
3485	020536		\$\$SET	RESREG,\$RESREG,\<TRAP+\$TRP>,\$TRP,<RESTORE R0-R5 ROUTINE>			
3486	020536	016206	\$RESREG	::CALL=RESREG	TRAP+10(104410)		RESTORE R0-R5 ROUTINE
3487							
3488	020540		SETTRAP	INOCT,INOCTR,<INPUT OCTAL ROUTINE>			
3489	020540		\$\$SET	INOCT,INOCTR,\<TRAP+\$TRP>,\$TRP,<INPUT OCTAL ROUTINE>			
3490	020540	013746	INOCTR	::CALL=INOCT	TRAP+11(104411)		INPUT OCTAL ROUTINE
3491	020542		SETTRAP	INAR,RINA,<INPUT INPUT ADDR. ROUTINE>			
3492	020542		\$\$SET	INAR,RINA,\<TRAP+\$TRP>,\$TRP,<INPUT INPUT ADDR. ROUTINE>			
3493	020542	013314	RINA	::CALL=INAR	TRAP+12(104412)		INPUT INPUT ADDR. ROUTINE
3494	020544		SETTRAP	OUTAR,ROUTA,<INPUT OUTPUT ADDR. ROUTINE>			
3495	020544		\$\$SET	OUTAR,ROUTA,\<TRAP+\$TRP>,\$TRP,<INPUT OUTPUT ADDR. ROUTINE>			
3496	020544	013406	ROUTA	::CALL=OUTAR	TRAP+13(104413)		INPUT OUTPUT ADDR. ROUTINE
3497	020546		SETTRAP	PATAR,RPATA,<INPUT PATTERN ROUTINE>			
3498	020546		\$\$SET	PATAR,RPATA,\<TRAP+\$TRP>,\$TRP,<INPUT PATTERN ROUTINE>			
3499	020546	016024	RPATA	::CALL=PATAR	TRAP+14(104414)		INPUT PATTERN ROUTINE
3500	020550		SETTRAP	DELA,RDELA,<INPUT DELAY TIME ROUTINE>			
3501	020550		\$\$SET	DELA,RDELA,\<TRAP+\$TRP>,\$TRP,<INPUT DELAY TIME ROUTINE>			
3502	020550	015060	RDELA	::CALL=DELA	TRAP+15(104415)		INPUT DELAY TIME ROUTINE
3503	020552		SETTRAP	DELAY2,RDELA2,<SECONDARY DELAY ROUTINE>			
3504	020552		\$\$SET	DELAY2,RDELA2,\<TRAP+\$TRP>,\$TRP,<SECONDARY DELAY ROUTINE>			
3505	020552	015602	RDELA2	::CALL=DELAY2	TRAP+16(104416)		SECONDARY DELAY ROUTINE
3506	020554		SETTRAP	DELAY,RDELAY,<ROUTINE TO DELAY XX MILLISEC>			
3507	020554		\$\$SET	DELAY,RDELAY,\<TRAP+\$TRP>,\$TRP,<ROUTINE TO DELAY XX MILLISEC>			
3508	020554	015544	RDELAY	::CALL=DELAY	TRAP+17(104417)		ROUTINE TO DELAY XX MILLISEC
3509	020556		SETTRAP	CPATR,RCPAT,<ROUTINE TO CHANGE PATTERNS>			
3510	020556		\$\$SET	CPATR,RCPAT,\<TRAP+\$TRP>,\$TRP,<ROUTINE TO CHANGE PATTERNS>			
3511	020556	015674	RCPAT	::CALL=CPATR	TRAP+20(104420)		ROUTINE TO CHANGE PATTERNS
3512	020560		SETTRAP	IDAC,RDACA,<ROUTINE TO INPUT DAC ADDR>			
3513	020560		\$\$SET	IDAC,RDACA,\<TRAP+\$TRP>,\$TRP,<ROUTINE TO INPUT DAC ADDR>			
3514	020560	013572	RDACA	::CALL=IDAC	TRAP+21(104421)		ROUTINE TO INPUT DAC ADDR
3515							
3516	020562		SETTRAP	CNTAR,RCNTA,<ROUTINE TO INPUT COUNTER MODULE ADDR>			
3517	020562		\$\$SET	CNTAR,RCNTA,\<TRAP+\$TRP>,\$TRP,<ROUTINE TO INPUT COUNTER MODULE ADDR>			
3518	020562	013500	RCNTA	::CALL=CNTAR	TRAP+22(104422)		ROUTINE TO INPUT COUNTER MODULE ADDR
3519	020564		SETTRAP	ADAR,RADA,<ROUTINE TO INPUT ADDS ADDR>			
3520	020564		\$\$SET	ADAR,RADA,\<TRAP+\$TRP>,\$TRP,<ROUTINE TO INPUT ADDS ADDR>			
3521	020564	013660	RADA	::CALL=ADAR	TRAP+23(104423)		ROUTINE TO INPUT ADDS ADDR
