

AD11K

PERFORMANCE TEST
MD-11-DZADL-A

EP-DZADL-A-DL-A
COPYRIGHT © 1976
FICHE 1 OF 1

NOV 1976
digital
MADE IN USA

This image shows a microfiche card. The left side contains a grid of frames, each containing a small image or document page. The right side is a large, mostly blank area, possibly representing a larger document page that is mostly obscured or blank. The card is oriented vertically.

.REM %

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZADL-A
 PRODUCT NAME: AD11K PERFORMANCE TEST
 DATE: AUGUST 21, 1976
 MAINTAINER: DIAGNOSTIC GROUP

COPYRIGHT (C) 1976

DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE IN EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

MAC11: 27.732) 25-SEP-76 10:38 PAGE 2

MAC11: 27.732) 25-SEP-76 10:38 PAGE 2

1.0 ABSTRACT

THIS DIAGNOSTIC HAS TWO STARTING ADDRESSES: 200 FOR STANDARD TOLERANCES AND 210 FOR TIGHTER OPTION TEST AREA TOLERANCES.

THIS DIAGNOSTIC TESTS THE AD11K WITH OR WITHOUT A WRAPAROUND MODULE (G5036).

WHEN STARTING THE DIAGNOSTIC, A SET OF TESTS IS LISTED AND THIS STATEMENT IS PRINTED OUT: "TYPE THE LETTER AND CARRIAGE RETURN OF THE DESIRED TEST:". THE FOLLOWING CHART INDICATES WHICH LETTER CORRESPONDS TO WHICH TEST:

- W: THE ENTIRE WRAPAROUND TEST (REQUIRES G5036 MODULE)
 - A. ANALOG SUBTESTS
 - B. NOISE TEST
 - C. INTERCHANNEL SETTLING TEST
 - D. DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST
- C: CALIBRATION TEST ONLY
- N: NOISE TEST ONLY
- S: INTERCHANNEL SETTLING ONLY
- L: LOGIC SUBTESTS ONLY
- A: AUTO TEST (REQUIRES G5036 MODULE)
 - A. LOGIC SUBTESTS
 - B. ANALOG SUBTESTS
 - C. NOISE TEST
 - D. INTERCHANNEL SETTLING TEST
 - E. DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 FAMILY COMPUTER WITH 8K OF MEMORY
TELETYPE
AD11K MODULE
VT55 TERMINAL SUPPORTED FOR GRAPHIC OUTPUT
G5036 WRAPAROUND MODULE

100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200

107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162

2.2 STORAGE

THIS PROGRAM USES ALL BK OF MEMORY AND IS NOT "CHAINABLE" ON AN BK CPU. THE PROGRAM IS "CHAINABLE" ON 12K OR GREATER. THE PROGRAM WILL DESTROY "ABSOLUTE LOADER" ON AN BK CPU, IF "W" OR "A" IS SELECTED.

3.0 LOADING PROCEDURE

PROCEDURE FOR LOADING NORMAL BINARY TAPES SHOULD BE FOLLOWED.

4.0 STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

STANDARD PDP-11 FORMAT

SW15=1 HALT ON ERROR
SW14=1 LOOP ON TEST
SW13=1 INHIBIT ERROR TYPEOUTS
SW12=1 HALT FOR VT55 DISPLAY
SW11=1 INHIBIT ITERATIONS
SW10=1 BELL ON ERROR
SW9 =1 LOOP ON ERROR
SW8 =1 LOOP ON TEST IN SWR <7:0>

200 IS THE STARTING ADDRESS OF THE DIAGNOSTIC FOR STANDARD TOLERANCES. 204 IS THE RESTART ADDRESS. 210 IS THE STARTING ADDRESS OF THE DIAGNOSTIC FOR THE OPTION TEST AREA'S TIGHTER TOLERANCES.

5.0 OPERATING PROCEDURE

START THE DIAGNOSTIC AT 200 OR 210. THE PROGRAM HEADING AND THE LIST OF TESTS AVAILABLE, WILL BE PRINTED OUT FOLLOWED BY A MESSAGE "TYPE THE LETTER AND CARRIAGE RETURN FOR THE DESIRED TEST:". THEN TYPE THE LETTER YOU WANT, ACCORDING TO THE TABLE LISTED AND HIT CARRIAGE RETURN.

163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203

TWO CONTROL CHARACTERS, ↑A AND ↑C, ARE SET ASIDE FOR INTERRUPTING A TEST AND TRANSFERRING CONTROL TO EITHER THE BEGINNING OF THE DIAGNOSTIC (↑C) OR TO THE BEGINNING OF THE SPECIFIC TEST WHICH WAS IN PROGRESS (↑A). DURING THE LOGIC TESTS WHILE A RESET IS BEING PERFORMED, ↑C OR ↑A WILL NOT BE EXECUTED UNTIL AFTER THE RESET HAS BEEN COMPLETED, THEREFORE HIT ↑C OR ↑A UNTIL IT IS SUCCESSFUL.

FOR MACHINES WITHOUT A HARDWARE SWITCH REGISTER, LOCATION SWREG (176) IS USED AS A SOFTWARE SWITCH REGISTER. TO MODIFY THE CONTENTS OF SWREG, TYPE ↑G. THE PROGRAM RESPONDS WITH THE CURRENT CONTENTS OF SWREG AND A SLASH. TYPE THE DESIRED NEW CONTENTS OF SWREG FOLLOWED BY A CARRIAGE RETURN.

IF "W" IS TYPED, THE PROGRAM WILL TYPE "XX AD11K'S FOUND". WHERE XX IS THE NUMBER OF AD11K'S IN OCTAL. IF THE NUMBER IS GREATER THAN 1, THE TEST WILL BE RUN SUCCESSIVELY ON EACH AD11K. THE PROGRAM WILL RUN THROUGH THE LOGIC SUBTESTS, THE NOISE TEST ON B EDGES, THE INTERCHANNEL SETTLING TEST ON B EDGES, AND THE DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST. A G5036 WRAPAROUND MODULE IS REQUIRED. THE PROGRAM SUPPORTS AD11K EXPANSION BEYOND 16 CHANNELS. TO RUN THIS TEST ON A GROUP OF CHANNELS OTHER THAN 0-17, LOAD 20, 40, OR 60 INTO LOCATION BASECH (1336) FOR CHANNELS 20-37, 40-57, 60-77.

IF "C" IS TYPED, THE PROGRAM WILL RUN THE CALIBRATION TEST AND WILL LOOP ON THAT TEST UNTIL THE OPERATOR HALTS IT. IF A CERTAIN AD11K IS TO BE TESTED, ITS STATUS REGISTER ADDRESS MUST BE LOADED INTO \$BASE (1250), AND ITS VECTOR ADDRESS MUST BE LOADED INTO THE LOW BYTE OF \$VECT1 (1244) (THE HIGH BYTE CONTAINING THE PRIORITY).

IF "N" IS TYPED, THE PROGRAM WILL RUN THE NOISE TEST TAGGED "BEGINN" AND WILL LOOP ON THIS TEST UNTIL THE OPERATOR HALTS IT. IF A CERTAIN AD11K IS TO BE TESTED ITS STATUS REGISTER ADDRESS MUST BE LOADED INTO \$BASE (1250), AND ITS VECTOR ADDRESS MUST BE LOADED INTO THE LOW BYTE OF \$VECT1 (1244) (THE HIGH BYTE CONTAINING THE PRIORITY).

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

IF "S" IS TYPED, THE PROGRAM WILL RUN THE INTERCHANNEL SETTling TEST TAGGED "BEGINS" AND WILL LOOP ON THIS TEST UNTIL THE OPERATOR HALTS IT. AT THE BEGINNING OF THIS TEST, THE OPERATOR MUST RESPOND TO THE STATEMENTS ASKING FOR THE "FROM" CHANNEL AND THE "TO" CHANNEL BY TYPING IN THE CHANNEL VALUE IN OCTAL AND HITTING CARRIAGE RETURN. IF A CERTAIN AD11K IS TO BE TESTED ITS STATUS REGISTER ADDRESS MUST BE LOADED INTO \$BASE (1250), AND ITS VECTOR ADDRESS MUST BE LOADED INTO \$VECT1 (1244) (THE HIGH BYTE CONTAINING THE PRIORITY).

IF "A" IS TYPED, THE PROGRAM WILL EXECUTE THE LOGIC TESTS, ANALOG TESTS, NOISE, SETTLE AND DIFFERENTIAL LINEARITY. AT THE BEGINNING OF THE TEST THE PROGRAM WILL TYPE "XX ADV11'S FOUND". WHERE XX IS THE NUMBER OF ADV11'S IN OCTAL IF THE NUMBER IS GREATER THAN 1, THE TEST WILL BE RUN SUCCESSIVELY ON EACH AD11K. THE PROGRAM SUPPORTS AD11K EXPANSION BEYOND 16 CHANNELS. TO RUN THIS TEST ON A GROUP OF CHANNELS OTHER THAN 3-17, LOAD 20,40, OR 60 INTO LOCATION BASECH (1336) FOR CHANNELS 20-37, 40-57, 60-77.

IF "L" IS TYPED, THE PROGRAM WILL EXECUTE THE LOGIC TESTS, PRINTING "END PASS" WHEN IT HAS COMPLETED AN ENTIRE PASS. AT THE BEGINNING OF THE TEST THE PROGRAM WILL TYPE "XX ADV11'S FOUND". WHERE XX IS THE NUMBER OF ADV11'S IN OCTAL IF THE NUMBER IS GREATER THAN 1, THE TEST WILL BE RUN SUCCESSIVELY ON EACH AD11K.

232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
2876.0 ERRORS

THIS PROGRAM USES THE DIAGNOSTIC "SYSMAC" PACKAGE FOR ERROR REPORTING AND TYPEOUT. THE ERROR INFORMATION CONSISTS OF THE FOLLOWING:

ERRPC: LOCATION AT WHICH AN ERROR WAS DETECTED.
 STREG: ADDRESS OF THE STATUS REGISTER.
 ADBUFF: ADDRESS OF THE BUFFER
 CHANL: CHANNEL VALUE
 NOMINAL: EXPECTED CORRECT DATA
 TOLERANCE: THE ACCEPTABLE DEVIATION FROM THE NOMINAL
 ACTUAL: ACTUAL DATA
 EXPECTED: EXPECTED CORRECT DATA

7.0 MISCELLANEOUS

7.1 EXECUTION TIME

EXECUTION TIME FOR EACH OF THE TESTS IS:

CALIBRATION: 8 CONVERSIONS/5 SECONDS @ 110 BAUD
 WRAPAROUND TEST: 17 MINUTES FIRST PASS; 35 MINUTES
 FOR SUCCESSIVE PASSES
 SETTling TEST: 1 MINUTE
 NOISE TEST: 1 MINUTE
 LOGIC TEST: 1 MINUTE
 AUTO TEST: 18 MINUTES FIRST PASS, 36 MINUTES
 FOR SUCCESSIVE PASSES

7.2 STATUS REGISTER AND VECTOR ADDRESSES AND PRIORITY

WHEN TESTING MORE THAN ONE AD11K, THE DIFFERENCE IN ADDRESSES IS PRESENTLY 40 FOR BUS ADDRESS AND VECTOR ADDRESS. THESE VALUES ARE IN VADR (BUS ADDRESS) (1332) AND VVCT (VECTOR ADDRESS) (1334). THE FIRST AD11K'S STATUS REGISTER ADDRESS MUST BE IN \$BASE (1250), ITS VECTOR ADDRESS MUST BE IN THE LOW BYTE OF \$VECT1 (1244), AND THE PRIORITY MUST BE IN THE HIGH BYTE OF \$VECT1.

7.3 AD11K PRIORITY

IF AD11K IS SET FOR A PRIORITY OTHER THAN 6, THE HIGH BYTE OF \$VECT1 (1244) MUST BE ADJUSTED ACCORDINGLY (THE LOW BYTE CONTAINING THE VECTOR ADDRESS). IF MORE THAN ONE AD11K IS BEING TESTED, ALL MUST BE SET AT THE SAME PRIORITY.

288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342

7.4 SWITCH REGISTER

IF A HARDWARE SWITCH REGISTER IS PRESENT AND THE OPERATOR DESIRES TO USE A SOFTWARE SWITCH REGISTER AND THE IG FEATURE; IT IS NECESSARY TO LOAD THE STARTING ADDRESS, SET THE HARDWARE SWITCH REGISTER TO ALL ONES (-1), AND HIT START. THE PROGRAM WILL THEN RUN WITH THE SOFTWARE SWITCH REGISTER.

7.5 VTSS GRAPHIC OUTPUT

THE SCREEN DISPLAY MAY BE HALTED FOR EXAMINATION BY SETTING BIT 12. AND THEN JUST HIT CONTINUE TO COMPLETE THE PROGRAM'S EXECUTION.

8.0 RESTRICTIONS

8.1 A G5036 WRAPAROUND MODULE MUST BE PRESENT WHEN RUNNING THE AUTO TEST AND THE WRAPAROUND TEST.

SWITCH ON G5036 MUST BE IN '0' POSITION.

THE WRAPAROUND (G5036) MODULE MUST BE CONNECTED AS FOLLOWS:
AD11K TO BCD8R CONNECTION A-A, VV-VV
BCD8R TO G5036 CONNECTION "UPSIDE-DOWN" A-VV, VV-A

9.0 PROGRAM DESCRIPTION

9.1 LOGIC TESTS

THESE 14 LOGIC SUBTESTS RUN SEQUENTIALLY WITHOUT FURTHER OPERATOR INTERVENTION AFTER HE/SHE HAS TYPED IN THE NUMBER OF AD11K'S TO BE TESTED. ITS PURPOSE IS TO CHECK THAT EACH OF THE MUX BITS CAN BE LOADED AND PROPERLY READ BACK; THAT INITIALIZE CLEARS THE EXTERNAL START ENABLE BIT, THE DONE BIT, THE INTERRUPT ENABLE BIT, THE OVERFLOW BIT, THE ERROR FLAG, AND THE A/D START BIT. IT ALSO CHECKS THAT THE A/D DONE FLAG SETS AT END OF CONVERSION AND CLEARS WHEN THE CONVERTED VALUE IS READ. IT CHECKS THE INTERRUPT LOGIC AND THE CORRECT SETTING OF THE ERROR FLAG.

343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398

9.2 CALIBRATION TEST

THIS TEST BEGINS WHEN THE OPERATOR TYPES "C". IT THEN LOADS THE CHANNEL FROM THE SWITCH REGISTER BITS 0-7 AND DOES A CONVERSION ON THAT CHANNEL. IF SWR BIT 13 IS DOWN, IT PRINTS OUT THE CONVERTED VALUE ON THE TELETYPE; OTHERWISE, IF SWR BIT 13 IS UP, IT PUTS THE CONVERTED VALUE IN THE DISPLAY REGISTER. THE OPERATOR MAY CHANGE THE CHANNEL AT ANY TIME DURING THE TEST, HOWEVER THE NEW VALUES FROM THE NEW CHANNEL WILL NOT BE PRINTED UNTIL THE NEXT LINE OF 8 VALUES IS PRINTED. THE 8 VALUES ON EACH LINE CORRESPOND TO ONLY ONE CHANNEL.

9.3 DIFFERENTIAL LINEARITY

THIS TEST IS TO DETERMINE IF A CHANGE IN THE INPUT VOLTAGE REPRESENTS A SIMILAR CHANGE IN THE RESULTING CONVERTED BINARY VALUE.

9.4 SETTling TEST

THE PURPOSE OF THIS TEST IS TO CHECK THAT THE TIME NEEDED TO SETTLE AND CORRECTLY REPORT A NEW INPUT VALUE AFTER SWITCHING CHANNELS DOES NOT EXCEED THE EXPECTED AMOUNT OF TIME FOR SUCH A CHANGE.

9.5 NOISE TEST

THIS TEST MEASURES THE INTERNAL SHORT-TERM REPEATABILITY NOISE WITHIN THE A/D. RMS NOISE EQUALS 1 STANDARD DEVIATION OF THE GAUSSIAN CURVE, PEAK NOISE EQUALS 2.3 STANDARD DEVIATION OF THE GAUSSIAN CURVE.

9.6 ANALOG TESTS

THESE 11 SUBTESTS CHECK THE CHANNELS AND THEIR OUTPUT.

%.
:TITLE MAINDEC-11-DZADL-A
:*COPYRIGHT (C) 1976
:*DIGITAL EQUIPMENT CORP.
:*MAYNARD, MASS. 01754
:*
:*PROGRAM BY VERA BREUER
:*
:*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
:*PACKAGE (MAINDEC-11-DZQAC-CO), MAR 21, 1976.

```

399
400
401
402
403      OC1100
404
405
406
407
408      000011
409      000012
410      000015
411      000200
412      177776
413
414      177774
415      177772
416      177570
417      177570
418
419
420      000000
421      000001
422      000002
423      000003
424      000004
425      000005
426      000006
427      000007
428
429
430
431
432      000000
433      000040
434      000100
435      000140
436      000200
437      000240
438      000300
439      000340
440
441
442      100000
443      040000
444      020000
445      010000
446      004000
447      002000
448      001000
449      000400
450      000200
451      000100
452      000040
453      000020
454      000010

;*
.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

;*GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0      ;;GENERAL REGISTER
R1= %1      ;;GENERAL REGISTER
R2= %2      ;;GENERAL REGISTER
R3= %3      ;;GENERAL REGISTER
R4= %4      ;;GENERAL REGISTER
R5= %5      ;;GENERAL REGISTER
R6= %6      ;;GENERAL REGISTER
R7= %7      ;;GENERAL REGISTER
.EQUIV R6,SP  ;;STACK POINTER
.EQUIV R7,PC  ;;PROGRAM COUNTER

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

;*SWITCH REGISTER SWITCH DEFINITIONS
SW15= 100000
SW14= 40000
SW13= 20000
SW12= 10000
SW11= 4000
SW10= 2000
SW09= 1000
SW08= 400
SW07= 200
SW06= 100
SW05= 40
SW04= 20
SW03= 10

```

```

455      000004      SW02=      4
456      000002      SW01=      2
457      000001      SW00=      1
458      .EQUIV      SW09,SW9
459      .EQUIV      SW08,SW8
460      .EQUIV      SW07,SW7
461      .EQUIV      SW06,SW6
462      .EQUIV      SW05,SW5
463      .EQUIV      SW04,SW4
464      .EQUIV      SW03,SW3
465      .EQUIV      SW02,SW2
466      .EQUIV      SW01,SW1
467      .EQUIV      SW00,SW0
468
469      .:DATA BIT DEFINITIONS (BIT00 TO BIT15)
470      100000      BIT15=     100000
471      040000      BIT14=     40000
472      020000      BIT13=     20000
473      010000      BIT12=     10000
474      004000      BIT11=     4000
475      002000      BIT10=     2000
476      001000      BIT09=     1000
477      000400      BIT08=     400
478      000200      BIT07=     200
479      000100      BIT06=     100
480      000040      BIT05=     40
481      000020      BIT04=     20
482      000010      BIT03=     10
483      000004      BIT02=     4
484      000002      BIT01=     2
485      000001      BIT00=     1
486      .EQUIV      BIT09,BIT9
487      .EQUIV      BIT08,BIT8
488      .EQUIV      BIT07,BIT7
489      .EQUIV      BIT06,BIT6
490      .EQUIV      BIT05,BIT5
491      .EQUIV      BIT04,BIT4
492      .EQUIV      BIT03,BIT3
493      .EQUIV      BIT02,BIT2
494      .EQUIV      BIT01,BIT1
495      .EQUIV      BIT00,BIT0
496
497      .:BASIC "CPU" TRAP VECTOR ADDRESSES
498      000004      ERRVEC= 4      ;; TIME OUT AND OTHER ERRORS
499      000010      RESVEC= 10     ;; RESERVED AND ILLEGAL INSTRUCTIONS
500      000014      TBITVEC=14     ;; "T" BIT
501      000014      TRTVEC= 14     ;; TRACE TRAP
502      000014      BPTVEC= 14     ;; BREAKPOINT TRAP (BPT)
503      000020      IOTVEC= 20     ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
504      000024      PWRVEC= 24     ;; POWER FAIL
505      000030      EMTVEC= 30     ;; EMULATOR TRAP (EMT) **ERROR**
506      000034      TRAPVEC=34     ;; "TRAP" TRAP
507      000060      TKVEC= 60      ;; TTY KEYBOARD VECTOR
508      000064      TPVEC= 64      ;; TTY PRINTER VECTOR
509      000240      PIRQVEC=240    ;; PROGRAM INTERRUPT REQUEST VECTOR
510      .SBTTL      OPERATIONAL SWITCH SETTINGS

```

511				.*			
512				.*	SWITCH	USE	
513				.*	-----	-----	
514				.*	15	HALT ON ERROR	
515				.*	14	LOOP ON TEST	
516				.*	13	INHIBIT ERROR TYPEOUTS	
517				.*	12	HALT FOR VTSS DISPLAY	
518				.*	11	INHIBIT ITERATIONS	
519				.*	10	BELL ON ERROR	
520				.*	9	LOOP ON ERROR	
521				.*	8	LOOP ON TEST IN SWR(7:0)	
522	170400				ABASE= 170400		
523	140340				AVECT1= 140340		
524	000300				APRIOR= 300		
525							
526					.SBTTL TRAP CATCHER		
527							
528	000000				.=0		
529					;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"		
530					;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS		
531					;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS		
532	000174				.=174		
533	000174	000000			DISPREG: .WORD 0	:: SOFTWARE DISPLAY REGISTER	
534	000176	000000			SWREG: .WORD 0	:: SOFTWARE SWITCH REGISTER	
535					.SBTTL STARTING ADDRESS(ES)		
536	000200	000137	001620		JMP @#BEGIN	:: JUMP TO STARTING ADDRESS OF PROGRAM	
537	000204	000137	002256		JMP @#BEG2	:: RESTART ADDRESS	
538	000210	000137	001626		JMP @#BEGIN2	:: START ADDRESS FOR OPTION TEST AREA	

539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571

.SBTTL ACT11 HOOKS

HOOKS REQUIRED BY ACT11

000214
000046
011764
000052
000000
000214
001000

\$SVPC= ;SAVE PC
.=46
\$ENDAD ;;1)SET LOC.46 TO ADDRESS OF \$ENDAD IN .SECP
.=52
.WORD 0 ;;2)SET LOC.52 TO ZERO
.= \$SVPC ;; RESTORE PC

.=1000
.SBTTL APT PARAMETER BLOCK

SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT

001000
000024
000200
000044
001000
001000

.\$X= ;;SAVE CURRENT LOCATION
.=24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
200 ;;FOR APT START UP
.=44 ;;POINT TO APT INDIRECT ADDRESS PNTR.
\$APTHDR ;;POINT TO APT HEADER BLOCK
.=.\$X ;;RESET LOCATION COUNTER

SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
INTERFACE SPEC.

\$APTHD:
\$HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
\$MBADR: .WORD \$MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
\$STMT: .WORD 1200. ;;RUN TIM OF LONGEST TEST
\$PASTM: .WORD 500. ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
\$UNITM: .WORD 1700. ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
\$ETEND-\$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

572
573
574
575
576
577
578
579 001100
580 001100 000000
581 001102 000
582 001103 000
583 001104 000000
584 001106 000000
585 001110 000000
586 001112 000000
587 001114 000
588 001115 001
589 001116 000000
590 001120 000000
591 001122 000000
592 001124 000000
593 001126 000000
594 001130 000000
595 001132 000000
596 001134 000
597 001135 000
598 001136 000000
599 001140 177570
600 001142 177570
601 001144 177560
602 001146 177562
603 001150 177564
604 001152 177566
605 001154 000
606 001155 002
607 001156 012
608 001157 000
609 001160 000000
610 001162 000000
611 001164 177607 000377
612 001170 077
613 001171 015
614 001172 000012
615
616
617
618
619
620 001174
621 001174 000000
622 001176 000000
623 001200 000000
624 001202 000000
625 001204 000000
626 001206 000000
627 001210 000000

.SBTTL COMMON TAGS

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

001100
SCMTAG: =1100
;; START OF COMMON TAGS
\$TSTNM: .WORD 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
\$GDDAT: .WORD 0 ;; CONTAINS 'GOOD' DATA
\$BDDAT: .WORD 0 ;; CONTAINS 'BAD' DATA
;; RESERVED--NOT TO BE USED
\$AUTOB: .WORD 0 ;; AUTOMATIC MODE INDICATOR
\$INTAG: .BYTE 0 ;; INTERRUPT MODE INDICATOR
\$SWR: .WORD 0 DSWR ;; ADDRESS OF SWITCH REGISTER
\$DISPLAY: .WORD 0 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"
\$STPFLG: .BYTE 0 ;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
\$TIMES: 0 ;; MAX. NUMBER OF ITERATIONS
\$ESCAPE: 0 ;; ESCAPE ON ERROR ADDRESS
\$BELL: .ASCIZ <207><377><377> ;; CODE FOR BELL
\$QUES: .ASCII /?/ ;; QUESTION MARK
\$CRLF: .ASCII <15> ;; CARRIAGE RETURN
\$LF: .ASCIZ <12> ;; LINE FEED

.SBTTL APT MAILBOX-ETABLE

.EVEN
\$MAIL: .WORD 0 ;; APT MAILBOX
\$MSGTY: .WORD 0 AMSGTY ;; MESSAGE TYPE CODE
\$FATAL: .WORD 0 AFATAL ;; FATAL ERROR NUMBER
\$TESTN: .WORD 0 ATESTN ;; TEST NUMBER
\$PASS: .WORD 0 APASS ;; PASS COUNT
\$DEVCT: .WORD 0 ADEVCT ;; DEVICE COUNT
\$UNIT: .WORD 0 AUNIT ;; I/O UNIT NUMBER
\$MSGAD: .WORD 0 AMSGAD ;; MESSAGE ADDRESS

628	001212	000000	\$MSGLG: .WORD	AMSGLG	:: MESSAGE LENGTH
629	001214		\$ETABLE:		:: APT ENVIRONMENT TABLE
630	001214	000	\$ENV: .BYTE	AENV	:: ENVIRONMENT BYTE
631	001215	000	\$ENVM: .BYTE	AENVM	:: ENVIRONMENT MODE BITS
632	001216	000000	\$SWREG: .WORD	ASWREG	:: APT SWITCH REGISTER
633	001220	000000	\$USWR: .WORD	AUSWR	:: USER SWITCHES
634	001222	000000	\$CPUOP: .WORD	ACPUOP	:: CPU TYPE, OPTIONS
635			*		BITS 15-11=CPU TYPE
636			*		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
637			*		11/70=06, PDQ=07, Q=10
638			*		BIT 10=REAL TIME CLOCK
639			*		BIT 9=FLOATING POINT PROCESSOR
640			*		BIT 8=MEMORY MANAGEMENT
641	001224	000	\$MAMS1: .BYTE	AMAMS1	:: HIGH ADDRESS, M.S. BYTE
642	001225	000	\$MTYP1: .BYTE	AMTYP1	:: MEM. TYPE, BLK#1
643			*		MEM. TYPE BYTE -- (HIGH BYTE)
644			*		900 NSEC CORE=001
645			*		300 NSEC BIPOLAR=002
646			*		500 NSEC MOS=003
647	001226	000000	\$MADR1: .WORD	AMADR1	:: HIGH ADDRESS, BLK#1
648			*		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
649	001230	000	\$MAMS2: .BYTE	AMAMS2	:: HIGH ADDRESS, M.S. BYTE
650	001231	000	\$MTYP2: .BYTE	AMTYP2	:: MEM. TYPE, BLK#2
651	001232	000000	\$MADR2: .WORD	AMADR2	:: MEM. LAST ADDRESS, BLK#2
652	001234	000	\$MAMS3: .BYTE	AMAMS3	:: HIGH ADDRESS, M.S. BYTE
653	001235	000	\$MTYP3: .BYTE	AMTYP3	:: MEM. TYPE, BLK#3
654	001236	000000	\$MADR3: .WORD	AMADR3	:: MEM. LAST ADDRESS, BLK#3
655	001240	000	\$MAMS4: .BYTE	AMAMS4	:: HIGH ADDRESS, M.S. BYTE
656	001241	000	\$MTYP4: .BYTE	AMTYP4	:: MEM. TYPE, BLK#4
657	001242	000000	\$MADR4: .WORD	AMADR4	:: MEM. LAST ADDRESS, BLK#4
658	001244	140340	\$VECT1: .WORD	AVECT1	:: INTERRUPT VECTOR#1, BUS PRIORITY#1
659	001246	000000	\$VECT2: .WORD	AVECT2	:: INTERRUPT VECTOR#2, BUS PRIORITY#2
660	001250	170400	\$BASE: .WORD	ABASE	:: BASE ADDRESS OF EQUIPMENT UNDER TEST
661	001252	000000	\$DEVN: .WORD	ADEVN	:: DEVICE MAP
662	001254	000000	\$CDW1: .WORD	ACDW1	:: CONTROLLER DESCRIPTION WORD#1
663	001256		\$ETEND:		
664			.MEXIT		

665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708

.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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
:*NOTE1: IF SITEMB 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

001256

\$ERRTS:

:ITEM 1
EM1 :STATUS REG. ERROR
DH1 :ERRPC STREG EXPECTED ACTUAL
DT1 :\$ERRPC, STREG, \$GDDAT, \$BDDAT
DF1

001256 014134
001260 014274
001262 014460
001264 014520

:ITEM 2
EM2 :FAILED TO INTERRUPT
DH3 :ERRPC STREG ACTUAL
DT3 :\$ERRPC, STREG, \$BDDAT
DF1

001266 014162
001270 014415
001272 014510
001274 014520

:ITEM 3
EM3 :UNEXPECTED INTERRUPT
DH3 :ERRPC STREG
DT3 :\$ERRPC, STREG
DF1

001276 014212
001300 014415
001302 014510
001304 014520

:ITEM 4
EM4 :ERROR ON A/D CHANNEL
DH2 :ERRPC STREG CHAN NOMINAL TOL ACTUAL
DT2 :\$ERRPC, STREG, CHANL, \$GDDAT, SPREAD, \$BDDAT
DF1

001306 014243
001310 014332
001312 014472
001314 014520

```

709 .SBTTL MISCELLANEOUS, TEMPORARY, AND STORAGE LOCATIONS
710 001316 170400 STREG: ABASE ;ADDRESS OF STATUS REGISTER
711 001320 170401 POST1: ABASE+1 ;UPPER BYTE OF STATUS REG.
712 001322 170402 ADBUFF: ABASE+2 ;ADDRESS OF A/D BUFFER
713 001324 140340 VECTOR: AVECT1 ;VECTOR ADDRESS
714 001326 000300 BASEBR: APRIOR ;INTERRUPT PRIORITY LEVEL
715 001330 140342 VECTR1: AVECT1+2
716 001332 000040 VADR: 40 ;INCREMENT FOR BUS ADDRESS
717 001334 000040 VVCT: 40 ;INCREMENT FOR VECTOR ADDRESS
718 001336 000000 BASECH: 0 ;BASE CHANNEL
719 001340 000060 KBVECT: 60
720 001342 000000 WIDE: 0 ;NO. OF WIDE STATES
721 001344 000000 NARROW: 0 ;NO. OF NARROW STATES
722 001346 000000 FIRST: 0
723 001350 000000 SKIPST: 0 ;NO. OF SKIPPED STATES
724 001352 000000 TEMP: 0
725 001354 000000 CH1: 0 ;WORK AREA
726 001356 000000 CH2: 0 ;FIRST CHANNEL
727 001360 000000 NBEXT: 0 ;SECOND CHANNEL
728 001362 000000 NMBEXT: 0 ;NO. OF ADLIK'S TO BE TESTED
729 001364 000000 DUMMY: 0 ;NO. OF ADLIK'S TO BE TESTED
730 001366 000000 CHANL: 0 ;DUMMY CHANNEL
731 001370 000000 TAADR: 0 ;CHANNEL VALUE
732 001372 000000 RNA: 0 ;TEST ADDRESS
733 001374 000000 RNB: 0 ;RANDOM
734 001376 000000 RNC: 0 ;NUMBER
735 001400 000000 RMS: 0 ;VALUES
736 001402 000000 PEAK: 0 ;RMS NOISE VALUE
737 001404 000000 FLAG: 0 ;PEAK NOISE VALUE
738 001406 000000 SPREAD: 0 ;VTSS FLAG
739 001410 000000 DAC: 0 ;DEVIATION FROM THE NOMINAL
740 001412 000000 DELAY: 0 ;SAR VALUE
741 001414 000000 EDGE: 0 ;TIME DELAY COUNTER
742 001416 000000 BITPNT: 0 ;EDGE VALUE
743 001420 000000 MIN: 0 ;MIN VALUE
744 001422 000000 WFTST: 0 ;OPTION TEST AREA FLAG
745 001424 000000 MAX: 0 ;MAX VALUE
746 001426 000000 PERCNT: 0 ;PERCENT FOR SAR ROUTINE
747 001430 000000 OUT: 0
748
749 001432 UNEXP:
750 001432 012737 001446 001162 MOV #1$, $ESCAPE ;;ESCAPE TO 1$ ON ERROR
751 001440 005237 001103 INC $ERFLG
752 001444 104003 ERROR 3
753 001446 005037 001162 1$: CLR $ESCAPE ;RETURN ESCAPE TO NORMAL
754 001452 000002 RTI ;UNEXPECTED INTERRUPT

```

755									
756	001454	010046							
757	001456	017700	177464						
759	001462	042700	177600						
759	001466	120027	000003						
760	001472	001010							
761	001474	104400	012134						
762	001500	012706	001100						
763	001504	004737	011254						
764	001510	000137	002256						
765	001514	120027	000001						
766	001520	001010							
767	001522	104400	012127						
768	001526	012706	001100						
769	001532	004737	011254						
770	001536	000177	177626						
771	001542	120027	000007						
772	001546	001021							
773	001550	023727	001140	177570					
774	001556	001415							
775	001560	104400	012141						
776	001564	017746	177350						
777									
778	001570	104402							
779	001572	006							
780	001573	001							
781	001574	104400	012321						
782	001600	104406							
783	001602	012677	177332						
784	001606	012600							
785	001610	000002							
786	001612	104400	012125						
787	001616	000773							

```

SBTTL CONTROL A AND C DECODERS
ISERV: MOV RO, -(SP) ;SAVE RO
MOV @STACK, RO ;GET CHARACTER
BIC #177600, RO
CMPB RO, #3 ;IS IT 'C'?
BNE 1$ ;ECHO CHARACTER
TYPE ,MSG ;ECHO CHARACTER
MOV #STACK, SF ;RESET & SET INTRPT. EN.
JSR PC, RST
JMP BEG2
1$: CMPB RO, #1 ;IS IT 'A'?
BNE 2$ ;ECHO CHARACTER
TYPE ,MSG ;ECHO CHARACTER
MOV #STACK, SF ;RESET & SET INTRPT. EN.
JSR PC, RST ;RETURN TO TEST
JMP @ADDR ;IS IT 'G'?
2$: CMPB RO, #7 ;HARDWARE SWREG?
BNE NONE ;ECHO CHARACTER
CMP SWR, #177570 ;SAVE @SWR FOR TYPEOUT
BEQ NONE ;TYPE SWREG
TYPE ,MSG ;GO TYPE--OCTAL ASCII
MOV @SWR, -(SP) ;TYPE 6 DIGITS
;TYPE LEADING ZEROS
TYP0S ;READ NEW VALUE
.BYTE 6 ;LOAD NEW SWREG VALUE
.BYTE 1
TYPE ,SLASH
RDOCT
MOV (SP)+, @SWR
MOV (SP)+, RO
POP0: MOV
RETURN: RTI
NONE: TYPE ,QUEST ;TYPE ""?
BR POP0
    
```

```

788          .SBTTL      INITIAL START-UP, HOUSEKEEPING, AND DIALOGUE
789 001620 005037 001422 BEGIN: CLR      WFTST
790 001624 000403          BR      RBEG
791 001626 012737 000001 001422 BEGIN2: MOV     #1, WFTST
792 001634 000005          RBEG:  RESET
793          .SBTTL      INITIALIZE THE COMMON TAGS
794          ;; CLEAR THE COMMON TAGS (%CMTAG) AREA
795 001636 012706 001100          MOV     #%CMTAG, R6      ;; FIRST LOCATION TO BE CLEARED
796 001642 005026          CLR     (R6)+          ;; CLEAR MEMORY LOCATION
797 001644 022706 001140          CMP     #SWR, R6      ;; DONE?
798 001650 001374          BNE     -6              ;; LOOP BACK IF NO
799 001652 012706 001100          MOV     #STACK, SP    ;; SETUP THE STACK POINTER
800          ;; INITIALIZE A FEW VECTORS
801 001656 012737 015116 000020 MOV     #$$SCOPE, @IOTVEC ;; IOT VECTOR FOR SCOPE ROUTINE
802 001664 012737 000340 000022 MOV     #340, @IOTVEC+2  ;; LEVEL 7
803 001672 012737 015374 000030 MOV     #ERROR, @EMTVEC ;; EMT VECTOR FOR ERROR ROUTINE
804 001700 012737 000340 000032 MOV     #340, @EMTVEC+2 ;; LEVEL 7
805 001706 012737 016702 000034 MOV     #STRAP, @TRAPVEC ;; TRAP VECTOR FOR TRAP CALLS
806 001714 012737 000340 000036 MOV     #340, @TRAPVEC+2; LEVEL 7
807 001722 012737 016742 000024 MOV     #SPWRDN, @PWRVEC ;; POWER FAILURE VECTOR
808 001730 012737 000340 000026 MOV     #340, @PWRVEC+2 ;; LEVEL 7
809 001736 013737 011744 011736 MOV     SENDCT, SEOPCT  ;; SETUP END-OF-PROGRAM COUNTER
810 001744 005037 001160          CLR     $TIMES          ;; INITIALIZE NUMBER OF ITERATIONS
811 001750 005037 001162          CLR     $ESCAPE        ;; CLEAR THE ESCAPE ON ERROR ADDRESS
812 001754 112737 000001 001115 MOVB   #1, $ERMAX      ;; ALLOW ONE ERROR PER TEST
813 001762 012737 001762 001106 MOV     #., $LPADR     ;; INITIALIZE THE LOOP ADDRESS FOR SCOPE
814 001770 012737 001770 001110 MOV     #., $LPERR     ;; SETUP THE ERROR LOOP ADDRESS
815          ;; SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
816          ;; EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
817 001776 013746 000004          MOV     @ERRVEC, -(SP)  ;; SAVE ERROR VECTOR
818 002002 012737 002036 000004 MOV     #64$, @ERRVEC  ;; SET UP ERROR VECTOR
819 002010 012737 177570 001140 MOV     #DSWR, SWR     ;; SETUP FOR A HARDWARE SWICH REGISTER
820 002016 012737 177570 001142 MOV     #DDISP, DISPLAY ;; AND A HARDWARE DISPLAY REGISTER
821 002024 022777 177777 177106 CMP     #-1, @SWR     ;; TRY TO REFERENCE HARDWARE SWR
822 002032 001012          BNE     66$          ;; BRANCH IF NO TIMEOUT TRAP OCCURRED
823          ;; AND THE HARDWARE SWR IS NOT = -1
824 002034 000403          BR      65$          ;; BRANCH IF NO TIMEOUT
825 002036 012716 002044          64$:  MOV     #65$, (SP)    ;; SET UP FOR TRAP RETURN
826 002042 000002          RTI
827 002044 012737 000176 001140 65$:  MOV     #SWREG, SWR    ;; POINT TO SOFTWARE SWR
828 002052 012737 000174 001142 MOV     #DISPREG, DISPLAY
829 002060 012637 000004          66$:  MOV     (SP)+, @ERRVEC ;; RESTORE ERROR VECTOR
830
831 002064 005037 001202          CLR     $PASS          ;; CLEAR PASS COUNT
832 002070 132737 000200 001215 BITB   #APTSIZE, $ENVM  ;; TEST USER SIZE UNDER APT
833 002076 001403          BEQ     67$          ;; YES, USE NON-APT SWITCH
834 002100 012737 001216 001140 MOV     #$$SWREG, SWR  ;; NO, USE APT SWITCH REGISTER
835 002106          67$:

```

```

836 002106 005037 001404 CLR FLAG ;CLEAR VT55 FLAG
837 002112 005737 000042 TST 2#42 ;IS IT CHAINED?
838 002116 001033 BNE REST1
839 .SBTTL DETERMINE IF VT55 TYPE TERMINAL IS PRESENT
840 002120 042777 000100 177016 BIC #100,25TKS
841 002126 104400 013561 TYPE CO ;TYPE ASCIZ STRING
842 002132 004737 002524 JSR PC,VTFLG ;GET A CHARACTER
843 002136 020027 000033 CMP RO,#33
844 002142 001017 BNE NOVT55 ;NO VT55 PRESENT
845 002144 004737 002524 JSR PC,VTFLG ;GET A CHARACTER
846 002150 020027 000057 CMP RO,#57
847 002154 001012 BNE NOVT55 ;NO VT55 PRESENT
848 002156 004737 002524 JSR PC,VTFLG ;GET A CHARACTER
849 002162 020027 000103 CMP RO,#103
850 002166 001403 BEQ VT55 ;VT55 IS PRESENT
851 002170 020027 000105 CMP RO,#105
852 002174 001002 BNE NOVT55
853 002176 005237 001404 VT55: INC FLAG

```

```

854      .SBTTL      DIALOGUE TO DETERMINE WHICH TEST TO RUN
855 002202 104400 013724  NCVT55: TYPE      ,HEAD1
856 002206 000005  REST1:  RESET
857 002210 004737 005472      JSR      PC, FIXONE      ; INITIALIZE ADDRESSES
858 002214 013700 001340      MOV      KBVECT, RO
859 002220 012720 001454      MOV      #ISERV, (RO)+
960 002224 012710 000340      MOV      #340, (RO)
861 002230 012737 062341 001372  MOV      #62341, RNA      ; RANDOM NO, VARIABLES
862 002236 012737 142315 001374  MOV      #142315, RNB
863 002244 012737 127623 001376  MOV      #127623, RNC
864 002252 004737 011540      JSR      PC, WFADJ      ; STANDARD OR OPTION TEST TOLERANCES?
865 002256 000005  BEG2:  RESET      ; RESTART ADDRESS
866 002260 005737 000042      TST      @#42      ; IS IT CHAINED?
867 002264 001402      BEQ      1$
868 002266 000137 005222      JMP      BEGL      ; GO TO LOGIC TESTS
869 002272 104400 013367  1$:  TYPE      ,MSG71
870 002276 104405  TRYAG: RDLIN
871 002300 052777 000100 176636  BIS      #100, @STKS
872 002306 005037 177776      CLR      PSW
873 002312 012600      MOV      (SP)+, RO      ; READ ANSWER
874 002314 142710 000040      BICB     #40, (RO)
875 002320 121027 000101      CMPB     (RO), #'A
876 002324 001002      BNE
877 002326 000137 005260      JMP      BFGINA      ;;NO, TRY C
878 002332 121027 000103  1$:  CMPB     (RO), #'C      ; GO TO AUTO TEST
879 002336 001002      BNE
880 002340 000137 005036      JMP      BEGINC      ; GO TO CALIBRATION TEST
881 002344 121027 000114  2$:  CMPB     (RO), #'L      ; IS IT L?
882 002350 001002      BNE
883 002352 000137 005222      JMP      BEGL      ; GO TO LOGIC TESTS
884 002356 121027 000116  3$:  CMPB     (RO), #'N      ; IS IT N?
885 002362 001002      BNE
886 002364 000137 005646      JMP      BEGINN      ; GO TO NOISE TEST
887 002370 121027 000123  4$:  CMPB     (RO), #'S      ; IS IT S?
888 002374 001002      BNE
889 002376 000137 005716      JMP      BEGINS      ; GO TO SETTLE TEST
890 002402 121027 000127  5$:  CMPB     (RO), #'W      ; IS IT W?
891 002406 001002      BNE
892 002410 000137 005346      JMP      BEGINW      ; NO, TRY AGAIN
893 002414 104400 012125  6$:  TYPE      QUEST      ; GO TO WRAPAROUND TEST
894 002420 000726      BR       TRYAG      ; WAIT FOR CHARACTER
895 002422 013737 001250 001126  TESTAD: MOV     $BASE, $BDDAT ; SETUP TO TEST FOR AD11K'S
896 002430 013746 000004      MOV     @ERRVEC, -(SP) ; SAVE ERRVEC
897 002434 012737 002466 000004      MOV     #25, ERRVEC   ; SET UP FOR TIME OUT ERROR
898 002442 005037 001360      CLR     NBEXT         ; CLEAR AD11K COUNTER
899 002446 005777 176454  1$:  TST     @BDDAT       ; ADDRESS AD11K
900 002452 005237 001360      INC     NBEXT         ; INCREMENT AD11K COUNTER
901 002456 063737 001332 001126  ADD     VADR, $BDDAT   ; GET NEXT AD11K
902 002464 000770      BR     1$           ;; TRY NEXT AD11K

```

MAINDEC-11-DZADL-A
DZADLA.P11

MACY11 27(732) 25-SEP-76 10:38 PAGE 22
DIALOGUE TO DETERMINE WHICH TEST TO RUN

903	002466	022626		2S:	CMP	(SP)+,(SP)+		;POP 2 WORDS OFF STACK
904	002470	013746	001360		MOV	NBEXT,-(SP)		;SAVE NBEXT FOR TYPECUT
905								;TYPE NUMBER OF AD11K'S
906	002474	104402			TYPOS			;GO TYPE--OCTAL ASCII
907	002476	002			.BYTE	2		;TYPE 2 DIGIT(S)
909	002477	000			.BYTE	0		;SUPPRESS LEADING ZEROS
909	002500	104400	012727		TYPE	MSG50		
910	002504	005337	001360		DEC	NBEXT		;ADJUST AD11K COUNT
911	002510	013737	001360	001362	MOV	NBEXT,NMBEXT		;KEEP COUNT OF NUMBER
912	002516	012637	000004		MOV	(SP)+,ERRVEC		;RESTORE ERRVEC
913	002522	000207			RTS	PC		
914								
915	002524	005000		VTFLG:	CLR	RO		;TEST FOR PRESENCE
916	002526	105777	176412	1S:	TSTB	2STKS		;OF VT55
917	002532	100404			BMI	2S		;VT55 RESPONDS WITH <33><57>[<103> OR <105>]
918	002534	005300			DEC	RO		
919	002536	001373			BNE	1S		
920	002540	005726			TST	(SP)+		;POP A WORD OFF STACK
921	002542	000617			BR	NOVT55		;NO VT55 PRESENT
922	002544	017700	176376	2S:	MOV	2STKB,RO		
923	002550	042700	177600		BIC	#177600,RO		
924	002554	000207			RTS	PC		;TEST VT55 CODE

```

925 002556
926
927
928
929 002556 012737 002556 001106
930 002564 012737 002556 001110
931 002572 012737 000400 001124
932 002600 004737 003654
933 002604 104001
934 002606 006137 001124
935 002612 023727 001124 040000
936 002620 001367
937
938
939
940
941 002622 000004
942 002624 012777 001432 176472
943 002632 012737 000100 001124
944 002640 004737 003654
945 002644 104001
946
947
948
949
950 002646 000004
951 002650 012737 000040 001124
952 002656 004737 003654
953 002662 104001
954
955
956
957
958 002664 000004
959 002666 012737 000020 001124
960 002674 004737 003654
961 002700 104001
962
963
964
965 002702 000004
966 002704 012737 100000 001124
967 002712 004737 003654
968 002716 104001