3522	020566		SETTRAP	INITR,RINIT,<ROUTINE TO ISSUE SYSTEM INITIALIZE>			
3523	020566		\$\$SET	INITR,RINIT,\<TRAP+\$TRP>,\$TRP,<ROUTINE TO ISSUE SYSTEM INITIALIZE>			
3524	020566	015642	RINIT	::CALL=INITR	TRAP+24(104424)		ROUTINE TO ISSUE SYSTEM INITIALIZE

022231	015	041412	047117	EM3:	.ASCIZ	<15><12>/CONVERT BIT FAILED TO SET/
022265	015	041412	047117	EM4:	.ASCIZ	<15><12>/CONVERT BIT FAILED TO CLEAR/
022323	015	041412	047101	EM5:	.ASCIZ	<15><12>/CAN'T READ DATA REGISTER/
022356	005015	030101	032460	EM6:	.ASCIZ	<15><12>/A005 FAILED TO INTERRUPT/
022411	015	040412	042104	EM7:	.ASCIZ	<15><12>/ADDR. OR GENERIC CODE INCORRECT ON INTERRUPT/
022470	005015	044522	020106	EM10:	.ASCIZ	<15><12>/RIF DID NOT CLEAR INTR FLAG ON A005/
022536	051105	047522	020122	DH1:	.ASCII	?ERROR A/D?
022552	005015	020040	041520		.ASCIZ	<15><12>/ PC ADDR/
022571	105	051122	051117	DH2:	.ASCII	/ERROR MODULE GOOD BAD/
022625	015	020012	050040		.ASCIZ	<15><12>/ PC ADDR DATA DATA/
022665	105	051122	051117	DH3:	.ASCII	/ERROR MODULE ICAR/
022713	015	020012	050040		.ASCIZ	<15><12>/ PC ADDR EXP'D REC'D/
022753	015	047012	020117	MNAE:	.ASCIZ	<15><12>/NO COUNTER MODULE ADDRS. ENTERED/
023015	015	041412	052517	EM11:	.ASCIZ	<15><12>/COULD NOT READ-WRITE INTO COUNTER MODULE BUFFER/
023077	015	020012	051105	DH4:	.ASCII	<15><12>/ ERROR COUNTER/
023120	005015	020040	041520		.ASCIZ	<15><12>/ PC ADDR/
023140	005015	047503	047125	EM12:	.ASCIZ	<15><12>/COUNTER MODULE-UP-COUNT INCORRECT/
023204	005015	047503	047125	EM13:	.ASCIZ	<15><12>/COUNTER MODULE FAILED TO INTERRUPT/
023251	015	041412	052517	EM14:	.ASCIZ	<15><12>/COUNTER MODULE-ADDR. OR GENERIC CODE INCORRECT ON INTERRUPT/
023347	015	041412	052517	EM15:	.ASCIZ	<15><12>/COUNTER MODULE DIDN'T STOP COUNTING ON OVERFLOW/
023431	015	051012	043111	EM16:	.ASCIZ	<15><12>/RIF DIDN'T CLEAR INTERRUPT FLAG ON COUNTER/
023506	005015	054523	027123	EM17:	.ASCIZ	<15><12>/SYS. INIT. FAILED TO CLEAR COUNTER BIT(S)/
023562	005015	046111	042514	EM20:	.ASCIZ	<15><12>/ILLEGAL INTERRUPT POSTED ON ICS BUS/
023630	005015	041511	020123	EM21:	.ASCIZ	<15><12>/ICS FAILED TO INTERRUPT CPU/
023666	005015	046111	042514	EM22:	.ASCIZ	<15><12>/ILLEGAL ADDR. OR GEN CODE BITS IN ICAR/
023737	015	051412	051531	EM23:	.ASCIZ	<15><12>/SYS. INIT. FAILED TO CLEAR COUNTER INTR. FLAG/
024017	015	041412	052517	EM24:	.ASCIZ	<15><12>/COUNTER STARTED COUNTING AFTER SYS. INITIALIZE
024100	005015	030101	032460	EM25:	.ASCIZ	<15><12>/A005 READ DUAL ADDR. ERROR/
024135	015	040412	030060	EM26:	.ASCIZ	<15><12>/A005 DUAL ADDR. ERROR/
024165	015	051412	047105	EM27:	.ASCIZ	<15><12>/SEND-RECEIVE DATA ERROR/
024217	015	042412	051122	DH5:	.ASCII	<15><12>/?ERROR A/D DUAL?