```

```

BEGINL:
*****
;TEST 1      FLOAT A ONE THRU MULTIPLEXER BITS
*****
TST1:  MOV      #TST1,$LPADR
        MOV      #TST1,$LPERR
        MOV      #BIT8,$GDDAT      ;LOAD FIRST BIT
2$:    JSR      PC,TESTIT
        ERROR    1                  ;FAILED TO LOAD + READ BIT
1$:    ROL      $GDDAT              ;GET NEXT BIT
        CMP     $GDDAT,#BIT14      ;FINISHED?
        BNE     2$                  ;;NO,GO TO NEXT TEST
*****
;TEST 2      LOAD AND READ BACK INTERRUPT ENABLE BIT6
*****
TST2:  SCOPE
        MOV      #UNEXP,$VECTOR    ;SETUP FOR UNEXPECTED INTERRUPT
        MOV      #BIT6,$GDDAT      ;LOAD EXPECTED DATA
        JSR      PC,TESTIT
        ERROR    1                  ;FAILED TO LOAD + READ INTERRUPT ENABLE
*****
;TEST 3      LOAD AND READ BACK CLOCK OVERFLOW START ENABLE BITS
*****
TST3:  SCOPE
        MOV      #BIT5,$GDDAT      ;LOAD EXPECTED DATA
        JSR      PC,TESTIT
        ERROR    1                  ;FAILED TO LOAD + READ CLOCK OVERFLOW START ENAB
*****
;TEST 4      LOAD AND READ BACK EXTERNAL START ENABLE BIT4
*****
TST4:  SCOPE
        MOV      #BIT4,$GDDAT      ;LOAD EXPECTED DATA
        JSR      PC,TESTIT
        ERROR    1                  ;FAILED TO LOAD + READ EXT. START ENABLE
*****
;TEST 5      LOAD AND READ BACK ERROR FLAG BIT15
*****
TST5:  SCOPE
        MOV      #BIT15,$GDDAT     ;LOAD EXPECTED DATA
        JSR      PC,TESTIT
        ERROR    1                  ;FAILED TO LOAD + READ ERROR FLAG

```

```

969      ;*****
970      ;*TEST 6      TEST INIT CLEARS BITS 1,4-6,8-13,15
971      ;*****
972      002720 000004      †ST6:  SCOPE
973      002722 012737 000300 001160      MOV      #300,$TIMES      ;;DO 300 ITERATIONS
974      002730 005037 001124      CLR      $GDDAT      ;LOAD EXPECTED DATA
975      002734 012777 137562 176354 25:  MOV      #137562,@STREG ;SET STATUS REGISTER
976      002742 012737 000010 001352      MOV      #10,TEMP
977      002750 005337 001352      15:  DEC      TEMP      ;DELAY FOR ONE-SHOT
978      002754 001375      BNE     15      ;IN ANALOG PACKAGE
979      002756 000005      RESET    ;INITIALIZE
980      002760 052777 000100 176156      BIS      #100,@$TKS      ;SET INTRPT. ENABLE
981      002766 017737 176324 001126      MOV      @STREG,$BDDAT ;READ STATUS REGISTER
982      002774 001401      BEQ     TST7      ;NEXT TEST
983      002776 104001      ERROR   1      ;RESET FAILED TO CLEAR AD ST. REG. BITS
984
985      ;*****
986      ;*TEST 7      TEST DONE FLAG SETS AND BIT0 CLEARS ON END OF CONV.
987      ;*****
988      003000 000004      †ST7:  SCOPE
989      003002 012700 001000      MOV      #BIT9,RO      ;STALL TIME COUNTER
990      003006 005277 176304      INC      @STREG      ;START CONVERSION
991      003012 012737 000200 001124      MOV      #BIT7,$GDDAT ;LOAD EXPECTED
992      003020 005300      15:  DEC      RO      ;STALL
993      003022 001376      BNE     15      ;TIME
994      003024 042777 100000 176264      BIC      #BIT15,@STREG ;MASK OUT ERROR BIT
995      003032 004737 003662      JSR     PC,TEST
996      003036 104001      ERROR   1      ;A/D DONE FLAG FAILED TO SET;BIT0 FAILED TO CLEAR
997      003040 017700 176256      MOV      @ADBUFF,RO ;CLEAR DONE FLAG FOR ITERATIONS
998
999      ;*****
1000     ;*TEST 10     TEST INIT CLEARS DONE FLAG
1001     ;*****
1002     003044 000004      †ST10: SCOPE
1003     003046 012737 000300 001160      MOV      #300,$TIMES      ;;DO 300 ITERATIONS
1004     003054 005037 001124      CLF     $GDDAT      ;CLEAR EXPECTED
1005     003060 005277 176232      INC      @STREG      ;START CONVERSION
1006     003064 105777 176226      25:  TSTB   @STREG
1007     003070 100375      BPL     25
1008     003072 000005      RESET
1009     003074 004737 003662      JSR     PC,TEST
1010     003100 104001      ERROR   1      ;DONE FLAG FAILED TO CLEAR
1011     003102 052777 000100 176034      BIS      #100,@$TKS      ;SET INTRPT. EN. BIT
1012
1013     ;*****
1014     ;*TEST 11     TEST A/D DONE FLAG CLEARS WHEN READ CONVERTED VALUE
1015     ;*****
1016     003110 000004      †ST11: SCOPE
1017     003112 012777 000001 176176      MOV      #BIT0,@STREG ;SET A/D START CONVERSION BIT
1018     003120 105777 176172      15:  TSTB   @STREG      ;WAIT FOR FLAG
1019     003124 100375      BPL     15
1020     003126 017700 176170      MOV      @ADBUFF,RO ;READ CONVERTED VALUE
1021     003132 004737 003662      JSR     PC,TEST
1022     003136 104001      ERROR   1      ;DONE FLAG FAILED TO CLEAR

```

```

1023 ;:*****
1024 ;*TEST 12 GENERATE INTERRUPT WHEN DONE FLAG SETS AFTER CONVERSION
1025 ;:*****
1026 003140 000004 †TST12: SCOPE
1027 ;* "ENTERING TEST 12" TYPED OUT TO TELL YOU THE NEXT
1028 ;*TEST THAT IS GOING TO BE EXECUTED. IT IS ONLY TYPED ON PASS 0.
1029 ;*THERE IS DANGER THAT THE UNIBUS COULD GET "HUNG" WHILE
1030 ;*EXECUTING TEST "12".
1031 003142 012700 000012 MOV #12,RO ;GET TEST NO.
1032 003146 004737 011152 JSR PC,DUMW ;PRINT MESSAGE
1033 003152 005037 177776 CLR PSW ;RESET PRIORITY
1034 003156 012777 003224 176140 MOV #1$,@VECTOR ;INTERRUPT VECTOR ADDRESS
1035 003164 012777 000101 176124 MOV #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
1036 003172 105777 176120 2$: TSTB @STREG ;WAIT FOR DONE
1037 003176 100375 BPL 2$ ;FLAG TO SET
1038 003200 017737 176112 001126 MOV @STREG,$BDDAT ;READ STATUS REGISTER
1039 003206 012737 000300 001124 MOV #BIT7!BIT6,$GDDAT ;GOOD DATA
1040 003214 104002 ERROR 2 ;FAILED TO INTERRUPT ON DONE
1041 003216 004737 011224 JSR PC,DUMC ;TYPE COMPLETED
1042 003222 000403 BR TST13 ;;BRANCH TO NEXT TEST
1043 003224 022626 1$: CMP (SP)+,(SP)+ ;RESET STACK POINTER
1044 003226 004737 011224 JSR PC,DUMC ;TYPE COMPLETED
1045
1046 ;:*****
1047 ;*TEST 13 TEST FOR CORRECTNESS OF PRIORITY OF INT. RUPT
1048 ;:*****
1049 003232 000004 †TST13: SCOPE
1050 ;* "ENTERING TEST 13" TYPED OUT TO TELL YOU THE NEXT
1051 ;*TEST THAT IS GOING TO BE EXECUTED. IT IS ONLY TYPED ON PASS 0.
1052 ;*THERE IS DANGER THAT THE UNIBUS COULD GET "HUNG" WHILE
1053 ;*EXECUTING TEST "13".
1054 003234 0 2700 000013 MOV #13,RO ;GET TEST NO.
1055 003240 004737 011152 JSR PC,DUMW ;PRINT MESSAGE
1056 003244 013737 001326 177776 MOV BASEBR,PSW ;GET ONE LEVEL LESS
1057 003252 162737 000040 177776 SUB #40,PSW ;PRIORITY
1058 003260 012777 003314 176036 MOV #1$,@VECTOR ;INTERRUPT VECTOR ADDRESS
1059 003266 012777 000101 176022 MOV #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
1060 003274 105777 176016 2$: TSTB @STREG ;WAIT FOR
1061 003300 100375 BPL 2$ ;DONE FLAG
1062 003302 017737 176010 001126 MOV @STREG,$BDDAT ;READ ST. REG.
1063 003310 104002 ERROR 2 ;INTERRUPT NOT SERVICED AT LEVEL 5
1064 003312 000401 BR 3$
1065 003314 022626 1$: CMP (SP)+,(SP)+ ;RESET STACK POINTER
1066 003316 012777 003376 176000 3$: MOV #4$,@VECTOR ;INTERRUPT VECTOR ADDRESS
1067 003324 013737 001326 177776 MOV BASEBR,PSW ;SET PRIORITY TO 6
1068 003332 012777 000101 175756 MOV #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE + START CONVERSION
1069 003340 105777 175752 5$: TSTB @STREG ;WAIT FOR
1070 003344 100375 BPL 5$ ;DONE FLAG
1071 003346 012777 001432 175750 6$: MOV #UNEXP,@VECTOR ;UNEXPECTED INTERRUPT VECTOR
1072 003354 005077 175736 CLR @STREG ;CLEAR BIT 6
1073 003360 005037 177776 CLR PSW ;RESET PRIORITY
1074 003364 017700 175732 MOV @ADBUFF,RO ;CLEAR DONE FLAG
1075 003370 004737 011224 JSR PC,DUMC ;TYPE COMPLETED
1076 003374 000403 BR TST14 ;;BRANCH TO NEXT TEST
1077 003376 022626 4$: CMP (SP)+,(SP)+ ;RESET STACK POINTER
1078 003400 104003 ERROR 3 ;INTERRUPTED AT LEVEL 6 - WRONG

```

```

1079 003402 000761          BN      65
1080
1081
1082
1083
1084 003404 000004          ;*****
;*TEST 14      TEST DONE FLAG CLEARS AFTER INTERRUPT
;*****
1084 003404 000004          TST14:  SCOPE
1085 003406 012737 000300 001160      MOV      #300,STIMES      ;;DO 300 ITERATIONS
1086 003414 012737 000100 001124      MOV      #BIT6,$GDDAT    ;LOAD GOOD DATA
1087 003422 012777 003456 175674      MOV      #15,$VECTOR    ;INTERRUPT VECTOR ADDRESS
1088 003430 012777 000101 175660      MOV      #BIT6!BIT0,$STREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
1089 003436 105777 175654          2$:    TSTB      $STREG      ;WAIT FOR
1090 003442 100375          BPL      2$              ;DONE FLAG
1091 003444 017737 175646 001126      MOV      $STREG,$BDDAT   ;READ STATUS REGISTER
1092 003452 104002          ERROR    2              ;FAILED TO GENERATE INTERRUPT
1093 003454 000412          BR       TST15          ;;GO TO NEXT TEST
1094 003456 022626          1$:    CMP      (SP)+,(SP)+ ;RESET STACK POINTER
1095 003460 004737 003662          JSR      PC,TEST
1096 003464 104001          ERROR    1              ;DONE FLAG NOT CLEARED AFTER AN INTERRUPT
1097 003466 017737 175630 001352      MOV      $ADDBUFF,TEMP   ;DUMMY READ BUFFER
1098 003474 012777 001432 175622      MOV      #UNEXP,$VECTOR ;LOAD INTRPT. ADDRESS
1099
1100
1101
1102
1103 003502 000004          ;*****
;*TEST 15      TEST ERROR FLAG SETS IF 2ND CONVERSION ENDS BEFORE READING BUFFER
;*****
1103 003502 000004          TST15:  SCOPE
1104 003504 012777 000001 175604      MOV      #BIT0,$STREG    ;START CONVERSION
1105 003512 105777 175600          1$:    TSTB      $STREG    ;WAIT FOR
1106 003516 100375          BPL      1$
1107 003520 012737 100200 001124      2$:    MOV      #BIT15!BIT7,$GDDAT ;LOAD EXPECTED VALUE
1108 003526 012777 000001 175562      MOV      #BIT0,$STREG    ;START 2ND CONVERSION
1109 003534 012700 001000          MOV      #BIT9,R0        ;WAIT FOR 2ND
1110 003540 005300          3$:    DEC      R0          ;CONVERSION TO END
1111 003542 001376          BNE      3$
1112 003544 004737 003662          4$:    JSR      PC,TEST
1113 003550 104001          ERROR    1              ;ERROR FLAG NOT SET WHEN 2ND
1114
1115 003552 017700 175544          MOV      $ADDBUFF,R0    ; CONVERT ENDS BEFORE READ BUFFER FROM FIRST
; CLEAR DONE FLAG

```

```

1116                                     ;:*****
1117                                     ;*TEST 16      TEST ERROR FLAG SETS IF START 2ND CONV. BEFORE DONE FLAG SETS
1118                                     ;:*****
1119 003556 000004                         TST16: SCOPE
1120 003560 012737 100000 001124          MOV     #BIT15,$GDDAT      ;LOAD EXPECTED DATA
1121 003566 012700 000100                 MOV     #100,R0           ;STALL TIME COUNTER
1122 003572 012777 000001 175516         MOV     #BIT0,@STREG      ;START CONVERSION
1123 003600 112777 000001 175510         MOVVB   #BIT0,@STREG      ;START NEXT CONVERSION
1124 003606 005300                         DECS:  DEC     R0          ;STALL TIME
1125 003610 001376                         BNE     DECS
1126 003612 017737 175500 001126         MOV     @STREG,$BDDAT     ;READ STATUS REGISTER
1127 003620 042737 077777 001126         BIC     #77777,$BDDAT    ;MASK OUT BIT 15
1128 003626 023737 001124 001126         CMP     $GDDAT,$BDDAT    ;COMPARE RESULTS
1129 003634 001401                         BEQ     1$               ;:BRANCH OVER ERROR
1130 003636 104001                         ERROR   1                ;ERROR FLAG NOT SET WHEN 2ND
1131                                     ;CONVERT BEGINS BEFORE FIRST DONE
1132 003640 017700 175456 001126         1$:   MOV     @ADBUFF,R0   ;READ CONVERTED VALUE
1133 003644 005077 175446                 CLR     @STREG           ;CLEAR STATUS REGISTER
1134 003650 000004                         SCOPE
1135 003652 000207                         RTS     PC               ;RETURN TO TEST SECTION
1136
1137
1138                                     ;:SUBROUTINE FOR LOGIC TESTS;;
1139 003654 013777 001124 175434         TESTIT: MOV    $GDDAT,@STREG ;LOAD EXPECTED VALUE
1140 003662 017737 175430 001126         TEST:  MOV    @STREG,$BDDAT ;READ ST. REG.
1141 003670 023737 001124 001126         CMP    $GDDAT,$BDDAT     ;COMPARE RESULTS
1142 003676 001002                         BNE    RETERR           ;:ERROR RETURN
1143 003700 062716 000002                 ADD    #2,(SP)          ;BUMP RETURN ADDRESS TO GET AROUND ERROR
1144 003704 000207                         RETERR: RTS    PC

```

```

1145 .SBTTL WRAPAROUND TEST SECTION
1146 003706 WRAP:
1147 :*****
1148 :*TEST 17 TEST CH14 GROUND
1149 :*****
1150 003706 000240 †ST17: NOP
1151 003710 012737 000010 001160 MOV #10,STIMES ;;DO 10 ITERATIONS
1152 003716 012737 000017 001102 MOV #STN-1,STSTNM
1153 003724 012737 003706 001110 MOV #TST17,SLPERR
1154 003732 012737 003706 001106 MOV #TST17,SLPADR
1155 003740 004537 010772 JSR RS,CONVRT ;DO 8 CONVERSIONS
1156 003744 000014 14 JSR RS,COMPAR ;COMPARE RESULTS
1157 003746 004537 011104 JSR RS,COMPAR ;COMPARE RESULTS
1158 003752 004000 4000 ;NOMINAL
1159 003754 011616 V50 ;TOLERANCE
1160 003756 104004 ERROR 4 ;ERROR-CH14 NOT GROUND-AD11K MUST BE IN
;SINGLE-ENDED CONFIGURATION,G5036 WRAPAROUND
;MODULE MUST BE PRESENT,CHECK CONNECTION A-VV,VV-A

:*****
: *TEST 20 TEST CONVERSION FROM EXT. START
:*****
1168 003760 000004 †ST20: SCOPE
1169 003762 012737 000010 001160 MOV #10,STIMES ;;DO 10 ITERATIONS
1170 003770 012777 000020 175320 MOV #BIT4,STREG ;ENABLE EXTERNAL START
1171 003776 012700 001000 MOV #BIT9,RO ;TIME DELAY COUNTER
1172 004002 012737 000220 001124 MOV #BIT7,BIT4,SGDCAT ;LOAD EXPECTED
1173 004010 012777 000200 175304 MOV #200,ADDBUFF ;SIMULATE EXT. START WITH G5036
;WRAPAROUND MODULE PRESENT
1174
1175 004016 005300 15: DEC RO
1176 004020 001376 BNE 15
1177 004022 004737 003662 JSR PC,TEST
1178 004026 104001 ERROR 1 ;FAILED TO DO CONVERSION FROM EXT. START
1179 004030 017700 175266 MOV ADDBUFF,RO
1180 004034 005077 175256 CLR STREG ;CLEAR ST. REG.

:*****
: *TEST 21 TEST CH0 GROUND
:*****
1186 004040 000004 †ST21: SCOPE
1187 004042 012737 000010 001160 MOV #10,STIMES ;;DO 10 ITERATIONS
1188 004050 004537 010772 JSR RS,CONVRT ;CONVERT 8 TIMES
1189 004054 000000 0 ;CONVERT 8 TIMES
1190 004056 004537 011104 JSR RS,COMPAR ;COMPARE RESULTS
1191 004062 004000 4000 ;NOMINAL
1192 004064 011610 V1 ;TOLERANCE
1193 004066 104004 ERROR 4 ;ERROR ON A/D CHANNEL

```

```

1194
1195
1196
1197 004070 000004
1198 004072 012737 000010 001160
1199 004100 004537 010772
1200 004104 000001
1201 004106 004537 011104
1202 004112 004000
1203 004114 011612
1204 004116 104004
1205
1206
1207
1208
1209 004120 000004
1210 004122 012737 000010 001160
1211 004130 004537 010772
1212 004134 000002
1213 004136 004537 011104
1214 004142 004632
1215 004144 011616
1216 004146 104004
1217
1218
1219
1220
1221
1222 004150 000004
1223 004152 012737 000010 001160
1224 004160 004537 010772
1225 004164 000003
1226 004166 004537 011104
1227 004172 006000
1228 004174 011624
1229 004176 104004
1230
1231
1232
1233
1234 004200 000004
1235 004202 012737 000010 001160
1236 004210 004537 010772
1237 004214 000004
1238 004216 004537 011104
1239 004222 002000
1240 004224 011624
1241 004226 104004

*****
*TEST 22 TEST CH1 GROUND
*****
†ST22: SCOPE
MOV #10,STIMES ;;DO 10 ITERATIONS
JSR R5,CONVRT ;CONVERT 8 TIMES
1 ;CHANNEL 1
JSR R5,COMPAR ;COMPARE RESULTS
4000 ;NOMINAL
V2 ;TOLERANCE
ERROR 4 ;ERROR ON A/D CHANNEL

*****
*TEST 23 TEST CH2 +1 VOLT
*****
†ST23: SCOPE
MOV #10,STIMES ;;DO 10 ITERATIONS
JSR R5,CONVRT ;CONVERT 8 TIMES
2 ;CHANNEL 2
JSR R5,COMPAR ;COMPARE RESULTS
4632 ;NOMINAL
V50 ;TOLERANCE
ERROR 4 ;ERROR ON A/D CHANNEL
;AD11K MUST BE SET UP FOR +OR- 5V OR +OR- 5.12V

*****
*TEST 24 TEST CH3 +2.5 VOLTS
*****
†ST24: SCOPE
MOV #10,STIMES ;;DO 10 ITERATIONS
JSR R5,CONVRT ;CONVERT 8 TIMES
3 ;CHANNEL 3
JSR R5,COMPAR ;COMPARE RESULTS
6000 ;NOMINAL
V240 ;TOLERANCE
ERROR 4 ;ERROR ON A/D CHANNEL

*****
*TEST 25 TEST CH4 -2.5 VOLTS
*****
†ST25: SCOPE
MOV #10,STIMES ;;DO 10 ITERATIONS
JSR R5,CONVRT ;CONVERT 8 TIMES
4 ;CHANNEL 4
JSR R5,COMPAR ;COMPARE RESULTS
2000 ;NOMINAL
V240 ;TOLERANCE
ERROR 4

```

```

1242
1243
1244
1245 004233 000004
1246 004232 012737 000001 001160
1247 004240 005077 175056
1248 004244 004737 005026
1249 004250 004537 010772
1250 004254 000012
1251 004256 013704 001352
1252 004262 004537 011104
1253 004266 002376
1254 004270 011622
1255 004272 104004
1256 004274 005037 001424
1257 004300 012702 000001
1258 004304 010277 175012 1$:
1259 004310 004737 005026
1260 004314 004537 010772
1261 004320 000012
1262 004322 005737 001424
1263 004326 001010
1264 004330 023727 001352 004000
1265 004336 002404
1266 004340 005237 001424
1267 004344 010237 001420
1268 004350 020227 000200 2$:
1269 004354 001003
1270 004356 013737 001352 004446
1271 004364 013703 001352 3$:
1272 004370 160437 001352
1273 004374 010304
1274 004376 004537 011104
1275 004402 000006
1276 004404 011626
1277 004406 104004
1278 004410 005202
1279 004412 020227 000400
1280 004416 001332
1281 004420 000005
1282 004422 052777 000100 174514
1283 004430 004737 005026
1284 004434 004537 010772
1285 004440 000012
1286 004442 004537 011104
1287 004446 000000 4$:
1288 004450 011612
1289 004452 104004