024245	015	020012	050040		.ASCIZ	<15><12>/ PC ADDR ADDR/
024276	005015	027056	031061	MOURB:	.ASCIZ	<15><12>/6.12 UNIPOLAR OR BIPOLAR (U OR B)
024344	005015	047516	040440	MNAD:	.ASCIZ	<15><12>/?NO A/D ADDR. ENTERED?

024373	015	051412	030127	MSWO:	.ASCIZ	<15><12>/SWOO=1 MEANS RETURN TO MONITOR/
024434	030455	027060	030060	MMOVFL:	.ASCIZ	/-10.0000 (OR OVERFLOW)/
024463	053	030051	030056	MPOVFL:	.ASCIZ	/+10.0000 (OR OVERFLOW)/
024512	005015	042522	042520	MREP:	.ASCIZ	<15><12>/REPEAT/
024523	040	042522	042101	MCALOT:	.ASCIZ	/ READING (OCTAL)/
024544	053040	046117	051524	MCALT1:	.ASCIZ	/ VOLTS (DECIMAL) = /
024570	000055			MMINUS:	.ASCIZ	/-/
024572	000053			MPLUS:	.ASCIZ	/+ /
024574	005015	042522	052123	PWRMSG:	.ASCIZ	<15><12>/RESTARTING AFTER A POWER FAILURE/<15><12><12>
024642	005001	042524	052123	MHT0:	.ASCIZ	<1><12>/TEST 0 - INPUT AND OUTPUT MODULE EXERCISER/<15><12>
024721	015	052012	051505	MHT1:	.ASCIZ	<15><12>/TEST 1 - INPUT OR OUTPUT MODULE EXERCISER/<15><12>
024777	015	052012	051505	MHT2:	.ASCIZ	<15><12>/TEST 2 - DAC CALIBRATION/<15><12>
025034	005015	042524	052123	MHT3:	.ASCIZ	<15><12>/TEST 3 - DAC INTERACTION/<15><12>
025071	015	052012	051505	MHT4:	.ASCIZ	<15><12>/TEST 4 - COUNTER MODULE TEST/<15><12>
025132	005015	042524	052123	MHT5:	.ASCIZ	<15><12>?TEST 5 - A/D LOGIC TEST?<15><12>
025166	005015	042524	052123	MHT6:	.ASCIZ	<15><12>?TEST 6 - A/D CALIBRATION?<15><12>
025223	015	052012	051505	MHT8:	.ASCIZ	<15><12>?TEST 7 - A/D REPEATABILITY?<15><12>
025262	005015	027070	020064	MLPAV:	.ASCIZ	<15><12>/8.4 LINE PRINTER OPTION AVAILABLE/
025326	005015	047111	040526	MIVP:	.ASCIZ	<15><12>/INVALID PATTERN MODIFIER/
025361	015	033012	030456	MGAIN:	.ASCIZ	<15><12>/6.13 GAIN /
025376	005015	027066	032061	MCHAN:	.ASCIZ	<15><12>/6.14 CHANS (SC,EC)/
025423	015	033012	030456	MTOL:	.ASCIZ	<15><12>/6.16 TOLERANCE /
025445	015	042412	051122	MCHER1:	.ASCIZ	<15><12>/ERROR! STARTING CHAN. GREATER THAN END CHAN./
025524	005015	040514	052123	MINNN:	.ASCIZ	<15><12>/LAST CHARACTER NOT A OCTAL DIGIT--RETYPE IT. /
025605	015	037412	047111	MINKN:	.ASCIZ	<15><12>/?INPUT NOT UNDERSTOOD--RETYPE IT./
025651	015	042412	051122	MCHANH:	.ASCIZ	<15><12>/ERROR! NO SUCH CHAN./
025700	005015	027066	032461	MAVEQ:	.ASCIZ	<15><12>/6.15 EXPECTED AVERAGE /
025731	015	047012	020117	MMSG:	.ASCIZ	<15><12>/NO SUCH GAIN! /
025751	015	047012	046525	MNTL:	.ASCIZ	<15><12>/NUMBER TO LARGE-MAX=7777/
026004	020040	000		M2SP:	.ASCIZ	/ /
026007	015	051012	050105	MREPFT:	.ASCIZ	<15><12>/REPEATIBILITY FORCED TYP0UT/
026045	015	051012	050105	MREPER:	.ASCIZ	<15><12>/REPEATABILITY ERROR! /

026074	005015	041440	040510	MREPT1:	.ASCIZ	<15><12>/	CHAN	GAIN	LOW	AVER	HIGH/<15><12>		
026146	005015	046040	020117	MREPT2:	.ASCII	<15><12>/	LO	-5	-4	-3	-2	-1	AV/
026211	040	025440	020061		.ASCIZ		+1	+2	+3	+4	+5	HI/<15><12>	
026251	015	033012	030456	MFILE:	.ASCIZ	<15><12>/	6.1 FILE BOX TO BE TESTED/						
026305	015	033012	031056	MVECT:	.ASCIZ	<15><12>/	6.2 ICS VECTOR ADDRESS/						
026336	005015	046111	042514	ILLEG:	.ASCIZ	<15><12>/	ILLEGAL NUMBER/<15><12>						
026361	015	040412	042104	NRANG1:	.ASCIZ	<15><12>/	ADDRESS /						
026374	047040	052117	053440	NRANG2:	.ASCIZ	/	NOT WITHIN FILE BOX RANGE/<15><12>						
026431	015	043012	046111	NOINT:	.ASCIZ	<15><12>/	FILE BOX DID NOT INTERRUPT-FATAL/						
026474	005015	044506	042514	FILINT:	.ASCIZ	<15><12>/	FILE BOX INTERRUPTED AT /						
026527	040	026455	041440	CKJMP:	.ASCIZ	/	-- CHECK JUMPERS /						
026552	005015	047516	026516	NON*ST:	.ASCIZ	<15><12>/	NON-EXISTENT FILE BOX/						

3537	026602	015	012
3538	026604	000000	
3539		026636	
3540	026636		
3541			
3542			
3543	026636		
3544	000001		

```

OUTBF1: .BYTE 15,12
OUTBF: 0
      .=. +30
BUFFER:
;*END ADDRESS IS FUNCTION OF HOW MANY CHANS. ARE BEING EXERCISED
;*AT ONE TIME IN TEST 7.
$ENDAD=.
      .END

```


.SERRO	1#	4#	2953
.SERRT	1#	3#	2995
.SMULT	1#		
.SPOWE	1#	3#	3114
.SRAND	1#	5#	2825
.SRODE	1#	4#	3291
.SROOC	1#		
.SREAO	1#	4#	3213
.SRTAN	1#		
.SSIVN	1#	5#	2776
.SSB2D	1#	5#	2864
.SSB2O	1#		
.SSCOP	1#	4#	3164
.SSIZE	1#		
.SSUPR	1#		
.STRAP	1#	4#	3432
.STYPB	1#		
.STYPD	1#		
.STYPR	1#	3#	3356
.STYPC	1#	3#	3035
.S40CA	1#		
.:17C	1#		

ADC	1970	2004	2545	2567	2850	2853	2916								
ADD	530	535	558	634	635	649	650	653	655	656	659	660	663	664	724
	901	1064	1086	1181	1402	1495	1496	1638	1664	1668	1800	1870	1954	1968	1969
	1971	1975	1984	2055	2060	2068	2072	2082	2086	2095	2347	2429	2431	2441	2456
	2544	2566	2638	2716	2773	2849	2851	2852	2854	2880	2915	2917	3010	3065	3075
	3333	3336	3385												
ASL	640	723	1623	1889	1901	1902	1903	1904	2771	2845	3007	3008	3009	3329	3331
	3332	3446													
ASR	1461	1935	1936	1937	1938	1998	2002								
BCC	1065	1156													
BCCS	2713														
BEC	528	673	692	747	793	799	822	829	852	857	877	888	896	898	929
	1010	1019	1033	1117	1143	1150	1172	1175	1177	1196	1213	1250	1298	1307	1396
	1413	1464	1498	1500	1565	1574	1598	1622	1628	1642	1666	1835	1842	1856	1866
	1898	1929	1983	2240	2264	2289	2312	2332	2371	2331	2393	2452	2454	2463	2511
	2521	2541	2564	2601	2662	2676	2696	2710	2715	2767	2969	3012	3017	3030	3092
	3194	3200	3322	3340	3388	3423									
BGE	521	2058	2395	3203											
BGT	504	605	1872	1960	1963	2413	2443	2461	2629	3099	3250	3324			
BHT	1670														
BIC	745	873	878	892	953	958	969	976	979	1062	1148	1258	1460	1724	1776
	1920	2017	2024	2038	2045	2363	2406	2407	2596	2772	3089	3237	3243	3251	
BICB	1008	1293													
BIS	607	788	858	874	882	954	959	970	977	980	1296	1312	1378	1462	1473
	1492	1558	1646	1725	1845	1921	1922	2242	2266	2290	2313	2333	2408	2699	2919