```

```

*****
:TEST 26 TEST VERNIER OFFSET DAC ON CH12
*****
↑ST26: SCOPE
MOV #1,STIMES ;;DO 1 ITERATION
CLR JADBUFF ;VERNIER DAC=0
JSR PC,DAWAIT ;DELAY FOR DAC SETTling
JSR R5,CONVRT ;CONV. CH12, DIRECT VERNIER DAC
12
MOV TEMP,R4 ;SAVE VALUE IN R4
JSR R5,COMPAR ;COMPARE RESULTS
2376 ;WITH -1.875 VOLTS
V115 ;TOLERANCE OF 10%
ERROR 4
CLR MAX
MOV #1,R2
1$: MOV R2,JADBUFF ;SET UP NEXT VERNIER DAC VALUE
JSR PC,DAWAIT ;DELAY FOR DAC SETTling
JSR R5,CONVRT ;CONVERT IT
12
TST MAX
BNE 2$
CMP TEMP,#4000
BLT 2$
INC MAX
MOV R2,MIN
2$: CMP R2,#200
BNE 3$
MOV TEMP,4$
3$: MOV TEMP,R3 ;SAVE VALUE
SUB R4,TEMP ;TEMP=DIFF. BETWEEN VALUE&PREVIOUS
MOV R3,R4 ;SET UP PREVIOUS VALUE FOR NEXT TIME THRU
JSR R5,COMPAR ;COMPARE RESULTS
6 ;WITH 15 MILLIVOLTS(1 DAC LSB)
V5
ERROR 4
INC R2
CMP R2,#400 ;DONE?
BNE 1$ ;NO-DO NEXT VERNIER DAC VALUE
RESET ;SHOULD SET VERNIER DAC TO 200
BIS #100,STKS
JSR PC,DAWAIT ;LET DAC SETTLE
JSR R5,CONVRT ;CONVERT IT
12
JSR R5,COMPAR ;COMPARE RESULTS
0
V2
ERROR 4

```

E03

MAINDEC-11-DZADL-A
DZADLA.P11 T27

MACY11 27(732) 25-SEP-76 10:38 PAGE 31
TEST CH13 +2.5 VOLTS

1290				
1291				
1292				
1293	004454	000004		
1294	004456	012737	000010	00:160
1295	004464	004537	010772	
1296	004470	000013		
1297	004472	004537	011104	
1298	004476	006000		
1299	004500	011620		
1300	004502	104004		
1301				
1302				
1303				
1304	004504	000004		
1305	004506	012737	000010	001160
1306	004514	004537	010772	
1307	004520	000017		
1308	004522	004537	011104	
1309	004526	007146		
1310	004530	011624		
1311	004532	104004		

```

*****
*TEST 27      TEST CH13 +2.5 VOLTS
*****
TST27:  SCOPE
        MOV      #10,STIMES      ;;DO 10 ITERATIONS
        JSR      R5,CONVRT      ;CONVERT 8 TIMES
        13
        JSR      R5,COMPAR      ;COMPARE RESULTS
        6000      ;NOMINAL
        V144      ;TOLERANCE
        ERROR    4

*****
*TEST 30      TEST CH17 +4V
*****
TST30:  SCOPE
        MOV      #10,STIMES      ;;DO 10 ITERATIONS
        JSR      R5,CONVRT      ;CONVERT 8 TIMES
        17      ;CHANNEL 17
        JSR      R5,COMPAR      ;COMPARE RESULTS
        7146      ;NOMINAL
        V240      ;TOLERANCE
        ERROR    4      ;ERROR ON A/D CHANNEL

```

```

1312
1313
1314
1315 004534 000004
1316 004536 012737 000001 001160
1317 004544 013737 001336 001366
1318 004552 013737 001336 001364
1319 004560 012737 004001 001414
1320 004566 004537 006504
1321 004572 000062
1322 004574 013737 001410 001352
1323 004602 004537 006504
1324 004606 000062
1325 004610 063737 001410 001352
1326 004616 162737 000062 001352
1327 004624 013700 001420
1328 004630 006300
1329 004632 160037 001352
1330 004636 104400 013573
1331 004642 013702 001352
1332 004646 004737 011374
1333 004652 104400 013606
1334 004656 004537 011104
1335 004662 000000
1336 004664 011630
1337 004666 000401
1338 004670 000403
1339 004672 104400 012375
1340 004676 000402
1341 004700 104400 012364

```

```

*****
;TEST 31      OFFSET ON CHO
*****
TST31:  SCOPE
        MOV     #1,STIMES      ;;DO 1 ITERATION
        MOV     BASECH,CHANL   ;LOAD CHANNEL
        MOV     BASECH,DUMMY   ;LOAD DUMMY
        MOV     #4001,EDGE
        JSR     RS,SARSUB
        SO.
        MOV     DAC,TEMP
        JSR     RS,SARSUB
        SO.
        ADD     DAC,TEMP
        SUB     #62,TEMP
        MOV     MIN,RO
        ASL     RO
        SUB     RO,TEMP
        TYPE    ,MOFSET        ;TYPE ASCIZ STRING
        MOV     TEMP,R2
        JSR     PC,DECTYP
        TYPE    ,MLSB          ;TYPE ASCIZ STRING
        JSR     RS,COMPAR      ;IS RESULT WITHIN LIMITS?
        O
        VSOD
        BR      OFFERR        ;NO-ERROR
        BR      OFFOK        ;YES-OK
OFFERR: TYPE    ,ERMSG
OFFOK:  BR      TST32
        TYPE    ,OKMSG
        ;;GO TO NEXT TEST

```

1342
1343
1344
1345
1346
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380

004704 000004
004706 012737 000001 001160
004714 012737 000116 001352
004722 004537 010564
004726 000015
004730 004537 010564
004734 000007
004736 004537 010564
004742 000016

004744 000004
004746 012737 000001 001160
004754 004537 006220
004760 000015
004762 000016
004764 012737 000116 001352
004772 004537 006220
004776 000016
005000 000015

005002 000004
005004 012737 000001 001160
005012 005737 001202
005016 001402
005020 004737 006712
005024 000207

005026 005000
005030 105300
005032 001376
005034 000207

```

*****
*TEST 32      NOISE TEST ON 8 EDGES
*****
†ST32:  SCOPE
        MOV      #1, $TIMES      ;;DO 1 ITERATION
        MOV      #16, TEMP      ;DAC VALUE
        JSR      R5, NO18       ;NOISE AT -FULL SCALE
        IS      15
        JSR      R5, NO18       ;NOISE AT MID-RANGE
        IS      7
        JSR      R5, NO18       ;NOISE AT +FULL SCALE
        IS      16

*****
*TEST 33      SETTLE TEST ON 8 EDGES
*****
†ST33:  SCOPE
        MOV      #1, $TIMES      ;;DO 1 ITERATION
        JSR      R5, SET8       ;SETTLE-POSITIVE DIRECTION
        IS      15
        MOV      #16, TEMP
        JSR      R5, SET8       ;SETTLE-NEGATIVE DIRECTION
        IS      16
        IS      15

*****
*TEST 34      DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST
*****
†ST34:  SCOPE
        MOV      #1, $TIMES      ;;DO 1 ITERATION
        TST      $PASS          ;FIRST TIME-SKIP DIFLIN
        BEQ      LEND
        JSR      PC, DIFLIN
LEND:   RTS      PC            ;RETURN TO TEST SECTION

DAWAIT: CLR      R0
IS:     DECB    R0
        BNE     IS
        RTS     PC
```

1381					.SBTTL	CALIBRATION TEST	
1382	005036	012737	005036	001370	BEGINC:	MOV #BEGINC,TADDR	; TEST ADDRESS IN TADDR
1383	005044	005077	174246			CLR @STREG	; CLEAR STATUS REGISTER
1384	005050	104400	013503			TYPE HEADS	; TYPE OUT HEADING
1385	005054	005037	177776			CLR PSW	
1386	005060	017700	174054		1\$:	MOV @SWR,R0	; READ CHANNEL FROM SWITCH REG.
1387	005064	042700	177700			SIC #177700,R0	; ISOLATE MUX BITS
1388	005070	032777	020000	174042		BIT #BIT13,@SWR	; IS BIT 13 SET?
1389	005076	001005				BNE 2\$:: YES, SKIP TYPEOUT
1390	005100	104400	012207			TYPE CH	
1391	005104	010046				MOV R0,-(SP)	:: SAVE R0 FOR TYPEOUT
1392							:: TYPE CHANNEL
1393	005106	104402				TYPOS	:: GO TYPE--OCTAL ASCII
1394	005110	002				.BYTE 2	:: TYPE 2 DIGIT(S)
1395	005111	000				.BYTE 0	:: SUPPRESS LEADING ZEROS
1396	005112	012777	001610	174204	2\$:	MOV #RETURN,@VECTOR	; ADDRESS AFTER INTRPT.
1397	005120	000300				SWAB R0	; SWITCH BYTES
1398	005122	052700	000100			BIS #BIT6,R0	
1399	005126	010077	174164			MOV R0,@STREG	; LOAD THE CHANNEL
1400	005132	012702	000010			MOV #10,R2	; TYPEOUT COUNTER
1401	005136	005277	174154		3\$:	INC @STREG	; START CONVERSION
1402	005142	000001				WAIT	; WAIT FOR INTRPT.
1403	005144	017700	174152			MOV @ADBUFF,R0	; READ CONVERTED VALUE
1404	005150	032777	020000	173762		BIT #BIT13,@SWR	; IS BIT 13 SET?
1405	005156	001403				BEQ 4\$; NOT SET, TYPE OUT LIST
1406	005160	010077	173756			MOV R0,@DISPLAY	; PUT VALUE IN DISPLAY FOR DISPLAY CONTR
1407	005164	000735				BR 1\$; REPEAT CONVERSION
1408	005166	104400	012212		4\$:	TYPE SPACE	
1409	005172	010046				MOV R0,-(SP)	:: SAVE R0 FOR TYPEOUT
1410							:: PRINT OCTAL CONVERTED VALUE
1411	005174	104402				TYPOS	:: GO TYPE--OCTAL ASCII
1412	005176	004				.BYTE 4	:: TYPE 4 DIGIT(S)
1413	005177	001				.BYTE 1	:: TYPE LEADING ZEROS
1414	005200	012701	010000			MOV #10000,R1	
1415	005204	005301			5\$:	DEC R1	
1416	005206	001376				BNE 5\$	
1417	005210	005302				DEC R2	; DECREMENT THE COUNTER
1418	005212	001351				BNE 3\$; NO CARRIAGE RETURN
1419	005214	104400	001171			TYPE \$CRLF	; CARRIAGE RETURN
1420	005220	000717				BR 1\$; REPEAT CONVERSION

1421					.SBTTL		LOGIC TEST SECTION	
1422	005222	012737	005222	001370	BEGL:	MOV	#BEGL, TADDR	; TEST ADDRESS
1423	005230	004737	002422			JSR	PC, TESTAD	; NO OF ADDITIONAL AD'S
1424	005234	004737	002556		1\$:	JSR	PC, BEGINL	; LOGIC TESTS
1425	005240	004737	005410			JSR	PC, BUMPAD	; MORE TO TEST?
1426	005244	000773				BR	1\$; TEST NEXT A/D
1427	005246	012737	005234	011706		MOV	#1\$, AGTST	; ADDRESS FOR EOP
1428	005254	000137	011710			JMP	SEOP	; TYPE END OF PASS
1429								
1430					.SBTTL		AUTO TEST	
1431	005260	012737	005260	001370	BEGINA:	MOV	#BEGINA, TADDR	; TEST ADDRESS
1432	005266	005037	001202			CLR	\$PASS	; CLEAR PASS COUNTER
1433	005272	004737	002422			JSR	PC, TESTAD	; NO. OF AD'S TO BE TESTED
1434	005276	004737	002556		1\$:	JSR	PC, BEGINL	; LOGIC TESTS
1435	005302	104400	012665			TYPE	MEND	; TYPE END OF LOGIC TEST
1436	005306	013746	001316			MOV	\$STREG, -(SP)	; SAVE STREG FOR TYPEOUT
1437	005312	104402				TYPOS		; TYPE OCTAL NUMBER
1438	005314	006				.BYTE	6	; TYPE 6 DIGITS
1439	005315	001				.BYTE	1	; TYPE LEADING ZEROS
1440	005316	104400	001171			TYPE	\$SCRLF	; TYPE A CR, LF
1441	005322	004737	003706			JSR	PC, WRAP	
1442	005326	004737	005410			JSR	PC, BUMPAD	; TEST NEXT A/D
1443	005332	000761				BR	1\$; TEST NEXT AD
1444	005334	012737	005276	011706		MOV	#1\$, AGTST	; ADDRESS FOR EOP
1445	005342	000137	011710			JMP	SEOP	; TYPE END OF PASS
1446								
1447					.SBTTL		WRAPAROUND TEST	
1448	005346	012737	005346	001370	BEGINW:	MOV	#BEGINW, TADDR	; TEST ADDRESS
1449	005354	005037	001202			CLR	\$PASS	; CLEAR PASS COUNT
1450	005360	004737	002422			JSR	PC, TESTAD	; NO. OF AD'S TO BE TESTED
1451	005364	004737	003706		1\$:	JSR	PC, WRAP	; WRAPAROUND TESTS
1452	005370	004737	005410			JSR	PC, BUMPAD	; MORE A/D'S TO BE TESTED?
1453	005374	000773				BR	1\$; YES-GO TEST NEXT AD11K
1454	005376	012737	005364	011706		MOV	#1\$, AGTST	
1455	005404	000137	011710			JMP	SEOP	; INCREMENTS \$PASS

1456					.SBTTL	DETERMINE IF MORE AD11K'S TO BE TESTED	
1457	005410	005737	001360		BUMPAD:	TST NBEXT	;ADDITIONAL AD'S?
1458	005414	001424				BEQ FIXADR	;NO-INITIALIZE ADDRESSES
1459	005416	063737	001332	001316		ADD VADR,STREG	;SET UP NEW ST. REG.
1460	005424	063737	001332	001320		ADD VADR,ADST1	;SET UP NEW ADST1
1461	005432	063737	001332	001322		ADD VADR,ADBUFF	;SET UP NEW BUFFER ADDRESS
1462	005440	063737	001334	001324		ADD VVCT,VECTOR	;SET UP NEW VECTOR
1463	005446	063737	001334	001330		ADD VVCT,VECTR1	
1464	005454	005077	173650			CLR VVECTR1	
1465	005460	005337	001360			DEC NBEXT	;ONE LESS AD11K
1466	005464	000446				BR BYPASS	
1467	005466	062716	000002		FIXADR:	ADD #2,(SP)	
1468	005472	013737	001250	001316	FIXONE:	MOV \$BASE,STREG	;RELOAD INITIAL ADDRESSES
1469	005500	013737	001250	001320		MOV \$BASE,ADST1	
1470	005506	013737	001250	001322		MOV \$BASE,ADBUFF	
1471	005514	005237	001320			INC ADST1	
1472	005520	062737	000002	001322		ADD #2,ADBUFF	
1473	005526	013737	001244	001324		MOV \$VECT1,VECTOR	
1474	005534	042737	170000	001324		BIC #170000,VECTOR	
1475	005542	113737	001245	001326		MOVB \$VECT1+1,BASEBR	
1476	005550	105037	001327			CLRB BASEBR+1	;CLEAR HIGH BYTE
1477	005554	013737	001324	001330		MOV VECTOR,VECTR1	
1478	005562	062737	000002	001330		ADD #2,VECTR1	
1479	005570	005077	173534			CLR VVECTR1	
1480	005574	013737	001362	001360		MOV NMBEXT,NBEXT	;RESET COUNTER
1481					::LOAD	..+2 AND HALT TRAP CATCH;;	
1482	005602	012700	000216		BYPASS:	MOV #216,R0	;FILL .+2
1483	005606	012701	000214			MOV #214,R1	;LOAD HALT
1484	005612	020137	001340		1\$:	CMP R1,KBVECT	
1485	005616	001410				BEQ 2\$	
1486	005620	010021				MOV R0,(R1)+	
1487	005622	005021				CLR (R1)+	
1488	005624	010100				MOV R1,R0	
1489	005626	005720				TST (R0)+	
1490	005630	020027	001002			CMP R0,#1002	
1491	005634	001366				BNE 1\$	
1492	005636	000207				RTS PC	;TEST NEXT A/D
1493	005640	022021			2\$:	CMP (R0)+,(R1)+	
1494	005642	022021				CMP (R0)+,(R1)+	
1495	005644	000762				BR 1\$	
1496							
1497							
1498					.SBTTL	NOISE TEST, 1 EDGE	
1499	005646	012737	005646	001370	BEGINN:	MOV #BEGINN,TADDR	;TEST ADDRESS IN TADDR
1500	005654	104400	012016			TYPE ,NOIMSG	;ASK FOR CHANNEL
1501	005660	104400	013522			TYPE ,ASKCH	
1502	005664	017737	173250	001354	1\$:	MOV \$SWR,CH1	;LOAD CHANNEL
1503	005672	042737	177700	001354		BIC #177700,CH1	
1504	005700	012737	000200	001352		MOV #200,TEMP	;LOAD DAC VALUE
1505	005706	004537	010300			JSR R5,NOITST	;GO TO NOISE SUBROUTINE
1506	005712	001354				CHI	
1507	005714	000763				BR 1\$	

K03

MAINDEC-11-DZADL-A
DZADLA.P11

MACY11 27(732) 25-SEP-76 10:38 PAGE 37
INTERCHANNEL SETTLING TEST, 1 EDGE

```

1508          .SBTTL      INTERCHANNEL SETTLING TEST, 1 EDGE
1509 005716 012737 005716 001370 BEGINS: MOV      #BEGINS,TADDR      ;TEST ADDRESS IN TADDR
1510 005724 104400 012036          TYPE      ,SETMSG        ;ASK FOR CHANNELS
1511 005730 104406          RDOCT
1512 005732 012637 001354          MOV      (SP)+,CH1
1513 005736 104400 012323          TYPE      ,TOMSG
1514 005742 104406          RDOCT
1515 005744 012637 001356          MOV      (SP)+,CH2
1516 005750 012737 000200 001352 BK3:  MOV      #200,TEMP      ;LOAD DAC
1517 005756 013737 001356 001366          MOV      CH2,CHANL
1518 005764 004737 006324          JSR      PC,GETEDG      ;GET EDGE VALUES
1519 005770 005002          CLR      R2
1520 005772 004737 006156          JSR      PC,SET1A      ;SCALING = .02 LSB
1521 005776 004737 006156          JSR      PC,SET1A      ;MAKE IT .01 LSB
1522 006002 100001          BPL      POSR2
1523 006004 005402          NEG      R2
1524 006006 010204          POSR2: MOV      R2,R4
1525 006010 012737 000001 006502          MOV      #1,EDGFLG
1526 006016 004737 006024          JSR      PC,TYPSET
1527 006022 000752          BR      BK3
1528 006024 004737 011374          TYPSET: JSR      PC,DECTYP
1529 006030 104400 012217          TYPE      LSB
1530 006034 013746 001356          MOV      CH2,-(SP)      ;;SAVE CH2 FOR TYPEOUT
1531          ;;TYPE CH
1532 006040 104402          TYPOS
1533 006042 002          .BYTE 2
1534 006043 000          .BYTE 0
1535 006044 104400 013614          TYPE      MAT
1536 006050 004737 006440          JSR      PC,TYPEDG
1537 006054 104400 012232          TYPE      SETCH
1538 006060 013746 001354          MOV      CH1,-(SP)      ;;SAVE CH1 FOR TYPEOUT
1539          ;;TYPE CH
1540 006064 104402          TYPOS
1541 006066 002          .BYTE 2
1542 006067 000          .BYTE 0
1543 006070 104400 012254          TYPE      ATMSG
1544 006074 013737 001354 006122          MOV      CH1,1$
1545 006102 163737 001336 006122          SUB      BASECH,1$
1546 006110 012777 000200 173204          MOV      #200,ADDBUFF
1547 006116 004537 010772          JSR      R5,CONVRT
1548 006122 000000          IS: 0
1549 006124 013746 001352          MOV      TEMP,-(SP)      ;;SAVE TEMP FOR TYPEOUT
1550          ;;TYPE VALUE
1551 006130 104402          TYPOS
1552 006132 004          .BYTE 4
1553 006133 001          .BYTE 1
1554 006134 020437 011636          CMP      R4,VSET
1555 006140 003003          BGT      ERR
1556 006142 104400 012364          TYPE      ,OKMSG
1557 006146 000207          RTS      PC

```

```

1558 006150 104400 012375      ERR:  TYPE  ERMSG
1559 006154 000207              RTS    PC
1560
1561
1562
1563      ;;SUBROUTINE FOR SETTLING TESTS;;
1564 006156 013737 001356 001364 SET1A:  MOV    CH2,DUMMY      ;LOAD DUMMY
1565 006164 004537 006504              JSR    R5,SARSUB      ;DO SAR ROUTINE AT 50%
1566 006170 000362              SO.
1567 006172 063702 001410              ADD    DAC,R2        ;ADD RESULT TO R2
1568 006176 013737 001354 001364      MOV    CH1,DUMMY      ;CHANGE DUMMY VALUE
1569 006204 004537 006504              JSR    R5,SARSUB      ;DO SAR ROUTINE AT 50%
1570 006210 000062              SO.
1571 006212 163702 001410              SUB    DAC,R2        ;SUBTRACT RESULT FROM R2
1572 006216 000207              RTS    PC            ;RETURN
1573
1574 006220 012537 001354      SET8:  MOV    (R5)+,CH1      ;GET FIRST CHANNEL
1575 006224 012537 001356              MOV    (R5)+,CH2      ;GET SECOND CHANNEL
1576 006230 063737 001336 001354      ADD    BASECH,CH1
1577 006236 063737 001336 001356      ADD    BASECH,CH2
1578 006244 004737 006324              JSR    PC,GETEDG      ;GET EDGE VALUES
1579 006250 005002              CLR    R2
1580 006252 012703 000010              MOV    #10,R3        ;SET UP COUNTER
1581 006256 004737 006156      SETAA: JSR    PC,SET1A      ;GET SETTLE VALUES
1582 006262 005237 001414              INC    EDGE
1583 006266 005303              DEC    R3
1584 006270 001372              BNE   SETAA          ;REPEAT 8 TIMES
1585 006272 162737 000010 001414      SUB    #10,EDGE
1586 006300 005702              TST   R2
1587 006302 100001              BPL   R2POS
1588 006304 005402              NEG   R2
1589 006306 010204      R2POS: MOV    R2,R4
1590 006310 012737 000010 006502      MOV    #8,EDGFLG
1591 006316 004737 006024              JSR   PC,TYPSET
1592 006322 000205              RTS    R5            ;TYPE OUT RESULTS
                                ;RETURN