	3094	3095													
BISB	2998														
BIT	851	1032	1564	1570	1627	1636	1834	1841	1865	2695	2766	2975	3179	3197	3327
BITB	3411														
BLE	554	718	1404	1814	1819	1874	1909	1987	2084	2293	2355	2399	2631		
BLOS	2245	2269	3264												
BLT	552	1183	1977	1981	2912	3100	3248	3326	3402						
BMI	716	1728	1925	2547	2569										
BNE	460	481	680	774	776	796	803	808	810	813	818	824	838	914	996
	1012	1021	1055	1079	1105	1111	1127	1385	1433	1503	1510	1560	1571	1600	1606
	1611	1618	1626	1637	1640	1648	1656	1674	1758	1771	1778	1782	1911	1924	1944
	1847	1892	1911	1967	1973	2001	2006	2062	2070	2074	2078	2088	2229	2253	2277
	2301	2321	2365	2373	2376	2379	2392	2385	2388	2411	2415	2425	2428	2437	2447
	2459	2532	2549	2571	2584	2599	2606	2665	2680	2720	2752	2848	2923	2976	2999
	3022	3090	3143	3180	3198	3239	3245	3268	3274	3319	3328	3382	3390	3398	3412
	3419														
BPL	626	850	944	1003	1005	1236	1316	1334	1420	1795	1994	2361	2368	2591	2593
	2667	2692	2981	3088	3235	3241	3376	3416							
BR	483	515	516	523	531	532	541	546	547	556	574	592	615	623	641
	710	711	721	880	899	906	1015	1045	1066	1087	1115	1146	1248	1295	1302
	1310	1415	1453	1469	1479	1484	1515	1580	1596	1633	1615	1763	1764	1774	1804
	1805	1817	1822	1829	1830	1838	1850	1851	1860	1878	1994	2234	2235	2258	2259
	2282	2283	2306	2307	2326	2327	2409	2554	2575	2603	2760	2761	2769	2914	3005
	3032	3066	3081	3102	3134	3161	3182	3188	3191	3246	3270	3338	3354	3378	3395
	3405	3414	3421												
BVS	633	3334	3337												
CLC	2711														
CLR	458	471	492	585	591	616	621	645	667	671	677	678	691	682	683
	687	688	689	694	699	727	763	764	770	772	779	780	781	789	842
	846	853	861	862	879	994	1001	1040	1083	1151	1168	1191	1194	1208	1244
	1255	1257	1280	1282	1294	1295	1305	1379	1447	1451	1476	1491	1616	1644	1661

	1667	1729	1785	1786	1787	1788	1791	1926	1951	1955	1956	2056	2066	2079	2350
	2351	2353	2357	2358	2359	2417	2448	2503	2535	2542	2543	2565	2589	2609	2678
CLRB	2717	2758	2759	2908	2997	3079	3141	3196	3316	3317					
CMP	934	935	1657	2924	3195	3275	3350	3394	3420						
	459	480	502	503	520	551	553	576	603	604	679	717	773	795	802
	812	817	895	897	1110	1142	1174	1176	1182	1297	1395	1473	1412	1432	1463
	1497	1499	1502	1509	1573	1599	1605	1610	1617	1647	1655	1673	1777	1813	1818
	1871	1873	1908	1959	1962	1976	1980	1986	2057	2069	2083	2244	2268	2292	2354
	2364	2370	2372	2375	2390	2392	2394	2398	2412	2442	2446	2451	2453	2460	2597
CMPB	2600	2605	2628	2630	2709	2714	2719	2918	3189	3202	3238	3244	3247	3249	3263
	746	1009	1011	1018	1020	1669	2378	2381	2384	2387	3267	3273	3318	3323	3325
	3387	3389	3397	3418	3422										
COM	1631	2520													
DEC	527	1796	1864	1891	1972	2000	2061	2073	2087	2421	2424	2427	2449	2548	2570
	2664	2666	2679	2681	2921	3006									
DECB	3087	3098	3401	3404											