```

```

1593          ;SUBROUTINE TO GET EDGE VALUE
1594          ;CALL=JSR PC,GETEDG
1595          ;CONVERSIONS ON A/D CHANNEL 'CHANL'
1596          ;RESULT IN EDGE, USES R0
1597 006324 013777 001352 172770 GETEDG: MOV TEMP, @ADBUFF
1598 006332 113700 001366          MOVB CHANL, R0          ;GET CHANNEL
1599 006336 000300          SWAB R0          ;SET UP A.D STATUS REG.
1600 006340 052700 000100          BIS #100, R0          ;ENABLE INTRPT.
1601 006344 010077 172746          MOV R0, @STREG
1602 006350 012700 000100          MOV #100, R0          ;DAC SETTLING DELAY
1603 006354 005300          IS: DEC R0
1604 006356 001376          BNE IS
1605 006360 005037 001414          CLR EDGE
1606 006364 012700 000010          MOV #10, R0
1607 006370 012777 001610 172726 CONV: MOV #RETURN, @VECTOR          ;RETURN ADDRESS
1608 006376 005277 172714          INC @STREG          ;START CONVERSION
1609 006402 000001          WAIT          ;WAIT FOR INTERRUPT
1610 006404 067737 172712 001414          ADD @ADBUFF, EDGE
1611 006412 005300          DEC R0
1612 006414 001370          BNE CONV
1613 006416 006237 001414          ASR EDGE
1614 006422 006237 001414          ASR EDGE
1615 006426 006237 001414          ASR EDGE
1616 006432 005537 001414          ADC EDGE
1617 006436 000207          RTS PC
1618
1619          ;;SUBROUTINE TO TYPE EDGE VALUES;;
1620 006440 013703 001414          TYPEDG: MOV EDGE, R3
1621 006444 010346          MOV R3, -(SP)          ;;SAVE R3 FOR TYPEOUT
1622          ;TYPE OCTAL VALUE OF EDGE
1623 006446 104402          TYPOS          ;GO TYPE--OCTAL ASCII
1624 006450 004          .BYTE 4          ;TYPE 4 DIGIT(S)
1625 006451 001          .BYTE 1          ;TYPE LEADING ZEROS
1626 006452 023727 006502 000001          CMP EDGFLG, #1
1627 006460 001407          BEQ RET
1628 006462 062703 000007          ADD #7, R3
1629 006466 104400 013564          TYPE C1          ;TYPE ASCII STRING
1630 006472 010346          MOV R3, -(SP)          ;SAVE R3 FOR TYPEOUT
1631          ;TYPE EDGE VALUE
1632 006474 104402          TYPOS          ;GO TYPE--OCTAL ASCII
1633 006476 004          .BYTE 4          ;TYPE 4 DIGIT(S)
1634 006477 001          .BYTE 1          ;TYPE LEADING ZEROS
1635 006500 000207          RET: RTS PC
1636 006502 000000          EDGFLG: 0

```

```

1637 ;SUBROUTINE TO DO SUCCESSIVE APPROXIMATION ROUTINE
1638 ;CALL=JSR R5,SARSUB
1639 ; XXX:XXX=PERCENT
1640 ;RESULT RETURNED IN 'DAC' USES R0,R1,R4
1641 006504 012537 001426 SARSUB: MOV (R5)+,PERCNT ;GET PERCENT
1642 006510 006337 001426 ASL PERCNT
1643 006514 006337 001426 ASL PERCNT
1644 006520 006337 001426 ASL PERCNT ;RESCALE PERCENT FOR 1600.
1645 006524 006337 001426 ASL PERCNT ;POINTS PER BURST
1646 006530 012737 000200 001416 SAR1: MOV #200,BITPNT ;INITIALIZE BIT POINTER AT MSB
1647 006536 005037 001410 CLR DAC ;INITIALIZE DAC VALUE
1648 006542 005000 TRY: CLR R0
1649 006544 063737 001416 001410 AND BITPNT,DAC ;TRY BIT
1650 006552 013777 001410 172542 MOV DAC,@ADBUFF
1651 006560 012737 000100 001412 MOV #100,DELAY
1652 006566 005337 001412 1$: DEC DELAY ;STALL TIME
1653 006572 001375 BNE 1$
1654 006574 012701 003100 MOV #1600,R1 ;SET UP FOR 1600. CONVERSIONS
1655 006600 113777 001364 172512 NXTCVT: MOVB DUMMY,@ADST1 ;PRESET MUX TO DUMMY CHANNEL
1656 006606 012777 001610 172510 MOV #RETURN,@VECTOR ;RETURN ADDRESS
1657 006614 052777 000101 172474 BIS #101,@STREG ;CONVERSION ON DUMMY CHANNEL
1658 006622 000001 WAIT ;WAIT FOR INTERRUPT
1659 006624 017704 172472 MOV @ADBUFF,R4 ;DUMMY READ
1660 006630 013704 001366 MOV CHANL,R4
1661 006634 000304 SWAB R4
1662 006636 052704 000101 BIS #101,R4 ;INTERRUPT ENABLE START
1663 006642 010477 172450 MOV R4,@STREG ;JUMP TO CHANNEL + START CONVERT
1664 006646 000001 WAIT ;WAIT FOR INTERRUPT
1665 006650 027737 172446 001414 CMP @ADBUFF,EDGE
1666 006656 002001 BGE 2$
1667 006660 005200 INC R0 ;COUNT RESULTS .LT. EDGE
1668 006662 005301 2$: DEC R1
1669 006664 001345 BNE NXTCVT
1670 006666 020037 001426 CMP R0,PERCNT
1671 006672 003003 BGT SHIFT
1672 006674 163737 001416 001410 SUB BITPNT,DAC ;TAKE THE BIT OUT
1673 006702 006237 001416 SHIFT: ASR BITPNT
1674 006706 001315 BNE TRY
1675 006710 000205 RTS

```

```

1576
1677 006712 104400 013010
1678 006716 005037 001430
1679 006722 012700 017744
1680 006726 012701 010000
1681 006732 005020
1682 006734 005301
1683 006736 001375
1684 006740 012700 017124
1685 006744 012701 000310
1686 006750 005003
1687 006752 005037 001430
1688 006756 005037 001342
1689 006762 005037 001344
1690 006766 005037 001346
1691 006772 005037 001350
1692 006776 005020
1693 007000 005301
1694 007002 001375
1695 007004 012700 000011
1696 007010 063700 001336
1697 007014 000300
1698 007016 052700 000100
1699 007022 010077 172270
1700 007026 012700 001440
1701 007032 012777 001610 172264
1702 007040 012701 007776
1703 007044 004737 010710
1704 007050 013702 001372
1705 007054 042702 177760
1706 007060 001402
1707 007062 005302
1708 007064 001376
1709 007066 005277 172224
1710 007072 000001
1711 007074 017702 172222
1712 007100 001413
1713 007102 020227 007777
1714 007106 001413
1715 007110 006302
1716 007112 005262 017744
1717 007116 100013
1718 007120 012762 077777 017744
1719 007126 000407
1720 007130 020227 007777
1721 007134 001400
1722 007136 005201
1723 007140 005263 001352
1724 007144 100403
1725 007146 005301
1726 007150 001335
1727 007152 000403
1728 007154 005037 001352
1729 007160 000772
1730 007162 005300
1731 007164 001325

;:DIFFERENTIAL LINEARITY SUBROUTINE;:
DIFLIN: TYPE MSG20
CLR OUT
MOV #BUFFER,RO
MOV #4096.,R1 ;4096 WORDS FOR HISTOGRAM
CLEAR1: CLR (RO)+ ;CLEAR BUFFER AREA
DEC R1
BNE CLEAR1
MOV #DIST,RO ;DISTRIBUTION BUFFER POINTER
MOV #200.,R1 ;200. WORDS FOR DISTRIBUTION
CLR R3
CLR OUT
CLR WIDE
CLR NARROW
CLR FIRST
CLR SKIPST
CLEAR2: CLR (RO)+ ;CLEAR DISTRIBUTION BUFFER AREA
DEC R1
BNE CLEAR2
CHANNL: MOV #11,RO ;CHANNEL 11
ADD BASECH,RO
SWAB RO ;LOAD MUX BITS
BIS #100,RO
MOV RO,ISTREG
MOV #800.,RO ;NOMINAL STATE WIDTH - 1 LSB
MOV #RETURN,VECTOR
AGAIN: MOV #4094.,R1
NEXT: JSR PC,RANDY ;GET RANDOM NUMBER
MOV RMA,R2
BIC #177760,R2 ;MASK IT TO 4 BITS ONLY
BEQ CONVR
DELAY3: DEC R2 ;STALL
BNE DELAY3 ;TIME
CONVR: INC ISTREG ;START CONVERSION
WAIT
MOV JADBUFF,R2 ;GET CONVERTED VALUE
BEQ DELAY1 ;IGNORE IF =0
CMP R2,#7777 ;IGNORE IF =7777
BEQ DELAY2
ASL R2
INC BUFFER(R2) ;MAKE HISTOGRAM
BPL OKAY
MOV #077777,BUFFER(R2) ;PREVENT OVERFLOW
BR OKAY
DELAY1: CMP R2,#7777 ;EQUALIZE LOOP TIME
BEQ DELAY2 ;WITH DUMMY INSTR.
DELAY2: INC R1
INC TEMP(R3)
OKAY: DEC R1
BNE NEXT
BR AROUND
NOTOK: CLR TEMP
BR OKAY
AROUND: DEC RO
BNE AGAIN

```

1732	007166	012700	007776	MOV	#4094, R0	
1733	007172	012701	017746	MOV	#BUFFER+2, R1	
1734	007176	012102		READ: MOV	(R1)+, R2	;GET STATE WIDTH
1735	007200	006202		ASR	R2	;1 LSB = 800.
1736	007202	006202		ASR	R2	
1737	007204	006202		ASR	R2	
1738	007206	005502		ADC	R2	;1 LSB = 100.
1739	007210	020227	000310	CMP	R2, #200.	;OUT OF RANGE?
1740	007214	002403		BLT	INRNGE	
1741	007216	005237	001430	INC	OUT	;YES - INCREMENT COUNTER
1742	007222	000423		BR	TYPBAD	
1743	007224	006302		INRNGE: ASL	R2	
1744	007226	005262	017124	INC	DIST(R2)	;MAKE STATE WIDTH DISTRIBUTION
1745	007232	006202		ASR	R2	
1746	007234	020227	000062	CMP	R2, #50.	;IS IT 1/2 LSB?
1747	007240	002007		BGE	NOTNAR	
1748	007242	005237	001344	INC	NARROW	
1749	007246	005702		TST	R2	;IS IT A SKIPPED STATE?
1750	007250	001002		BNE	3IS	
1751	007252	005237	001350	INC	SKIPST	
1752	007256	000405		BR	TYPBAD	
1753	007260	020227	000226	NOTNAR: CMP	R2, #150.	;IS IT 1.5 LSB?
1754	007264	003426		BLE	LAST	
1755	007266	005237	001342	INC	WIDE	
1756	007272	005737	001346	TYPBAD: TST	FIRST	
1757	007276	001004		BNE	6OS	
1758	007300	005237	001346	INC	FIRST	
1759	007304	104400	012167	TYPE	STATE	
1760	007310	010103		6OS: MOV	R1, R3	
1761	007312	162703	017746	SUB	#BUFFER+2, R3	
1762	007316	006203		ASR	R3	
1763	007320	010346		MOV	R3, -(SP)	::SAVE R3 FOR TYPEOUT
1764						::TYPE STATE
1765	007322	104402		TYPOS		::GO TYPE--OCTAL ASCII
1766	007324	004		.BYTE	4	::TYPE 4 DIGIT(S)
1767	007325	001		.BYTE	1	::TYPE LEADING ZEROS
1768	007326	104400	012163	TYPE	DASH	
1769	007332	004737	011374	JSR	PC, DECTYP	
1770	007336	104400	012154	TYPE	LSBMSG	
1771	007342	005300		LAST: DEC	R0	
1772	007344	001314		BNE	READ	
1773	007346	112737	000177	MOV	#177, DECPNT	
1774	007354	013702	001350	MOV	SKIPST, R2	;GET NO. OF SKIPPED STATES
1775	007360	004737	011374	JSR	PC, DECTYP	;TYPE IT
1776	007364	104400	012412	TYPE	SKPMSG	;TYPE MESSAGE
1777	007370	005737	001350	TST	SKIPST	
1778	007374	001403		BEQ	IS	
1779	007376	104400	012375	TYPE	ERMSG	;TYPE "ERROR"
1780	007402	000402		BR	NAR	
1781	007404	104400	012364	IS: TYPE	,OKMSG	;TYPE #OK#


```

1825          ;CHANGE HISTOGRAM ERROR TO RELATIVE ACCURACY ERROR
1826
1827 007636 005001 RELACC: CLR R1 ;RUNNING ERROR = 0
1828 007640 005003 CLR R3 ;MAXIMUM ERROR = 0
1829 007642 104400 013435 TYPE MSG21
1830 007646 012700 017746 MOV #BUFFER+2,R0
1831 007652 011002 NXTSTA: MOV (R0),R2 ;STATE WIDTH = R2
1832 007654 162702 001440 SUB #800.,R2 ;STATE WIDTH ERROR IN R2
1833 007660 060201 ADD R2,R1 ;UPDATE RUNNING ERROR
1834 007662 010120 MOV R1,(R0)+ ;SAVE IN BUFFER
1835 007664 010104 MOV R1,R4 ;SAVE IN R4 ALSO
1836 007666 100001 BPL PLUS ;IS IT POSITIVE?
1837 007670 005404 NEG R4 ;NO - MAKE IT POSITIVE
1838 007672 020403 PLUS: CMP R4,R3 ;CHECK AGAINST PREVIOUS MAX. ERROR
1839 007674 003405 BLE NOTNEW ;NOT A NEW MAXIMUM
1840 007676 010403 MOV R4,R3 ;UPDATE MAXIMUM IN R3
1841 007700 010005 MOV R0,R5
1842 007702 162705 017746 SUB #BUFFER+2,R5
1843 007706 006205 ASR R5 ;R5=EDGE VALUE AT MAX. RELACC
1844 007710 020027 037742 NOTNEW: CMP R0,#BUFFER+8190. ;DONE?
1845 007714 001356 BNE NXTSTA ;NO - REPEAT
1846 007716 006203 ASR R3 ;RESCALE FROM 1 LSB = 800. SCALING
1847 007720 006203 ASR R3 ;TO 1 LSB = 100. SCALING
1848 007722 006203 ASR R3
1849 007724 005503 ADC R3
1850 007726 010302 MOV R3,R2
1851 007730 004737 011374 JSR PC,DECTYP
1852 007734 104400 013462 TYPE LINEA
1853 007740 010546 MOV R5,-(SP) ;;SAVE R5 FOR TYPEOUT
1854 ;;TYPE VALUE
1855 007742 104402 TYPOS ;;GO TYPE--OCTAL ASCII
1856 007744 004 .BYTE 4 ;;TYPE 4 DIGIT(S)
1857 007745 001 .BYTE 1 ;;TYPE LEADING ZEROS
1858 007746 104400 012321 TYPE SLASH ;PRINT '/'
1859 007752 005205 INC R5
1860 007754 010546 MOV R5,-(SP) ;;SAVE R5 FOR TYPEOUT
1861 ;;TYPE VALUE
1862 007756 104402 TYPOS ;;GO TYPE--OCTAL ASCII
1863 007760 004 .BYTE 4 ;;TYPE 4 DIGIT(S)
1864 007761 001 .BYTE 1 ;;TYPE LEADING ZEROS
1865 007762 020337 011640 CMP R3,VLIN
1866 007766 003403 BLE 41$
1867 007770 104400 012375 TYPE ERMSG
1868 007774 000402 BR 42$
1869 007776 104400 012364 41$: TYPE OKMSG
1870 010002 005737 001404 42$: TST FLAG ;VT55?
1871 010006 001503 BEQ L02
1872 010010 012700 017744 MOV #BUFFER,R0
1873 010014 012701 010000 MOV #4096.,R1

```

1874	010020	011002		GETDAT: MOV	(R0),R2		
1875	010022	006202		ASR	R2		;GET RELATIVE ACCURACY ERROR SCALED 1LSB = 200.
1876	010024	006202		ASR	R2		;RESCALE IT TO 1 LSB = 100.
1877	010026	006202		ASR	R2		
1878	010030	005502		ADC	R2		
1879	010032	062702	000166	ADD	#118, R2		;AND MOVE IT TO MID-SCREEN
1880	010036	010220		MOV	R2,(R0)+		;PUT IT BACK INTO BUFFER
1881	010040	005301		DEC	R1		
1882	010042	001366		BNE	GETDAT		
1883	010044	012700	017744	MOV	#BUFFER,R0		
1884	010050	012704	017744	MOV	#BUFFER,R4		
1885	010054	012705	017746	MOV	#BUFFER+2,R5		
1886	010060	012701	001000	MOV	#512, R1		
1887	010064	012702	000007	NXTB: MOV	#7, R2		
1888	010070	012003		MOV	(R0)+, R3		
1889	010072	010337	001420	MOV	R3,MIN		;MINIMUM
1890	010076	010337	001424	MOV	R3,MAX		;MAXIMUM
1891	010102	012003		NXTCMP: MOV	(R0)+, R3		
1892	010104	020337	001420	CMP	R3,MIN		
1893	010110	002002		BGE	MAXTST		
1894	010112	010337	001420	MOV	R3,MIN		;NEW MINIMUM
1895	010116	020337	001424	MAXTST: CMP	R3,MAX		
1896	010122	003402		BLE	TSTB		
1897	010124	010337	001424	MOV	R3,MAX		;NEW MAXIMUM
1898	010130	005302		TSTB: DEC	R2		
1899	010132	001363		BNE	NXTCMP		
1900	010134	013724	001420	MOV	MIN,(R4)+		
1901	010140	013725	001424	MOV	MAX,(R5)+		
1902	010144	022425		CMP	(R4)+,(R5)+		;BUMP EACH ONCE MORE
1903	010146	005301		DEC	R1		
1904	010150	001345		BNE	NXTB		
1905	010152	104400	012750	TYPE	,MSG18		
1906	010156	104400	013671	TYPE	,BUFF2		;TYPE BUFF2
1907	010162	012700	017744	MOV	#BUFFER,R0		
1908	010166	004737	010220	JSR	PC,LOAD		
1909	010172	104400	013571	TYPE	,C3		;TYPE ASCIZ STRING
1910	010176	012700	017746	MOV	#BUFFER+2,R0		
1911	010202	004737	010220	JSR	PC,LOAD		
1912	010206	104400	013566	TYPE	,C2		;TYPE ASCIZ STRING
1913	010212	004737	010242	JSR	PC,DELCLR		
1914	010216	000207		L02: RTS	PC		
1915	010220	012701	001000	LOAD: MOV	#512, R1		
1916	010224	012002		LOAD0: MOV	(R0)+,R2		
1917	010226	005720		TST	(R0)+		
1918	010230	004737	011272	JSR	PC,LOADY		
1919	010234	005301		DEC	R1		
1920	010236	001372		BNE	LOAD0		
1921	010240	000207		RTS	PC		

```

1922 010242 005000          DELCLR: CLR      RO
1923 010244 012701 000020      MOV      #20,R1      ;DELAY BEFORE CLEANING SCREEN
1924 010250 005300          1$:      DEC      RO
1925 010252 001376          BNE      1$
1926 010254 005301          DEC      R1
1927 010256 001374          BNE      1$
1928 010260 032777 010000 170652  BIT      #BIT12,0SWR ;TEST FOR HALT FOR DISPLAY
1929 010266 001401          BEQ      2$          ;;DON'T HALT FOR DISPLAY
1930 010270 000000          HALT
1931 010272 104400 013711      2$:      TYPE     VTINIT
1932 010276 000207          RTS      PC
1933                                     ;;NOISE SUBROUTINE;;
1934 010300 013537 001366          NOITST: MOV      @R5)+,CHANL ;LOAD CHANNEL
1935 010304 013737 001366 001364      MOV      CHANL,DUMMY ;LOAD DUMMY CHANNEL
1936 010312 004737 006324          JSR      PC,GETEDG ;GET EDGE VALUE
1937 010316 004737 010472          JSR      PC,NOIA ;GET RMS AND PEAK VALUES
1938 010322 012737 000001 006502      MOV      #1,EDGFLG
1939 010330 004737 010336          JSR      PC,TYPRP ;TYPE RMS AND PEAK VALUES
1940 010334 000205          RTS      RS
1941
1942
1943
1944
1945
1946                                     ;;TYPE RMS AND PEAK VALUES;;
1947 010336 104400 012261          TYPRP: TYPE     NOI
1948 010342 005737 001400          TST      RMS
1949 010346 100002          BPL      POSRMS
1950 010350 005037 001400          CLR      RMS ;RMS<0,SET RMS=0
1951 010354 005737 001402          POSRMS: TST      PEAK
1952 010360 100002          BPL      POSPEA
1953 010362 005037 001402          CLR      PEAK ;PEAK<0,SET PEAK=0
1954 010366 013702 001400          POSPEA: MOV      RMS,R2
1955 010372 004737 011374          JSR      PC,DECTYP
1956 010376 104400 012634          TYPE     MESR
1957 010402 013702 001402          MOV      PEAK,R2
1958 010406 004737 011374          JSR      PC,DECTYP
1959 010412 104400 012647          TYPE     MESP
1960 010416 004737 006440          JSR      PC,TYPEDG
1961 010422 104400 012271          TYPE     CHAN
1962 010426 013746 001366          MOV      CHANL,-(SP) ;SAVE CHANL FOR TYPEOUT
1963                                     ;TYPE CHANL
1964 010432 104402          TYPOS   ;GO TYPE--OCTAL ASCII
1965 010434 002          .BYTE  2 ;TYPE 2 DIGIT(S)
1966 010435 000          .BYTE  0 ;SUPPRESS LEADING ZEROS
1967 010436 023737 001400 011632      CMP      RMS,VNR ;WITHIN LIMITS?
1968 010444 003007          BGT      ER
1969 010446 023737 001402 011634      CMP      PEAK,VNP ;WITHIN LIMITS?
1970 010454 003003          BGT      ER
1971 010456 104400 012364          TYPE     OKMSG
1972 010462 000207          RTS      PC
1973 010464 104400 012375          ER:     TYPE     ERMSG
1974 010470 000207          RTS      PC

```

```

1975                                     ::SUBROUTINES FOR NOISE TEST;;
1976 010472 005037 001400 NOIA: CLR RMS ;CLEAR RMS VLAUE
1977 010476 005037 001402 CLR PEAK ;CLEAR PEAK VALUE
1978 010502 004537 006504 NOI1: JSR R5,SARSUB ;DO SAR ROUTINE AT 16%
1979 010506 000020 16.
1980 010510 063737 001410 001400 ADD DAC,RMS ;ADD RESULT TO RMS
1981 010516 004537 006504 JSR R5,SARSUB ;DO SAR ROUTINE AT 84%
1982 010522 000124 84.
1983 010524 163737 001410 001400 SUB DAC,RMS ;SUBTRACT RESULT FROM RMS
1984 010532 004537 006504 JSR R5,SARSUB ;DO SAR ROUTINE AT 1%
1985 010536 000001 1
1986 010540 063737 001410 001402 ADD DAC,PEAK ;ADD RESULT TO PEAK
1987 010546 004537 006504 JSR R5,SARSUB ;DO SAR ROUTINE AT 99%
1988 010552 000143 99.
1989 010554 163737 001410 001402 SUB DAC,PEAK ;SUBTRACT RESULT FROM PEAK
1990 010562 000207 RTS PC ;RETURN
1991
1992 010564 012537 001366 NOI8: MOV (R5)+,CHANL ;GET CHANNEL VALUE
1993 010570 063737 001336 001366 ADD BASECH,CHANL
1994 010576 013737 001366 001364 MOV CHANL,DUMMY ;LOAD DUMMY CHANNEL
1995 010604 004737 006324 JSR PC,GETEDG ;GET EDGE VALUES
1996 010610 005037 001400 CLR RMS ;CLEAR RMS VALUE
1997 010614 005037 001402 CLR PEAK ;CLEAR PEAK VALUE
1998 010620 012737 000010 010706 MOV #10,10$ ;SET UP COUNTER
1999 010626 004737 010502 1$: JSR PC,NOI1 ;GET NOISE VALUES
2000 010632 005237 001414 INC EDGE
2001 010636 005337 010706 DEC 10$
2002 010642 001371 BNE 1$ ;REPEAT 8 TIMES
2003 010644 162737 000010 001414 SUB #10,EDGE
2004 010652 006237 001400 ASR RMS ;SCALE IT TO 1 LSB=100.
2005 010656 005537 001400 ADC RMS
2006 010662 006237 001402 ASR PEAK
2007 010666 005537 001402 ADC PEAK
2008 010672 012737 000010 006502 MOV #8,EDGFLG
2009 010700 004737 010336 JSR PC,TYPRP ;TYPE RESULTS
2010 010704 000205 RTS R5 ;RETURN
2011 010706 000000 10$: 0 ;COUNTER
2012
2013
2014 ::RANDOM NUMBER GENERATOR;;
2015 010710 063737 001374 001372 RANDY: ADD RNB,RNA
2016 010716 063737 001376 001372 ADD RNC,RNA
2017 010724 005537 001372 ADC RNA
2018 010730 063737 001372 001374 ADD RNA,RNB
2019 010736 063737 001376 001374 ADD RNC,RNB
2020 010744 005537 001374 ADC RNB
2021 010750 063737 001372 001376 ADD RNA,RNC
2022 010756 063737 001374 001376 ADD RNB,RNC
2023 010764 005537 001376 ADC RNC
2024 010770 000207 RTS PC

```

```

2025      ;:ROUTINE TO AVERAGE 8 CONVERSIONS;;
2026 010772 012500      CONVRT: MOV      (R5)+,RO      ;GET CHANNEL VALUE
2027 010774 063700 001336      ADD      BASECH,RO
2028 011000 010037 00136E      MOV      RO,CHANL
2029 011004 000300      SWAB     RO
2030 011006 005037 001352      CLR      TEMP
2031 011012 010077 170300      MOV      RO,@STREG      ;LOAD CHANNEL INTO MIX BITS
2032 011016 012700 010000      MOV      #10000,RO
2033 011022 005300      2$: DEC      RO
2034 011024 001376      BNE     2$
2035 011026 012777 001610 170270      MOV      @RETURN,@VECTOR      ;LOAD VECTOR
2036 011034 012700 000010      MOV      #10,RO      ;SET UP COUNTER
2037 011040 152777 000101 170250      1$: BISB     #101,@STREG      ;SET INTRPT. EN., START CONV.
2038 011046 000001      WAIT     ;WAIT FOR CONVERSION
2039 011050 067737 170246 001352      ADD      @ADBUFF,TEMP      ;READ BUFFER
2040 011056 005300      DEC      RO
2041 011060 001367      BNE     1$      ;DO 8 TIMES
2042 011062 006237 001352      ASR     TEMP      ;AVERAGE VALUE
2043 011066 006237 001352      ASR     TEMP
2044 011072 006237 001352      ASR     TEMP
2045 011076 005537 001352      ADC     TEMP
2046 011102 000205      RTS      ;RETURN
2047
2048      ;COMPARE $CDDAT AND $BDDAT:;
2049 011104 012537 001124      COMPAR: MOV      (R5)+,$CDDAT      ;GET GOOD DATA
2050 011110 013537 001406      MOV      @($R5)+,$SPREAD      ;GET SPREAD
2051 011114 013737 001352 001126      MOV      TEMP,$BDDAT      ;GET BAD(ACTUAL) DATA
2052 011122 013701 001126      MOV      $BDDAT,R1
2053 011126 013700 001124      MOV      $CDDAT,RO
2054 011132 160100      SUB     R1,RO      ;GET DIFFERENCE
2055 011134 100001      BPL     7$
2056 011136 005400      NEG     RO
2057 011140 020037 001406      7$: CMP     RO,SPREAD      ;COMPARE IT TO SPREAD
2058 011144 003001      BGT     10$      ;GO TO ERROR PRINTOUT
2059 011146 005725      TST     (R5)+      ;BUMP RETURN POINTER AROUND ERROR CALL
2060 011150 000205      10$: RTS     R5

```

```

2061                                     ;;SUBROUTINE TO TYPE INTRPT. TST MSG.;;
2062 011152 005737 001202      DUMW:  TST  $PASS
2063 011156 001021              BNE   20$
2064 011160 012737 011222 001110  MOV  #20$, $LPERR
2065 011166 012737 011222 001106  MOV  #20$, $LPADR
2066 011174 104400 013621      TYPE  METST
2067 011200 010046              MOV  RO, -(SP)
2068                                     ;; TYPE ASCIZ STRING
2069 011202 104402              TYPOS
2070 011204 002                .BYTE 2
2071 011205 000                .BYTE 0
2072 011206 104400 012711      TYPE  ONAD
2073 011212 013746 001316      MOV  $STREG, -(SP)
2074                                     ;; SAVE STREG FOR TYPEOUT
2075 011216 104402              TYPOS
2076 011220 006                .BYTE 6
2077 011221 001                .BYTE 1
2078 011222 000207              20$:  RTS  PC
2079                                     ;; TYPE BUS ADDRESS
2080 011224 005737 001202      DUMC:  TST  $PASS
2081 011230 001010              BNE   30$
2082 011232 012737 011252 001110  MOV  #30$, $LPERR
2083 011240 012737 011252 001106  MOV  #30$, $LPADR
2084 011246 104400 012306      TYPE  DONE
2085 011252 000207              30$:  RTS  PC

```

```

;; SAVE RO FOR TYPEOUT
;; TYPE TEST M^
;; GO TYPE--OCTAL ASCII
;; TYPE 2 DIGIT(S)
;; SUPPRESS LEADING ZEROS
;; SAVE STREG FOR TYPEOUT
;; TYPE BUS ADDRESS
;; GO TYPE--OCTAL ASCII
;; TYPE 6 DIGITS
;; TYPE LEADING ZEROS

```

```

2086                                     ;SUBROUTINE TO RESET & SET INTRPT. EN.;
2087 011254 000005 RST: RESET
2088 011256 052777 000100 167660 BIS #100, @STKS
2089 011264 005037 177776 CLR PSW
2090 011270 000207 RTS PC
2091
2092                                     ;SUBROUTINE LOADY:
2093 011272 005702 LOADY: TST R2 ;ROUTINE TO LOAD VLAUE INTO R2
2094 011274 100001 BPL PLUSR2 ;AS A VT55 Y-VALUE
2095 011276 005002 CLR R2
2096 011300 020227 000353 PLUSR2: CMP R2, #235.
2097 011304 002402 BLT LESS
2098 011306 012702 000353 MOV #235., R2
2099 011312 010203 LESS: MOV R2, R3
2100 011314 042702 177740 BIC #177740, R2
2101 011320 052702 000340 BIS #40, R2
2102 011324 105777 167620 B10: TSTB @STPS ;PRINT CHARACTER
2103 011330 100375 BPL B10
2104 011332 110277 167614 MOVB R2, @STPB
2105 011336 006203 ASR R3
2106 011340 006203 ASR R3
2107 011342 006203 ASR R3
2108 011344 006203 ASR R3
2109 011346 006203 ASR R3
2110 011350 042703 177770 BIC #177770, R3
2111 011354 052703 000040 BIS #40, R3
2112 011360 105777 167564 B11: TSTB @STPS ;PRINT CHARACTER
2113 011364 100375 BPL B11
2114 011366 110377 167560 MOVB R3, @STPB
2115 011372 000207 RTS PC
2116
2117

```

```

2118      ;;SUBROUTINE TO TYPE DECIMAL VALUE;;
2119      ;;IN R2 AS X.XX;;
2120 011374 005702      DECTYP: TST      R2      ;TEST VALUE TO BE TYPED
2121 011376 100003      BPL      POS
2122 011400 104400 012123      TYPE      MINUS      ;TYPE MINUS SIGN
2123 011404 005402      NEG      R2
2124 011406 020227 001747      POS:  CMP      R2,#999.      ;>999. REPLACE IT WITH 999.
2125 011412 003402      BLE      OKAYD
2126 011414 012702 001747      MOV      #999.,R2
2127 011420 105037 014455      OKAYD: CLRB     ONES      ;CLEAR ONES
2128 011424 105037 014454      CLRB     TENS      ;CLEAR TENS
2129 011430 105037 014452      CLRB     HUNS      ;CLEAR HUNS
2130 011434 005702      TESTR2: TST      R2      ;CONVERT VALUE TO A DECIMAL VALUE
2131 011436 001424      BEQ      TYP0UT
2132 011440 005302      DEC      R2
2133 011442 105237 014455      INCB     ONES
2134 011446 123727 014455 000012      CMPB     ONES,#10.
2135 011454 001367      BNE      TESTR2
2136 011456 105037 014455      CLRB     ONES
2137 011462 105237 014454      INCB     TENS
2138 011466 123727 014454 000012      CMPB     TENS,#10.
2139 011474 001357      BNE      TESTR2
2140 011476 105037 014454      CLRB     TENS
2141 011502 105237 014452      INCB     HUNS
2142 011506 000752      BR      TESTR2
2143 011510 152737 000060 014452      TYP0UT: BISB     #60,HUNS      ;PREPARE FOR TYP0UT
2144 011516 152737 000060 014454      BISB     #50,TENS
2145 011524 152737 000060 014455      BISB     #60,ONES
2146 011532 104400 014452      TYPE     HUNS      ;TYPE VALUE
2147 011536 000207      RTS      PC
2148
2149 011540 012701 011632      WFADJ: MOV      #VNR,R1      ;SUBROUTINE TO SET UP LIMITS
2150 011544 005737 001336      TST      BASECH      ;TESTING AN AM11K?
2151 011550 001403      BEQ      1$      ;;
2152 011552 012702 011664      MOV      #VARLT3,R2      ;BASECH NOT ZERO, USE AM11K LIMITS
2153 011556 000410      BR      3$      ;;
2154 011560 005737 001422      1$:  TST      WFTST
2155 011564 001003      BNE      2$
2156 011566 012702 011644      MOV      #VARLT1,R2      ;WFTST=0,USE NORMAL LIMITS
2157 011572 000402      BR      3$
2158 011574 012702 011654      2$:  MOV      #VARLT2,R2      ;WFTST=1,USE OPTION AREA LIMITS
2159 011600 012221      3$:  MOV      (R2)+,(R1)+
2160 011602 005711      TST      (R1)
2161 011604 100375      BPL      3$
2162 011606 000207      RTS      PC

```


.SBTTL END OF PASS ROUTINE

*INCREMENT THE PASS NUMBER (\$PASS)
*TYPE "END PASS"
*IF THERES A MONITOR GO TO IT
*IF THERE ISN'T JUMP TO AGATST
*IF IT IS DESIRED TO HAVE A BELL INDICATE THE "END OF PASS" LOCATION
*\$SENDMG CAN BE CHANGED TO 7.

2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207 011710
2208 011710 000240
2209 011712 005037 001102
2210 011716 005037 001160
2211 011722 005237 001202
2212 011726 042737 100000 001202
2213 011734 005327
2214 011736 000001
2215 011740 003015
2216 011742 012737
2217 011744 000001
2218 011746 011736
2219 011750 104400 012003
2220 011754 013700 000042
2221 011760 001405
2222 011762 000005
2223 011764 004710
2224 011766 000240
2225 011770 000240
2226 011772 000240
2227 011774
2228 011774 000137
2229 011776 011674
2230 012000 377 377 000
2231 012003 015 042412 042116
2232 012010 050040 051501 000123
2233

\$EOP: NOP
CLR \$STNM ;; ZERO THE TEST NUMBER
CLR \$TIMES ;; ZERO THE NUMBER OF ITERATIONS
INC \$PASS ;; INCREMENT THE PASS NUMBER
BIC #100000,\$PASS ;; DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;; LOOP?
\$EOPCT: .WORD 1
BGT \$DOAGN ;; YES
MOV (PC)+,2(PC)+ ;; RESTORE COUNTER
\$ENDCT: .WORD 1
TYPE \$SENDMG ;; TYPE "END PASS"
\$GET42: MOV 2#42,RO ;; GET MONITOR ADDRESS
BEQ \$DOAGN ;; BRANCH IF NO MONITOR
RESET ;; CLEAR THE WORLD
\$ENDAD: JSR PC,(RO) ;; GO TO MONITOR
NOP ;; SAVE ROOM
NOP ;; FOR
NOP ;; ACT11
\$DOAGN: JMP 2(PC)+ ;; RETURN
\$RTNAD: .WORD AGATST
\$ENULL: .BYTE -1,-1,0 ;; NULL CHARACTER STRING
\$SENDMG: .ASCIZ <15><12>/END PASS/

2234				
2235	012016	005015	047516	051511
2236	012024	020105	042524	052123
2237	012032	026455	000040	
2238	012036	005015	042523	052124
2239	012044	044514	043516	052040
2240	012052	051505	026524	020055
2241	012060	054524	042520	042040
2242	012066	051505	051111	042105
2243	012074	023440	051106	046517
2244	012102	020047	044103	047101
2245	012110	042516	020114	020046
2246	012116	051103	020072	000
2247	012123	055	000	
2248	012125	077	000	
2249	012127	136	101	040
2250	012132	040	000	
2251	012134	136	103	040
2252	012137	040	000	
2253	012141	136	107	015
2254	012144	012	123	127
2255	012147	122	105	107
2256	012152	072	000	
2257	012154	046040	041123	005015
2258	012162	000		
2259	012163	055	020055	000
2260	012167	123	040524	042524
2261	012174	026455	053440	042111
2262	012202	044124	005015	000
2263	012207	103	000110	
2264	012212	020040	020040	000
2265	012217	040	051514	020102
2266	012224	047117	041440	000110
2267	012232	051440	052105	046124
2268	012240	047111	020107	051106
2269	012246	046517	041440	000110
2270	012254	040440	020124	000
2271	012261	116	044517	042523
2272	012266	020072	000	
2273	012271	040	047117	041440
2274	012276	040510	047116	046105
2275	012304	000040		
2276	012306	020040	020040	047504
2277	012314	042516	005015	000
2278	012321	057	000	
2279	012323	124	050131	020105
2280	012330	042504	044523	042522
2281	012336	020104	052047	023517
2282	012344	041440	040510	047116
2283	012352	046105	023040	041440
2284	012360	035122	000040	
2285	012364	020040	020040	045517
2286	012372	005015	000	

.SBTTL ASCII MESSAGES

NOIMSG: .ASCIZ <15><12>/NOISE TEST-- /

SETMSG: .ASCIZ <15><12>/SETTLING TEST-- TYPE DESIPEO 'FROM' CHANNEL & CR: /

MINUS: .BYTE 55,0
QUEST: .BYTE 77,0
AMSG: .BYTE 136,101,40,40,0

CMSG: .BYTE 136,103,40,40,0

GMSG: .BYTE 136,107,15,12,123,127,122,105,107,72,0

LSBMSG: .ASCIZ / LSB/<15><12>

DASH: .ASCIZ /-- /
STATE: .ASCIZ /STATE-- WIDTH/<15><12>

CH: .ASCIZ /CH/
SPACE: .ASCIZ / /
LSB: .ASCIZ / LSB ON CH/

SETCH: .ASCIZ / SETTLING FROM CH/

ATMSG: .ASCIZ / AT /
NOI: .ASCIZ /NOISE: /

CHAN: .ASCIZ / ON CHANNEL /

DONE: .ASCIZ / DONE/<15><12>

SLASH: .ASCIZ #/#
TOMSG: .ASCIZ /TYPE DESIRED 'TO' CHANNEL & CR: /

OKMSG: .ASCIZ / OK/<15><12>

E05

MD-11-DZADL-A
DZADLA.P11

MACY11 27(732)
ASCII MESSAGES

25-SEP-76 10:38 PAGE 57

2395	013476	040440	020124	000	
2396	013503	015	041412	046101	HEADS: .ASCII <15><12>/CALIBRATION--/
2397	013510	041111	040522	044524	
2398	013516	047117	026455		
2399	013522	051440	052105	041440	ASKCH: .ASCIZ . SET CHANNEL IN SWR LOW BYTE/<15><12>
2400	013530	040510	047116	046105	
2401	013536	044440	020116	053523	
2402	013544	020122	047514	020127	
2403	013552	054502	042524	005015	
2404	013560	000			
2405	013561	033	000132		C0: .ASCIZ <33><132>
2406	013564	000055			C1: .ASCIZ <55>
2407	013566	031033	000		C2: .ASCIZ <33><62>
2408	013571	112	000		C3: .ASCIZ <112>
2409	013573	015	047412	043106	MOFSET: .ASCIZ <15><12>/OFFSET =/
2410	013600	042523	020124	000075	
2411	013606	046040	041123	000040	MLSB: .ASCIZ / LSB /
2412	013614	040440	020124	000	MAT: .ASCIZ / AT /
2413	013621	015	020012	047105	METST: .ASCIZ <15><12>/ ENTERING TEST /
2414	013626	042524	044522	043516	
2415	013634	052040	051505	020124	
2416	013642	000			
2417	013643	033	061	101	BUFF1: .BYTE 33,61,101,61,111,62,114,41,60,45,63,51,66,55,71,61,74,110,41,40,112,0
2418	013646	061	111	062	
2419	013651	114	041	060	
2420	013654	045	063	051	
2421	013657	066	055	071	
2422	013662	061	074	110	
2423	013665	041	040	112	
2424	013670	000			
2425	013671	033	061	101	BUFF2: .BYTE 33,61,101,47,111,61,104,50,65,44,62,110,40,40,102,0
2426	013674	047	111	061	
2427	013677	104	050	065	
2428	013702	044	062	110	
2429	013705	040	040	102	
2430	013710	000			
2431	013711	033	110	033	VTINIT: .BYTE 33,110,33,112,33,61,101,40,33,62,0
2432	013714	112	033	061	
2433	013717	101	040	033	
2434	013722	062	000		
2435	013724	005015	046412	026504	HEAD1: .ASCII <15><12><12>/MD-11-DZADL-A AD11K DIAGNOSTIC/<15><12>
2436	013732	030461	042055	040532	
2437	013740	046104	040455	020040	
2438	013746	020040	042101	030461	
2439	013754	020113	044504	043501	
2440	013762	047516	052123	041511	
2441	013770	005015			
2442	013772	040412	020072	052501	.ASCII <12>/A: AUTO TEST/
2443	014000	047524	052040	051505	
2444	014006	124			
2445	014007	015	041412	020072	.ASCII <15><12>/C: CALIBRATION/
2446	014014	040503	044514	051102	
2447	014022	052101	047511	116	
2448	014027	015	046012	020072	.ASCII <15><12>/L: LOGIC TEST/
2449	014034	047514	044507	020103	
2450	014042	042524	052123		

F05

MAINDEC-11-DZADL-A
DZADLA.P11

MACY11 27(732)
ASCII MESSAGES

25-SEP-76 10:38 PAGE 58

2451	014046	005015	035116	047040
2452	014054	044517	042523	052040
2453	014062	051505	124	

.ASCII <15<12>/N: NOISE TEST/

2454	014065	015	051412	020072		.ASCII <15><12>/S: SETTLE TEST/
2455	014072	042523	052124	042514		
2456	014100	052040	051505	124		
2457	014105	015	053412	020072		.ASCIZ <15><12>/W: WRAPAROUND TEST/<15><12>
2458	014112	051127	050101	051101		
2459	014120	052517	042116	052040		
2460	014126	051505	006524	000012		
2461	014134	005015	052123	052101	EM1:	.ASCIZ <15><12>/STATUS REG. ERROR/<15><12>
2462	014142	051525	051040	043505		
2463	014150	020056	051105	047522		
2464	014156	006522	000012			
2465	014162	005015	040506	046111	EM2:	.ASCIZ <15><12>/FAILED TO INTERRUPT/<15><12>
2466	014170	042105	052040	020117		
2467	014176	047111	042524	051122		
2468	014204	050125	006524	000012		
2469	014212	005015	047125	054105	EM3:	.ASCIZ <15><12>/UNEXPECTED INTERRUPT/<15><12>
2470	014220	042520	052103	042105		
2471	014226	044440	052116	051105		
2472	014234	052522	052120	005015		
2473	014242	000				
2474	014243	015	042412	051122	EM4:	.ASCIZ <15><12>#ERROR ON A/D CHANNEL#<15><12>
2475	014250	051117	047440	020116		
2476	014256	027501	020104	044103		
2477	014264	047101	042516	006514		
2478	014272	000012				
2479	014274	051105	050122	020103	DH1:	.ASCIZ /ERRPC STREG EXPECTED ACTUAL/<15><12>
2480	014302	052123	042522	020107		
2481	014310	054105	042520	052103		
2482	014316	042105	040440	052103		
2483	014324	040525	006514	000012		
2484	014332	051105	050122	020103	DH2:	.ASCIZ /ERRPC STREG CHANNEL NOMINAL TOLERANCE ACTUAL/
2485	014340	051440	051124	043505		
2486	014346	020040	041440	040510		
2487	014354	047116	046105	020040		
2488	014362	047516	044515	040516		
2489	014370	020114	052040	046117		
2490	014376	051105	047101	042503		
2491	014404	020040	041501	052524		
2492	014412	046101	000			
2493	014415	105	051122	041520	DH3:	.ASCIZ /ERRPC STREG ACTUAL/<15><12>
2494	014422	020040	020040	020040		
2495	014430	052123	042522	020107		
2496	014436	020040	040440	052103		
2497	014444	040525	006514	000012		

2498	014452	000				HUNS:	.BYTE	0
2499	014453	056				DECPNT:	.BYTE	56
2500	014454	000				TENS:	.BYTE	0
2501	014455	000	000			ONES:	.BYTE	0,0
2502		014460				.EVEN		
2503								
2504	014460	001116	001316	001124	DT1:	\$ERRPC, STREG, \$GDDAT, \$BDDAT, 0		
2505	014466	001126	000000					
2506	014472	001116	001316	001366	DT2:	\$ERRPC, STREG, CHANL, \$GDDAT, SPREAD, \$BDDAT, 0		
2507	014500	001124	001406	001126				
2508	014506	000000						
2509	014510	001116	001316	001126	DT3:	\$ERRPC, STREG, \$BDDAT, 0		
2510	014516	000000						
2511								
2512	014520	000000			DF1:	0		
2513								
2514								
2515								
2516								

.SBTTL TTY INPUT ROUTINE

.ENABL LSB

.DSABL LSB

;; THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY

;; CALL:

;; * RDCHR ;: INPUT A SINGLE CHARACTER FROM THE TTY
;; * RETURN HERE ;: CHARACTER IS ON THE STACK
;; * ;: WITH PARITY BIT STRIPPED OFF

\$RDCHR: MOV (SP), -(SP) ;: PUSH DOWN THE PC

MOV 4(SP), 2(SP) ;: SAVE THE PS

1\$: TSTB 2\$TKS ;: WAIT FOR

BPL 1\$;: A CHARACTER

MOVB 2\$TKB, 4(SP) ;: READ THE TTY

BIC #177, 4(SP) ;: GET RID OF JUNK IF ANY

CMP 4(SP), #23 ;: IS IT A CONTROL-S?

BNE 3\$;: BRANCH IF NO

2\$: TSTB 2\$TKS ;: WAIT FOR A CHARACTER

BPL 2\$;: LOOP UNTIL ITS THERE

MOVB 2\$TKB, -(SP) ;: GET CHARACTER

BIC #177, (SP) ;: MAKE IT 7-BIT ASCII

CMP (SP)+, #21 ;: IS IT A CONTROL-Q?

BNE 2\$;: IF NOT DISCARD IT

BR 1\$;: YES, RESUME

3\$: CMP 4(SP), #140 ;: IS IT UPPER CASE?

BLT 4\$;: BRANCH IF YES

CMP 4(SP), #175 ;: IS IT A SPECIAL CHAR?