EMT	42														
HALT	28	587	2982	3133	3160	3377									
INC	526	609	834	843	905	945	1112	1170	1171	1522	1632	1672	1792	1869	1907
	1910	1985	2059	2071	2076	2077	2085	2401	2405	2445	2465	2498	2514	2607	2712
	2847	2913	2971	3093	3101	3142	3201								
INCB	1651	1653	2968	3206	3424										
IOT	43														
JMP	32	35	588	697	728	749	811	819	831	835	917	986	1013	1034	1058
	1068	1082	1089	1107	1338	1525	1538	1563	1566	1732	1867	2092	2366	2374	2377
	2380	2383	2386	2389	2396	2400	2439	2457	2466	2470	2474	2478	2481	2501	2506
	2515	2524	2572	2599	2697										
JSR	925	927	930	1567	1569	1583	1589	1756	2243	2267	2291	2315	2335	2741	2879
	2977	3396	3403	3410											
MOV	457	461	463	464	465	466	467	468	469	470	472	476	477	478	479
	484	486	487	488	490	491	498	499	500	501	512	514	524	525	532
	534	536	537	542	543	545	557	559	565	566	567	568	569	570	573
	575	581	586	589	598	599	600	601	602	606	608	620	625	627	628
	629	630	636	637	638	643	644	646	648	651	654	657	661	665	670
	674	675	676	684	685	690	693	698	700	701	705	708	709	722	725
	726	744	766	767	768	769	771	785	787	790	791	792	794	800	801
	804	809	825	826	827	839	840	841	844	847	871	875	881	883	884
	886	893	894	902	903	923	924	926	928	936	942	948	956	962	972
	1029	1030	1042	1043	1061	1063	1085	1099	1102	1103	1109	1116	1118	1124	1125
	1126	1136	1137	1138	1139	1140	1141	1149	1163	1164	1165	1167	1169	1173	1189
	1190	1195	1206	1207	1209	1212	1242	1243	1252	1254	1272	1273	1274	1276	1277
	1278	1289	1290	1291	1306	1320	1328	1329	1330	1374	1377	1380	1381	1382	1383
	1384	1392	1393	1394	1408	1409	1410	1411	1417	1431	1442	1443	1444	1445	1446
	1458	1459	1475	1490	1494	1501	1508	1559	1568	1584	1598	1591	1615	1619	1620
	1624	1645	1649	1662	1723	1726	1753	1754	1761	1762	1769	1775	1779	1789	1797
	1798	1802	1803	1812	1827	1828	1839	1840	1857	1858	1863	1868	1975	1984	1995
	1887	1895	1896	1897	1898	1900	1906	1912	1919	1923	1927	1950	1952	1953	1957
	1958	1961	1964	1965	1974	1997	2012	2016	2019	2023	2026	2037	2040	2044	2047
	2051	2053	2054	2064	2065	2067	2080	2081	2093	2096	2232	2233	2241	2256	2257
	2265	2280	2281	2287	2304	2305	2314	2324	2325	2334	2346	2348	2352	2356	2369
	2416	2418	2419	2422	2423	2426	2430	2435	2438	2444	2464	2536	2540	2563	2594
	2595	2602	2618	2619	2627	2636	2640	2648	2661	2663	2675	2677	2693	2721	2723
	2742	2756	2757	2765	2774	2797	2798	2799	2800	2801	2802	2803	2804	2805	2806
	2813	2814	2815	2816	2818	2819	2820	2921	2822	2823	2839	2840	2841	2842	2843
	2844	2855	2856	2858	2859	2860	2877	2878	2881	2900	2901	2902	2903	2904	2905
	2906	2907	2970	2972	2996	3002	3011	3016	3021	3023	3027	3062	3070	3071	3072

	3078	3085	3103	3104	3105	3106	3107	3120	3121	3123	3124	3125	3126	3127	3129
	3130	3131	3132	3139	3140	3145	3147	3148	3149	3150	3151	3152	3153	3154	3157