BGT 4\$;: BRANCH IF YES

BIC #40, 4(SP) ;: MAKE IT UPPER CASE

4\$: RTI ;: GO BACK TO USER

;; THIS ROUTINE WILL INPUT A STRING FROM THE TTY

;; CALL:

;; * RDLIN ;: INPUT A STRING FROM THE TTY
;; * RETURN HERE ;: ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
;; * ;: TERMINATOR WILL BE A BYTE OF ALL 0'S

\$RDLIN: MOV R3, -(SP) ;: SAVE R3

1\$: MOV #177, R3 ;: GET ADDRESS

2\$: CMP #177, R3 ;: BUFFER FULL?

BLOS 4\$;: BR IF YES

RDCHR ;: GO READ ONE CHARACTER FROM THE TTY

MOVB (SP)+, (R3) ;: GET CHARACTER

10\$: CMPB #177, (R3) ;: IS IT A RUBOUT

BNE 3\$;: SKIP IF NOT

4\$: TYPE ' ' ;: TYPE A ' '

BR 1\$;: CLEAR THE BUFFER AND LOOP

3\$: MOVB (R3), 9\$;: ECHO THE CHARACTER

TYPE , 9\$

2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533 014522 011646
2534 014524 016666 000004 000002
2535 014532 105777 164406
2536 014536 100375
2537 014540 117766 164402 000004
2538 014546 042766 177600 000004
2539 014554 026627 000004 000023
2540 014562 001013
2541 014564 105777 164354
2542 014570 100375
2543 014572 117746 164350
2544 014576 042716 177600
2545 014602 022627 000021
2546 014606 001366
2547 014610 000750
2548 014612 026627 000004 000140
2549 014620 002407
2550 014622 026627 000004 000175
2551 014630 003003
2552 014632 042766 000040 000004
2553 014640 000002
2554
2555
2556
2557
2558
2559
2560
2561 014642 010346
2562 014644 012703 014750
2563 014650 022703 C:4760
2564 014654 101405
2565 014656 104404
2566 014660 112613
2567 014662 122713 000177
2568 014666 001003
2569 014670 104400 001170
2570 014674 000763
2571 014676 111337 014746
2572 014702 104400 014746

2573	014706	122723	000015			CMPB	#15,(R3)+	::CHECK FOR RETURN
2574	014712	001356				BNE	2\$::LOOP IF NOT RETURN
2575	014714	105063	177777			CLRB	-1(R3)	::CLEAR RETURN (THE 15)
2576	014720	104400	001172			TYPE	\$LF	::TYPE A LINE FEED
2577	014724	012603				MOV	(SP)+,R3	::RESTORE R3
2578	014726	011646				MOV	(SP)-,(SP)	::ADJUST THE STACK AND PUT ADDRESS OF THE
2579	014730	016666	000004	000002		MOV	4(SP),2(SP)	::FIRST ASCII CHARACTER ON IT
2580	014736	012766	014750	000004		MOV	#STTYIN,4(SP)	
2581	014744	000002				RTI		::RETURN
2582	014746	000			9\$:	.BYTE	0	::STORAGE FOR ASCII CHAR. TO TYPE
2583	014747	000				.BYTE	0	::TERMINATOR
2584	014750	000010			\$TTYIN:	.BLKB	8.	::RESERVE 8 BYTES FOR TTY INPUT
2585	014760	052536	005015	000	\$CNTLU:	.ASCIZ	/↑U/<15><12>	::CONTROL "U"
2586	014765	136	006507	000012	\$CNTLG:	.ASCIZ	/↑G/<15><12>	::CONTROL "G"
2587	014772	005015	053523	020122	\$MSWR:	.ASCIZ	<15><12>/SWR = /	
2588	015000	020075	000					
2589	015003	040	047040	053505	\$MNEW:	.ASCIZ	/ NEW = /	
2590	015010	036440	000040					

K05

MAINDEC-11-DZADL-A MACY11 27(722) 25-SEP-76 10:38 PAGE 63
 DZADLA.P11 READ AN OCTAL NUMBER FROM THE TTY

.SBTTL READ AN OCTAL NUMBER FROM THE TTY

```

;*****
;THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
;CHANGE IT TO BINARY.
;CALL:
;*      RDOCT          ;: READ AN OCTAL NUMBER
;*      RETURN HERE   ;: LOW ORDER BITS ARE ON TOP OF THE STACK
;*                    ;: HIGH ORDER BITS ARE IN $HIOCT
  
```

2591										
2592										
2593										
2594										
2595										
2596										
2597										
2598										
2599										
2600										
2601	015014	011646			\$RDOCT:	MOV	(SP), -(SP)			:: PROVIDE SPACE FOR THE
2602	015016	016666	000004	000002		MOV	4(SP), 2(SP)			:: INPUT NUMBER
2603	015024	010046				MOV	RO, -(SP)			:: PUSH RO ON STACK
2604	015026	010146				MOV	R1, -(SP)			:: PUSH R1 ON STACK
2605	015030	010246				MOV	R2, -(SP)			:: PUSH R2 ON STACK
2606	015032	104405			1\$:	RDLIN				:: READ AN ASCII LINE
2607	015034	012600				MOV	(SP)+, RO			:: GET ADDRESS OF 1ST CHARACTER
2608	015036	005001				CLR	R1			:: CLEAR DATA WORD
2609	015040	005002				CLR	R2			
2610	015042	112046			2\$:	MOV8	(RO)+, -(SP)			:: PICKUP THIS CHARACTER
2611	015044	001412				BEQ	3\$:: IF ZERO GET OUT
2612	015046	006301				ASL	R1			:: *2
2613	015050	006102				ROL	R2			
2614	015052	006301				ASL	R1			:: *4
2615	015054	006102				ROL	R2			
2616	015056	006301				ASL	R1			:: *8
2617	015060	006102				ROL	R2			
2618	015062	042716	177770			BIC	#1C7, (SP)			:: STRIP THE ASCII JUNK
2619	015066	062601				ADD	(SP)+, R1			:: ADD IN THIS DIGIT
2620	015070	000764				BR	2\$:: LOOP
2621	015072	005726			3\$:	TST	(SP)+			:: CLEAN TERMINATOR FROM STACK
2622	015074	010166	000012			MOV	R1, 12(SP)			:: SAVE THE RESULT
2623	015100	010237	015114			MOV	R2, \$HIOCT			
2624	015104	012602				MOV	(SP)+, R2			:: POP STACK INTO R2
2625	015106	012601				MOV	(SP)+, R1			:: POP STACK INTO R1
2626	015110	012600				MOV	(SP)+, RO			:: POP STACK INTO RO
2627	015112	000002				RTI				:: RETURN
2628	015114	000000			\$HIOCT:	.WORD	0			:: HIGH ORDER BITS GO HERE

.SBTTL SCOPE HANDLER ROUTINE

*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
*AND LOAD THE TEST NUMBER(\$TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
*AND LOAD THE ERROR FLAG (\$ERFLG) INTO DISPLAY<15:08>
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW14=1 LOOP ON TEST
*SW11=1 INHIBIT ITERATIONS
*SW09=1 LOOP ON ERROR
*SW08=1 LOOP ON TEST IN SWR<7:0>
*CALL
* SCOPE ;;SCOPE=IOT

2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643 015116
2644 015116 032777 040000 164014
2645 015124 001114
2646
2647 015126 000416
2648
2649 015130 013746 000004
2650 015134 012737 015154 000004
2651 015142 005737 177060
2652 015146 012637 000004
2653 015152 000463
2654 015154 022626
2655 015156 012637 000004
2656 015162 000423
2657 015164
2658 015164 032777 000400 163746
2659 015172 001404
2660 015174 127737 163740 001102
2661 015202 001465
2662 015204 105737 001103
2663 015210 001421
2664 015212 123737 001115 001103
2665 015220 101015
2666 015222 032777 001000 163710
2667 015230 001404
2668 015232 013737 001110 001106
2669 015240 000446
2670 015242 105037 001103
2671 015246 005037 001160
2672 015252 000415
2673 015254 032777 004000 163656
2674 015262 001011
2675 015264 005737 001202
2676 015270 001406
2677 015272 005237 001104
2678 015276 023737 001160 001104
2679 015304 002024
2680 015306 012737 000001 001104
2681 015314 013737 015372 001160
2682 015322 105237 001102
2683 015326 113737 001102 001200
2684 015334 011637 001106

\$\$SCOPE:
1\$: BIT #BIT14,\$SWR ;;LOOP ON PRESENT TEST?
BNE \$OVER ;;YES IF SW14=1
;****START OF CODE FOR THE XOR TESTER****
\$XTSTR: BR 6\$;;IF RUNNING ON THE "XOR" TESTER CHANGE
; THIS INSTRUCTION TO A "NOP" (NOP=240)
; SAVE THE CONTENTS OF THE ERROR VECTOR
MOV \$#ERRVEC,-(SP) ;;SET FOR TIMEOUT
MOV \$5,\$#ERRVEC ;;TIME OUT ON XOR?
TST \$#177060 ;;RESTORE THE ERROR VECTOR
MOV (SP)+,\$#ERRVEC ;;GO TO THE NEXT TEST
BR \$SVLAD ;;CLEAR THE STACK AFTER A TIME OUT
5\$: CMP (SP)+,(SP)+ ;;RESTORE THE ERROR VECTOR
MOV (SP)+,\$#ERRVEC ;;LOOP ON THE PRESENT TEST
BR 7\$
6\$;****END OF CODE FOR THE XOR TESTER****
BIT #BIT08,\$SWR ;;LOOP ON SPEC. TEST?
BEQ 2\$;;BR IF NO
CMPB \$SWR,\$TSTNM ;;ON THE RIGHT TEST? SWR<7:0>
BEQ \$OVER ;;BR IF YES
2\$: TSTB \$ERFLG ;;HAS AN ERROR OCCURRED?
BEQ 3\$;;BR IF NO
CMPB \$ERMAX,\$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
BHI 3\$;;BR IF NO
BIT #BIT09,\$SWR ;;LOOP ON ERROR?
BEQ 4\$;;BR IF NO
7\$: MOV \$LPERR,\$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
BR \$OVER
4\$: CLRB \$ERFLG ;;ZERO THE ERROR FLAG
CLR \$TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
BR 1\$;;ESCAPE TO THE NEXT TEST
3\$: BIT #BIT11,\$SWR ;;INHIBIT ITERATIONS?
BNE 1\$;;BR IF YES
TST \$PASS ;;IF FIRST PASS OF PROGRAM
BEQ 1\$;;INHIBIT ITERATIONS
INC \$ICNT ;;INCREMENT ITERATION COUNT
CMP \$TIMES,\$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
BGE \$OVER ;;BR IF MORE ITERATION REQUIRED
1\$: MOV #1,\$ICNT ;;REINITIALIZE THE ITERATION COUNTER
MOV \$MXCNT,\$TIMES ;;SET NUMBER OF ITERATIONS TO DO
\$SVLAD: INCB \$TSTNM ;;COUNT TEST NUMBERS
MOVB \$TSTNM,\$TESTN ;;SET TEST NUMBER IN APT MAILBOX
MOV (SP),\$LPADR ;;SAVE SCOPE LOOP ADDRESS

M05

MAINDEC-11-DZADL-A MACY11 27(732) 25-SEP-76 10:38 PAGE 65
 DZADLA.P11 SCOPE HANDLER ROUTINE

```

2685 015340 011637 001110          MOV      (SP), $LPERR      ;; SAVE ERROR LOOP ADDRESS
2686 015344 005037 001162          CLR      $ESCAPE         ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
2687 015350 112737 000001 001115    MOVB     #1, $ERMAX       ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
2688 015356 013777 001102 163556    $OVER:  MOV      $TSTNM, $DISPLAY ;; DISPLAY TEST NUMBER
2689 015364 013716 001106          MOV      $LPADR, (SP)    ;; FUDGE RETURN ADDRESS
2690 015370 000002          RTI                     ;; FIXES PS
2691 015372 003720          $MXCNT: 2000           ;; MAX. NUMBER OF ITERATIONS
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
015374 105237 001103          $ERROR:
7$:      INCB     $ERFLG      ;; SET THE ERROR FLAG
        BEQ      7$          ;; DON'T LET THE FLAG GO TO ZERO
        MOV      $TSTNM, $DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
        BIT      #BIT10, $SWR  ;; BELL ON ERROR?
        BEQ      1$          ;; NO - SKIP
        TYPE     $BELL        ;; RING BELL
1$:      INC      $ERTTL      ;; COUNT THE NUMBER OF ERRORS
        MOV      (SP), $ERRPC  ;; GET ADDRESS OF ERROR INSTRUCTION
        SUB      #2, $ERRPC
        MOVB     $ERRPC, $ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
        BIT      #BIT13, $SWR  ;; SKIP TIMEOUT IF SET
        BNE     20$          ;; SKIP TYPEOUTS
        JSR     PC, $ERRTYP    ;; GO TO USER ERROR ROUTINE
        TYPE     , $CRLF
20$:     CMPB     #APTENV, $ENV  ;; RUNNING IN APT MODE
        BNE     2$          ;; NO, SKIP APT ERROR REPORT
        MOVB     $ITEMB, 21$   ;; SET ITEM NUMBER AS ERROR NUMBER
        JSR     PC, $ATY4     ;; REPORT FATAL ERROR TO APT
21$:     .BYTE    0
        .BYTE    0
22$:     BR      22$          ;; APT ERROR LOOP
2$:      TST      $SWR        ;; HALT ON ERROR
        BPL     3$          ;; SKIP IF CONTINUE
        HALT
3$:      BIT      #BIT09, $SWR  ;; LOOP ON ERROR SWITCH SET?
        BEQ      4$          ;; BR IF NO
        MOV      $LPERR, (SP) ;; FUDGE RETURN FOR LOOPING
4$:      TST      $ESCAPE     ;; CHECK FOR AN ESCAPE ADDRESS
        BEQ      5$          ;; BR IF NONE
        MOV      $ESCAPE, (SP) ;; FUDGE RETURN ADDRESS FOR ESCAPE
5$:      CMP      #$ENDAD, $#42 ;; ACT-11 AUTO-ACCEPT?
        BNE     6$          ;; BRANCH IF NO
  
```

```

2741 015564 000000          HALT                ;;YES
2742 015566                6$:
2743 015566 000002          RTI                  ;;RETURN
2744                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
2745
2746                                ;:*****
2747                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
2748                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
2749                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
2750
2751 015570                $ERRTYP:
2752 015570 104400 001171    TYPE          $SCRLF                ;; "CARRIAGE RETURN" & "LINE FEED"
2753 015574 010046          MOV          RO,-(SP)                ;; SAVE RO
2754 015576 005000          CLR          RO                        ;; PICKUP THE ITEM INDEX
2755 015600 153700 001114    BISB         @#$ITEMB,RO
2756 015604 001004          BNE          1$                        ;; IF ITEM NUMBER IS ZERO, JUST
2757                                ;; TYPE THE PC OF THE ERROR
2758 015606 013746 001116    MOV          $ERRPC,-(SP)                ;; SAVE $ERRPC FOR TYPEOUT
2759                                ;; ERROR ADDRESS
2760 015612 104401          TYPOC
2761 015614 000426          BR          6$
2762 015616 005300                1$: DEC          RO
2763 015620 006300          ASL          RO
2764 015622 006300          ASL          RO
2765 015624 006300          ASL          RO
2766 015626 062700 001256    ADD          #$ERRTB,RO                ;; FORM TABLE POINTER
2767 015632 012037 015642    MOV          (RO)+,2$                ;; PICKUP "ERROR MESSAGE" POINTER
2768 015636 001404          BEQ          3$                        ;; SKIP TYPEOUT IF NO POINTER
2769 015640 104400          TYPE
2770 015642 000000                2$: .WORD      0
2771 015644 104400 001171    TYPE          $SCRLF                ;; "CARRIAGE RETURN" & "LINE FEED"
2772 015650 012037 015650    MOV          (RO)+,4$                ;; PICKUP "DATA HEADER" POINTER
2773 015654 001404          BEQ          5$                        ;; SKIP TYPEOUT IF 0
2774 015656 104400          TYPE
2775 015660 000000                4$: .WORD      0
2776 015662 104400 001171    TYPE          $SCRLF                ;; "CARRIAGE RETURN" & "LINE FEED"
2777 015666 011000          MOV          (RO),RO                ;; PICKUP "DATA TABLE" POINTER
2778 015670 001004          BNE          7$                        ;; GO TYPE THE DATA
2779 015672 012600          MOV          (SP)+,RO                ;; RESTORE RO
2780 015674 104400 001171    TYPE          $SCRLF                ;; "CARRIAGE RETURN" & "LINE FEED"
2781 015700 000207          RTS          PC                        ;; RETURN
2782 015702
2783 015702 013046          MOV          @ (RO)+,-(SP)                ;; SAVE @ (RO)+ FOR TYPEOUT
2784 015704 104401          TYPOC
2785 015706 005710          TST          (RO)
2786 015710 001770          BEQ          6$                        ;; IS THERE ANOTHER NUMBER?
2787 015712 104400 015720    TYPE          8$                        ;; BR IF NO
2788 015716 000771          BR          7$                        ;; TYPE TWO(2) SPACES
2789 015720 020040 000                8$: .ASCIZ    / /
2790 015724          .EVEN
    
```

.SBTTL TYPE ROUTINE

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

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

:OR
: TYPE
: MESADR

2808 015724 105737 001157
2809 015730 100002
2810 015732 000000
2811 015734 000430
2812 015736 010046
2813 015740 017600 000002
2814 015744 122737 000001 001214
2815 015752 001011
2816 015754 132737 000100 001215
2817 015762 001405
2818 015764 010037 015774
2819 015770 004737 016214
2820 015774 000000
2821 015776 132737 000040 001215
2822 016004 001003
2823 016006 112046
2824 016010 001005
2825 016012 005726
2826 016014 012600
2827 016016 062716 000002
2828 016022 000002
2829 016024 122716 000011
2830 016030 001430
2831 016032 122716 000200
2832 016036 001006
2833 016040 005726
2834 016042 104400
2835 016044 001171
2836 016046 105037 016202
2837 016052 000755
2838 016054 004737 016136
2839 016060 123726 001156
2840 016064 001350
2841 016066 013746 001154
2842
2843 016072 105366 000001
2844 016076 002770
2845 016100 004737 016136
2846 016104 105337 016202

STYPE: TSTB \$TFPLG ;; IS THERE A TERMINAL?
BPL 1\$ BR IF YES
HALT HALT HERE IF NO TERMINAL
BR 3\$ LEAVE
1\$: MOV RD, -(SP) SAVE RD
MOV 02(SP), RD GET ADDRESS OF ASCIZ STRING
CMPB \$APTENV, \$ENV RUNNING IN APT MODE
BNE 62\$ NO GO CHECK FOR APT CONSOLE
BITB \$APTPOOL, \$ENVM SPOOL MESSAGE TO APT
BEQ 62\$ NO GO CHECK FOR CONSOLE
MOV RD, 61\$ SETUP MESSAGE ADDRESS FOR APT
JSR PC, \$ATY3 SPOOL MESSAGE TO APT
61\$: .WORD 0 MESSAGE ADDRESS
62\$: BITB \$APTCSUP, \$ENVM APT CONSOLE SUPPRESSED
BNE 60\$ YES, SKIP TYPE OUT
2\$: MOVB (RD)+, -(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)+, RD RESTORE RD
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 TYPE A CR AND LF
CLRB \$CHARCNT ;; CLEAR CHARACTER COUNT
BR 2\$ GET NEXT CHARACTER
5\$: JSR PC, \$TYPEC GO TYPE THIS CHARACTER
6\$: 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.
7\$: DECB 1(SP) DOES A NULL NEED TO BE TYPED?
BLT 6\$ BR IF NO--GO POP THE NULL OFF OF STACK
JSR PC, \$TYPEC GO TYPE A NULL
DECB \$CHARCNT ;; DO NOT COUNT AS A COUNT

```

2847 016110 000770          BR      75          ;;LOOP
2848
2849          ;HORIZONTAL TAB PROCESSOR
2850
2851 016112 112716 000040      85:     MOVB     #' (SP)          ;;REPLACE TAB WITH SPACE
2852 016116 004737 016136      95:     JSR      PC,$TYPEC          ;;TYPE A SPACE
2853 016122 132737 000007 016202      BITB     #',$SCHARCNT          ;;BRANCH IF NOT AT
2854 016130 001372          BNE     95          ;;TAB STOP
2855 016132 005726          TST     (SP)+          ;;POP SPACE OFF STACK
2856 016134 000724          BR      25          ;;GET NEXT CHARACTER
2857 016136 105777 163006      $TYPEC: TSTB     25TPS          ;;WAIT UNTIL PRINTER IS READY
2858 016142 100375          BPL     $TYPEC
2859 016144 116677 000002 163000      MOVB     2(SP),25TPB          ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2860 016152 122766 000015 000002      CMPB     #CR,2(SP)          ;;IS CHARACTER A CARRIAGE RETURN?
2861 016160 001003          BNE     15          ;;BRANCH IF NO
2862 016162 105037 016202      CLRB     $SCHARCNT          ;;YES--CLEAR CHARACTER COUNT
2863 016164 000406          BR      $TYPEX          ;;EXIT
2864 016170 122766 000012 000002      15:     CMPB     #LF,2(SP)          ;;IS CHARACTER A LINE FEED?
2865 016176 001402          BEQ     $TYPEX          ;;BRANCH IF YES
2866 016200 105227          INCB     (PC)+          ;;COUNT THE CHARACTER
2867 016202 000000      $SCHARCNT: .WORD 0          ;;CHARACTER COUNT STORAGE
2868 016204 000207      $TYPEX: RTS      PC
2869
2870          .SBTTL  APT COMMUNICATIONS ROUTINE
2871
2872          ;*****
2873 016206 112737 000001 016452      $ATY1: MOVB     #1,$FFLG          ;;TO REPORT FATAL ERROR
2874 016214 112737 000001 016450      $ATY3: MOVB     #1,$MFLG          ;;TO TYPE A MESSAGE
2875 016222 000403          BR      $ATYC
2876 016224 112737 000001 016452      $ATY4: MOVB     #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
2877 016232          $ATYC:
2878 016232 010046          MOV     RO,-(SP)          ;;PUSH RO ON STACK
2879 016234 010146          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
2880 016236 105737 016450          TSTB     $MFLG          ;;SHOULD TYPE A MESSAGE?
2881 016242 001450          BEQ     55          ;;IF NOT: BR
2882 016244 122737 000001 001214      CMPB     #APTENV,$ENV          ;;OPERATING UNDER APT?
2883 016252 001031          BNE     35          ;;IF NOT: BR
2884 016254 132737 000100 001215      BITB     #APTPOOL,$ENVM          ;;SHOULD SPOOL MESSAGES?
2885 016262 001425          BEQ     35          ;;IF NOT: BR
2886 016264 017600 000004          MOV     24(SP),RO          ;;GET MESSAGE ADDR.
2887 016270 062766 000002 000004      ADD     #2,4(SP)          ;;BUMP RETURN ADDR.
2888 016276 005737 001174          15:     TST     $MSGTYPE          ;;SEE IF DONE W/ LAST XMISSION?
2889 016302 001375          BNE     15          ;;IF NOT: WAIT
2890 016304 010037 001210      MOV     RO,$MSGAD          ;;PUT ADDR IN MAILBOX
2891 016310 105720          25:     TSTB     (RO)+          ;;FIND END OF MESSAGE
2892 016312 001376          BNE     25
2893 016314 163700 001210      SUB     $MSGAD,RO          ;;SUB START OF MESSAGE
2894 016320 006200          ASR     RO          ;;GET MESSAGE LNGTH IN WORDS
2895 016322 010037 001212      MOV     RO,$MSGGLT          ;;PUT LENGTH IN MAILBOX
2896 016326 012737 000004 001174      MOV     #4,$MSGTYPE          ;;TELL APT TO TAKE MSG.
2897 016334 000413          BR      55
2898 016336 017637 000004 016362      35:     MOV     24(SP),45          ;;PUT MSG ADDR IN JSR LINKAGE
2899 016344 062766 000002 000004      ADD     #2,4(SP)          ;;BUMP RETURN ADDRESS
2900 016352 013746 177776          MOV     177776,-(SP)          ;;PUSH 177776 ON STACK
2901 016356 004737 015724          JSR     PC,$TYPE          ;;CALL TYPE P.CRO
2902 016362 000000          45:     .WORD 0