	3184	3185	3187	3190	3204	3205	3207	3208	3209	3232	3233	3261	3262	3277	3278
	3279	3290	3307	3309	3310	3311	3312	3314	3315	3330	3342	3344	3345	3346	3379
	3380	3384	3399	3442	3443	3447									
MOV8	856	931	932	933	946	955	960	971	978	981	982	593	1006	1007	1014
	1643	1550	1652	1654	1663	2362	2920	2974	3063	3064	3067	3068	3069	3073	3076
	3077	3096	3236	3242	3266	3271	3320	3321	3381	3409	3417	3445			
NEG	3074	3341													
NOP	571	578	806	1210	1245	1281	1314	1449	1478	2487	2621				
RESET	703	1572	1575	1576	1577	1578	2694								
RETURN	1603	1913	1934	1939	1988	2101									
ROL	647	652	658	662	666	872	1155	2402	2403	2404	2846	3080	3082	3083	3084
	3086														
ROR	949	950	951	952	957	963	964	965	966	967	968	973	974	975	1292
	1639	1718	1719	1720	1721	1930	1931	1932	1933	1999	2003				
RTI	485	630	2807	2824	2984	3108	3159	3210	3252	3281	3347	3386			
RTS	984	1517	1609	1614	1660	1722	1730	2633	2861	2882	2926	3025	3426	3448	
SBC	2910														
SEV	631														
SUB	529	533	900	1493	1671	1790	1794	1886	1890	1978	1979	2008	2052	2349	2420
	2440	2450	2488	2639	2652	2909	2911	2973	3335						
SWAB	891	947	961												
TRAP	3451	3462	3465	3468	3473	3476	3479	3482	3485	3489	3492	3495	3498	3501	3504
	3507	3510	3513	3517	3520	3523	3526	3529							
TST	610	622	624	639	672	691	715	775	798	807	815	816	821	823	928
	837	876	885	887	913	943	995	1054	1078	1104	1256	1259	1313	1321	1419
	1474	1597	1621	1625	1641	1665	1727	1757	1770	1781	1801	1809	1823	1843	1846
	1855	1893	1924	1928	1966	1982	2005	2228	2239	2252	2263	2276	2288	2300	2311
	2320	2331	2410	2414	2436	2458	2462	2510	2531	2546	2568	2583	2620	2632	2651
	2751	2980	3029	3091	3186	3199	3339	3349	3383	3391	3413	3444			
TSTB	849	1002	1004	1235	1315	1333	2360	2367	2590	2592	3193	3234	3240	3375	3415
XOR	632														
.ASCII	188	189	3536												
.ASCIZ	190	3033	3285	3286	3287	3289	3536								
.BLKB	2947	3284													
.BYTE	159	160	165	166	174	175	183	184	185	186	1690	1683	1686	1689	1692
	1695	1698	1701	1704	1707	1710	1713	2132	3109	3110	3111	3112	3282	3283	3537
.DSABL	3220														
.ENABL	1	2	3218												
.END	3544														
.ENDC	14	33	42	134	148	153	157	159	187	188	193	453	461	462	465
	467	469	471	472	473	490	2781	2830	2869	2889	2958	2961	2968	2972	2977
	2978	2979	2980	2984	2985	2990	3006	3035	3040	3119	3129	3131	3138	3144	3146
	3157	3159	3163	3169	3172	3175	3179	3181	3192	3193	3195	3197	3201	3206	3207
	3208	3211	3212	3218	3219	3220	3225	3253	3255	3262	3264	3267	3269	3285	3291
	3296	3361	3381	3437	3443	3446	3460	3461	3463	3466	3469	3470	3471	3472	3474
	3475	3477	3478	3480	3481	3483	3486	3487	3490	3493	3496	3499	3502	3505	3508
	3511	3514	3518	3521	3524	3527	3530								
.EQUIV	42	43	51	66	67	96	97	98	99	100	101	102	103	104	105
	124	125	126	127	128	129	130	131	132	133					
.EVEN	1716	3034	3536												
.IF	10	31	40	106	134	152	156	158	187	188	192	193	453	456	461
	463	465	467	469	471	472	490	2780	2829	2868	2888	2957	2960	2968	2971
	2975	2977	2978	2980	2983	2984	2985	2989	3005	3021	3039	3118	3129	3131	3137
	3144	3146	3155	3157	3159	3163	3168	3171	3175	3179	3191	3193	3194	3195	3197

	3199	3207	3208	3210	3211	3212	3217	3219	3220	3224	3225	3253	3254	3262	3263
	3267	3268	3284	3285	3291	3295	3360	3381	3436	3442	3446	3450	3452	3461	3463
	3466	3469	3470	3471	3472	3474	3475	3477	3478	3480	3481	3483	3486	3487	3490
.IFF	3493	3496	3499	3502	3505	3508	3511	3514	3518	3521	3524	3527	3530		
	40	153	156	158	197	193	461	2781	2830	2869	2889	2958	2960	2972	2983
	2984	2985	2990	3006	3035	3040	3119	3138	3155	3169	3191	3193	3195	3211	3212
.IFT	3218	3220	3225	3227	3232	3253	3255	3264	3268	3285	3296	3361	3437	3443	
.IFTF	2978	3197	3227	3232											
.IIF	2977	3195	3220	3225	3228										
	9	14	19	20	28	191	462	465	471	472	473	582	1592	2013	2020
	2027	2041	2048	2637	2649	2961	2962	2963	2964	2968	2983	2984	2985	3003	3029
	3172	3173	3174	3175	3179	3196	3208	3211	3212	3218	3277	3285	3291	3355	3428
	3460	3463	3466	3469	3474	3477	3480	3483	3486	3490	3493	3496	3499	3502	3505
.IRP	3508	3511	3514	3518	3521	3524	3527	3530							
	453	2797	2818	2839	2858	3123	3130	3145	3147	3310	3344				
.LIST	1	2	28	148	187	453	473	2984	3175	3253	3450	3460	3461	3463	3464
	3466	3467	3469	3470	3474	3475	3477	3478	3480	3481	3483	3484	3486	3487	3490
	3491	3493	3494	3496	3497	3499	3500	3502	3503	3505	3506	3508	3509	3511	3512
	3514	3515	3518	3519	3521	3522	3524	3525	3527	3528	3530	3531	3536		
.MACRO	1	149	3450												
.MCALL	3	4	5	148	473										
.NLIST	1	2	28	148	187	453	473	2984	3175	3253	3450	3460	3461	3463	3464
	3466	3467	3469	3470	3474	3475	3477	3478	3480	3481	3483	3484	3486	3487	3490
	3491	3493	3494	3496	3497	3499	3500	3502	3503	3505	3506	3508	3509	3511	3512
	3514	3515	3518	3519	3521	3522	3524	3525	3527	3528	3530	3531	3536		
.PAGE	149	193													
.REPT	28	2151	2168	2185	2207										
.SBTTL	22	31	38	149	193	455	2777	2826	2865	2885	2954	2986	3036	3115	3165
.TITLE	3214	3292	3357	3433	3452										
.WORD	9														
	28	29	30	158	161	162	163	164	167	168	169	170	171	172	173
	176	177	178	388	389	391	392	394	2151	2152	2153	2154	2155	2156	2157
	2158	2159	2160	2161	2162	2163	2164	2165	2168	2169	2170	2171	2172	2173	2174
	2175	2176	2177	2178	2179	2180	2181	2182	2185	2186	2187	2188	2189	2190	2191
	2192	2193	2194	2195	2196	2197	2198	2199	2207	2208	2209	2210	2211	2212	2213
	2214	2215	2216	2217	2218	2219	2220	2221	2643	2862	2863	2883	2922	3014	3019
	3113	3156	3158	3352	3425										

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

*.DZICAB.SEQ/SOL/CRF=SYSMAC.SML,DZICAB.P11
 RUN-TIME: 54 46 6 SECONDS
 RUN-TIME RATIO: 314/107=2.9
 CORE USED: 34K (67 PAGES)