```

```

2903 016364          55:
2904 016364 105737 016452 108:  TSTB  $FFLG  ;; SHOULD REPORT FATAL ERROR?
2905 016370 001416          BEQ  12$  ;; IF NOT: BR
2906 016372 005737 001214  TST  $ENV  ;; RUNNING UNDER APT?
2907 016376 001413          BEQ  12$  ;; IF NOT: BR
2908 016400 005737 001174 118:  TST  $MSGTYPE  ;; FINISHED LAST MESSAGE?
2909 016404 001375          SNE  11$  ;; IF NOT: WAIT
2910 016406 017637 000004 001176  MOV  24(SP), $FATAL  ;; GET ERROR #
2911 016414 062766 000002 000004  ADD  #2, 4(SP)  ;; BUMP RETURN ADDR.
2912 016422 005237 001174  INC  $MSGTYPE  ;; TELL APT TO TAKE ERROR
2913 016426 105037 016452 128:  CLR  $FFLG  ;; CLEAR FATAL FLAG
2914 016432 105037 016451  CLR  $LFLG  ;; CLEAR LOG FLAG
2915 016436 105037 016450  CLR  $MFLG  ;; CLEAR MESSAGE FLAG
2916 016442 012601  MOV  (SP)+, R1  ;; POP STACK INTO R1
2917 016444 012600  MOV  (SP)+, R0  ;; POP STACK INTO R0
2918 016446 000207  RTS  PC  ;; RETURN
2919 016450          000  $MFLG: .BYTE 0  ;; MESSG. FLAG
2920 016451          000  $LFLG: .BYTE 0  ;; LOG FLAG
2921 016452          000  $FFLG: .BYTE 0  ;; FATAL FLAG
2922          016454  .EVEN
2923          000200  APTSIZE=200
2924          000001  APTENV=001
2925          000100  APTSPool=100
2926          000040  APTCSUP=040

```

2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982

```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPOS   ;;CALL FOR TYPEOUT
*   .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*   .BYTE  M              ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS
*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*$TYPOS OR $TYPOC
*CALL:
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPON   ;;CALL FOR TYPEOUT
*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
*CALL:
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPOC   ;;CALL FOR TYPEOUT

2953 016454 017646 000000 016677 $TYPOS: MOV     2(SP),-(SP)      ;; PICKUP THE MODE
2954 016460 116637 000001 016701 MOV     1(SP), $OFILL  ;; LOAD ZERO FILL SWITCH
2955 016466 112637 016701 MOV     (SP)+, $OMODE+1 ;; NUMBER OF DIGITS TO TYPE
2956 016472 062716 000002 ADD     #2, (SP)      ;; ADJUST RETURN ADDRESS
2957 016476 000406 BR      $TYPON
2958 016500 112737 000001 016677 $TYPOC: MOV     #1, $OFILL  ;; SET THE ZERO FILL SWITCH
2959 016506 112737 000006 016701 MOV     #6, $OMODE+1  ;; SET FOR SIX(6) DIGITS
2960 016514 112737 000005 016676 $TYPON: MOV     #5, $OCNT  ;; SET THE ITERATION COUNT
2961 016522 010346 MOV     R3, -(SP)    ;; SAVE R3
2962 016524 010446 MOV     R4, -(SP)    ;; SAVE R4
2963 016526 010546 MOV     R5, -(SP)    ;; SAVE R5
2964 016530 113704 016701 MOV     $OMODE+1, R4 ;; GET THE NUMBER OF DIGITS TO TYPE
2965 016534 005404 NEG     R4
2966 016536 062704 000006 ADD     #6, R4      ;; SUBTRACT IT FOR MAX. ALLOWED
2967 016542 110437 016700 MOV     R4, $OMODE  ;; SAVE IT FOR USE
2968 016546 113704 016677 MOV     $OFILL, R4  ;; GET THE ZERO FILL SWITCH
2969 016552 016605 000012 MOV     12(SP), R5  ;; PICKUP THE INPUT NUMBER
2970 016556 005003 CLR     R3          ;; CLEAR THE OUTPUT WORD
2971 016560 006105 15:    ROL     R5          ;; ROTATE MSB INTO "C"
2972 016562 000404 BR      35
2973 016564 006105 25:    ROL     R5          ;; GO DO MSB
2974 016566 006105 ROL     R5          ;; FORM THIS DIGIT
2975 016570 006105 ROL     R5
2976 016572 010503 MOV     R5, R3
2977 016574 006103 35:    ROL     R3          ;; GET LSB OF THIS DIGIT
2978 016576 105337 016700 DECB   $OMODE      ;; TYPE THIS DIGIT?
2979 016602 100016 BPL    75         ;; BR IF NO
2980 016604 042703 177770 BIC    #177770, R3 ;; GET RID OF JUNK
2981 016610 001002 BNE    45         ;; TEST FOR 0
2982 016612 005704 TST    R4         ;; SUPPRESS THIS 0?
                BEQ    55         ;; BR IF YES

```

2983	016616	005204		4\$:	INC	R4	:: DON'T SUPPRESS ANYMORE 0'S
2984	016620	052703	000060		BIS	#'0,R3	:: MAKE THIS DIGIT ASCII
2985	016624	052703	000040	5\$:	BIS	#' R3	:: MAKE ASCII IF NOT ALREADY
2986	016630	110337	016674		MOVB	R3,8\$:: SAVE FOR TYPING
2987	016634	104400	016674		TYPE	8\$:: GO TYPE THIS DIGIT
2988	016640	105337	016676	7\$:	DECB	\$OCNT	:: COUNT BY 1
2989	016644	003347			BGT	2\$:: BR IF MORE TO DO
2990	016646	002402			BLT	6\$:: BR IF DONE
2991	016650	005204			INC	R4	:: INSURE LAST DIGIT ISN'T A BLANK
2992	016652	000744			BR	2\$:: GO DO THE LAST DIGIT
2993	016654	012605		6\$:	MOV	(SP)+,R5	:: RESTORE R5
2994	016656	012604			MOV	(SP)+,R4	:: RESTORE R4
2995	016660	012603			MOV	(SP)+,R3	:: RESTORE R3
2996	016662	016666	000002 000004		MOV	2(SP),4(SP)	:: SET THE STACK FOR RETURNING
2997	016670	012616			MOV	(SP)+,(SP)	
2998	016672	000002			RTI		:: RETURN
2999	016674	000		8\$:	.BYTE	0	:: STORAGE FOR ASCII DIGIT
3000	016675	000			.BYTE	0	:: TERMINATOR FOR TYPE ROUTINE
3001	016676	000		\$OCNT:	.BYTE	0	:: OCTAL DIGIT COUNTER
3002	016677	000		\$OFILL:	.BYTE	0	:: ZERO FILL SWITCH
3003	016700	000000		\$OMODE:	.WORD	0	:: NUMBER OF DIGITS TO TYPE

.SBTTL POWER DOWN AND UP ROUTINES

3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086

016742 012737 017106 000024
016750 012737 000340 000026
016756 010046
016760 010146
016762 010246
016764 010346
016766 010446
016770 010546
016772 017746 162142
016776 010637 017112
017002 012737 017014 000024
017010 000000
017012 000776

017014 012737 017106 000024
017022 013706 017112
017026 005037 017112
017032 005237 017112
017036 001375
017040 012677 162074
017044 012605
017046 012604
017050 012603
017052 012602
017054 012601
017056 012600
017060 012737 016742 000024
017066 012737 000340 000026
017074 104400
017076 017114
017100 012716
017102 002256
017104 000002
017106 000000
017110 000776
017112 000000
017114 005015 047520 042527
017122 000122

017124 000310
017744 010000

000001

```
*****  
:POWER DOWN ROUTINE  
$PWRDN: MOV $SILLUP, @#PWRVEC ;; SET FOR FAST UP  
MOV #340, @#PWRVEC+2 ;; PRIO:7  
MOV RO, -(SP) ;; PUSH RO ON STACK  
MOV R1, -(SP) ;; PUSH R1 ON STACK  
MOV R2, -(SP) ;; PUSH R2 ON STACK  
MOV R3, -(SP) ;; PUSH R3 ON STACK  
MOV R4, -(SP) ;; PUSH R4 ON STACK  
MOV R5, -(SP) ;; PUSH R5 ON STACK  
MOV @SWR, -(SP) ;; PUSH @SWR ON STACK  
MOV SP, $SAVR6 ;; SAVE SP  
MOV $PWRUP, @#PWRVEC ;; SET UP VECTOR  
HALT  
BR .-2 ;; HANG UP  
  
*****  
:POWER UP ROUTINE  
$PWRUP: MOV $SILLUP, @#PWRVEC ;; SET FOR FAST DOWN  
MOV $SAVR6, SP ;; GET SP  
CLR $SAVR6 ;; WAIT LOOP FOR THE TTY  
1$: INC $SAVR6 ;; WAIT FOR THE INC  
BNE 1$ ;; OF WORD  
MOV (SP)+, @SWR ;; POP STACK INTO @SWR  
MOV (SP)+, R5 ;; POP STACK INTO R5  
MOV (SP)+, R4 ;; POP STACK INTO R4  
MOV (SP)+, R3 ;; POP STACK INTO R3  
MOV (SP)+, R2 ;; POP STACK INTO R2  
MOV (SP)+, R1 ;; POP STACK INTO R1  
MOV (SP)+, R0 ;; POP STACK INTO R0  
MOV $PWRDN, @#PWRVEC ;; SET UP THE POWER DOWN VECTOR  
MOV #340, @#PWRVEC+2 ;; PRIO:7  
TYPE ;; REPORT THE POWER FAILURE  
$PWRMG: .WORD $POWER ;; POWER FAIL MESSAGE POINTER  
MOV (PC)+, (SP) ;; RESTART AT BEG2  
$PWRAD: .WORD BEG2 ;; RESTART ADDRESS  
RTI  
$SILLUP: HALT ;; THE POWER UP SEQUENCE WAS STARTED  
BR .-2 ;; BEFORE THE POWER DOWN WAS COMPLETE  
$SAVR6: 0 ;; PUT THE SP HERE  
$POWER: .ASCIZ <15><12>"POWER"  
  
.EVEN  
.EVEN  
DIST: .BLKW 200. ;; STATE-WIDTH DISTRIBUTION  
BUFFER: .BLKW 4096. ;; BUFFER AREA  
  
.END
```


K06

CHAN	012271	1961	2273#											
CHANL	001366	730#	1317*	1517*	1598	1660	1934*	1935	1962	1992*	1993*	1994	2028*	2506
CHANNL	007004	1695#												
CH1	001354	725#	1502*	1503*	1506	1512*	1538	1544	1568	1574*	1576*			
CH2	001356	726#	1515*	1517	1530	1564	1575*	1577*						
CLEAR1	006732	1681#	1683											
CLEAR2	006776	1692#	1694											
CMSG	012134	761	2251#											
COMPAR	011104	1157	1190	1201	1213	1226	1238	1252	1274	1286	1297	1308	1334	2049#
CONV	006376	1608#	1612											
CONVR	007066	1706	1709#											
CONVRT	010772	1155	1188	1199	1211	1224	1236	1249	1260	1284	1295	1306	1547	2026#
CR =	000015	410#	2860	2870										
CRLF =	000200	411#	2831	2870										
CO	013561	841	2405#											
C1	013564	1629	2406#											
C2	013566	1822	1912	2407#										
C3	013571	1909	2408#											
DAC	001410	739#	1322	1325	1567	1571	1647*	1649*	1650	1672*	1980	1983	1986	1989
DASH	012163	1768	2259#											
DAWAIT	005026	1248	1259	1283	1377#									
DDISP =	177570	417#	600	820										
DECPNT	014453	1773#	1802*	2499#										
DECTYP	011374	1332	1528	1769	1775	1783	1787	1790	1801	1851	1955	1958	2120#	
DECS	003606	1124#	1125											
DELAY	001412	740#	1651*	1652*										
DELAY1	007130	1712	1720#											
DELAY2	007136	1714	1721	1722#										
DELAY3	007062	1707#	1708											
DELCLR	010242	1811	1823	1913	1922#									
DF1	014520	687	694	700	706	2512#								
DH1	014274	685	2479#											
DH2	014332	704	2484#											
DH3	014415	692	698	2493#										
DIFLIN	006712	1374	1677#											
DISPLA	001142	600#	820*	828*	1406*	2688*	2709*							
DISPRE	000174	533#	828											
DIST	017124	1684	1744*	1814	3083#									
DOME	012306	2084	2276#											
DSWR =	177570	416#	599	819										
DT1	014460	686	2504#											
DT2	014472	705	2506#											
DT3	014510	693	699	2509#										
DUMC	011224	1041	1044	1075	2080#									
DUMMY	001364	729#	1318*	1564*	1568*	1655	1935*	1994*						
DUMW	011152	1032	1055	2062#										
EDGE	001414	741#	1319*	1582*	1585*	1605*	1610*	1613*	1614*	1615*	1615*	1620	1665	2000*
EDGFLG	006502	2003#												
EMTVEC=	000030	1525*	1590*	1626	1636#	1938*	2008*							
EM1	014134	505#	803*	804*										
EM2	014162	684	2461#											
EM3	014162	691	2465#											
EM4	014212	697	2469#											
EM4	014243	703	2474#											
ER	010464	1968	1970	1973#										
ERMSG	012375	1339	1558	1779	1794	1806	1867	1973	2287#					

RBEG	001634	790	792*											
RDCHR =	104404	2565	3034*											
RDLIN =	104405	870	2606	3035*										
RDOCT =	104406	782	1511	1514	3036*									
READ	007173	1734*	1772											
RELACC	007636	1810	1827*											
REST1	002206	838	856*											
RESVEC=	000010	499*												
RET	006500	1627	1635*											
RETERR	003704	1142	1144*											
RETURN	001610	785*	1396	1607	1656	1701	2035							
RMS	001400	735*	1948	1950*	1954	1967	1976*	1980*	1983*	1996*	2004*	2005*		
RNA	001372	732*	861*	1704	2015*	2016*	2017*	2018	2021					
RNB	001374	733*	862*	2015	2018*	2019*	2020*	2022						
RNC	001376	734*	863*	2016	2019	2021*	2022*	2023*						
RST	011254	763	769	2087*										
RD	=%000000	420*	756	757*	758*	759	765	771	784*	843	846	849	851	858*
		859*	860*	873*	874*	875	878	881	884	887	890	915*	918*	922*
		923*	989*	992*	997*	1020*	1031*	1054*	1074*	1109*	1110*	1115*	1121*	1124*
		1132*	1171*	1175*	1179*	1327*	1328*	1329	1377*	1378*	1386*	1387*	1391	1397*
		1398*	1399	1403*	1406	1409	1482*	1486	1488*	1489	1490	1493	1494	1598*
		1599*	1600*	1601	1602*	1603*	1606*	1611*	1648*	1667*	1670	1679*	1681*	1684*
		1692*	1695*	1696*	1697*	1698*	1699	1700*	1730*	1732*	1771*	1800*	1804	1814*
		1816	1830*	1831	1834*	1841	1844	1872*	1874	1880*	1883*	1888	1891	1907*
		1910*	1916	1917	1922*	1924*	2026*	2027*	2028	2029*	2031	2032*	2033*	2036*
		2040*	2053*	2054*	2056*	2057	2067	2220*	2223	2603	2607*	2610	2626*	2753
		2754*	2755*	2762*	2763*	2764*	2765*	2766*	2767	2772	2777*	2779*	2783	2785
		2812	2813*	2818	2823	2826*	2878	2886*	2890	2891	2893*	2894*	2895	2917*
		3012	3013*	3014	3015*	3016*	3017*	3018*	3043	3068*				
R1	=%000001	421*	1414*	1415*	1483*	1484	1486*	1487*	1488	1493	1494	1654*	1668*	1680*
		1682*	1685*	1693*	1702*	1722*	1725*	1733*	1734	1760	1815*	1820*	1827*	1833*
		1834	1835	1873*	1881*	1886*	1903*	1915*	1919*	1923*	1926*	2052*	2054	2149*
		2159*	2160	2604	2608*	2612*	2614*	2616*	2619*	2622	2625*	2879	2916*	3044
		3067*												
R2	=%000002	422*	1257*	1258	1267	1268	1278*	1279	1331*	1400*	1417*	1519*	1523*	1524
		1567*	1571*	1579*	1586	1588*	1589	1704*	1705*	1707*	1711*	1713	1715*	1716*
		1718*	1720	1734*	1735*	1736*	1737*	1738*	1739	1743*	1744*	1745*	1746	1749
		1753	1774*	1782*	1785*	1786*	1789*	1797*	1798*	1799*	1800	1816*	1818*	1831*
		1832*	1833	1850*	1874*	1875*	1876*	1877*	1878*	1879*	1880	1887*	1898*	1916*
		1954*	1957*	2093	2095*	2096	2098*	2099	2100*	2101*	2104	2120	2123*	2124
		2126*	2130	2132*	2152*	2156*	2158*	2159	2605	2609*	2613*	2615*	2617*	2623
		2624*	3045	3066*										
R2POS	006306	1587	1589*											
R3	=%000003	423*	1271*	1273	1580*	1583*	1620*	1621	1629*	1630	1686*	1723*	1760*	1761*
		1762*	1763	1828*	1838	1840*	1846*	1847*	1848*	1849*	1850	1865	1888*	1889
		1890	1891*	1892	1894	1895	1897	2039*	2105*	2106*	2107*	2108*	2109*	2110*
		2111*	2114	2561	2562*	2563	2566*	2567	2571	2573	2575*	2577*	2960	2969*
		2975*	2976*	2979*	2984*	2985*	2986	2935*	3046	3065*				
R4	=%000004	424*	1251*	1272	1273*	1524*	1554	1539*	1659*	1660*	1661*	1662*	1663	1835*
		1837*	1838	1640	1884*	1900*	1902	2961	2963*	2964*	2965*	2966	2967*	2981
		2983*	2991*	2994*	3047	3064*								
R5	=%000005	425*	1155*	1157*	1188*	1190*	1199*	1201*	1211*	1213*	1224*	1226*	1236*	1238*
		1249*	1252*	1260*	1274*	1284*	1286*	1295*	1297*	1306*	1308*	1320*	1323*	1334*
		1348*	1350*	1352*	1360*	1364*	1505*	1547*	1565*	1569*	1574	1575	1592*	1641
		1675*	1841*	1842*	1843*	1853	1859*	1860	1865*	1901*	1902	1934	1940*	1978*
		1981*	1984*	1987*	1992	2010*	2026	2046*	2049	2050	2059	2060*	2962	2968*

TYPE = 104400	761	767	775	781	786	841	855	869	893	909	1330	1333	1339
	1341	1384	1390	1408	1419	1435	1440	1500	1501	1510	1513	1529	1535
	1537	1543	1556	1558	1629	1677	1759	1768	1770	1776	1779	1781	1784
	1788	1791	1794	1796	1803	1806	1808	1812	1813	1822	1829	1852	1858
	1867	1869	1905	1906	1909	1912	1931	1947	1956	1959	1961	1971	1973
	2066	2072	2084	2122	2146	2219	2569	2572	2576	2712	2720	2752	2769
	2771	2774	2776	2780	2787	2834	2987	3028*	3071				
TYPEDG = 006440	1536	1620*	1960										
TYPDC = 104401	2760	2784	3029*										
TYPON = 104403	3031*												
TYPOS = 104402	778	906	1393	1411	1437	1532	1540	1551	1623	1632	1765	1855	1862
	1964	2069	2075	3030*									
TYPDUT 011510	2131	2143*											
TYPRP 010336	1939	1947*	2009										
TYPSET 006024	1526	1528*	1591										
UNEXP 001432	749*	942	1071	1098									
VADR 001332	716*	901	1459	1460	1461								
VARLT1 011644	2156	2179*											
VARLT2 011654	2158	2184*											
VARLT3 011664	2152	2189*											
VECTOR 001324	713*	942*	1034*	1058*	1066*	1071*	1087*	1098*	1396*	1462*	1473*	1474*	1477
	1607*	1656*	1701*	2035*									
VECTRI 001330	715*	1463*	1464*	1477*	1478*	1479*							
VLIN 011640	1865	2176*											
VNP 011634	1969	2174*											
VNR 011632	1967	2149	2173*										
VSET 011636	1554	2175*											
VTFLG 002524	842	845	848	915*									
VTINIT 013711	1931	2431*											
VTSS 002176	850	853*											
VVCT 001334	717*	1462	1463										
V1 011610	1192	2163*											
V10 011614	2165*												
V115 011622	1254	2168*											
V144 011620	1299	2167*											
V2 011612	1203	1288	2164*										
V240 011624	1228	1240	1310	2169*									
V5 011626	1276	2170*											
V50 011616	1159	1215	2166*										
V500 011630	1336	2171*											
WFAOJ 011540	864	2149*											
WFTST 001422	744*	789*	791*	2154									
WIDE 001342	720*	1688*	1755*	1785	1798								
WIDMSG 012473	1788	2299*											
WRAP 003706	1146*	1441	1451										
SAPTHD 001000	559	565*											
SASTAT= ***** U	2904	2919											
SATYC 016232	2875	2877*											
SATY1 016206	2873*												
SATY3 016214	2819	2874*											
SATY4 016224	2725	2876*											
SAUTOB 001134	596*												
SBASE 001250	660*	895	1468	1469	1470								
SBDADR 001122	591*												
SBDGAT 001126	593*	895*	899	901*	981*	1038*	1062*	1091*	1126*	1127*	1128	1140*	1141
	2051*	2052	2504	2506	2509								

.SAPTY	1#	390#	2870
.SASTA	1#		
.SCATC	1#	390#	526
.SCMTA	1#	390#	572
.SDB2D	1#		
.SDB2O	1#		
.SDIV	1#		
.SEOP	1#	390#	2197
.SERRO	1#	390#	2692
.SERRT	1#	390#	2744
.SMULT	1#		
.SFARM	390#		
.SPOWE	1#	390#	3037
.SRAND	1#	390#	
.SRDDE	1#		
.SRDOC	1#	390#	2591
.SREAD	1#	390#	2517
.SR2AZ	1#		
.SSAVE	1#	390#	
.SSB2D	1#		
.SSB2O	1#		
.SSCOP	1#	390#	2629
.SSIZE	1#		
.SSPAC	390#		
.SSUPR	1#		
.SSWDO	390#		
.STRAP	1#	390#	3004
.STYPB	1#		
.STYPD	1#	390#	
.STYPE	1#	390#	2791
.STYPO	1#	390#	2927
.S4OCA	1#		
.1170	1#		

ADC	1616	1738	1849	1878	2005	2007	2017	2020	2023	2045					
ADD	901	1143	1325	1459	1460	1461	1462	1463	1467	1472	1478	1567	1576	1577	1610
	1628	1649	1696	1786	1798	1799	1833	1879	1980	1986	1993	2015	2016	2018	2019
ASL	2021	2022	2027	2039	2619	2766	2827	2887	2899	2911	2955	2965			
ASR	1328	1642	1643	1644	1645	1715	1743	2612	2614	2616	2763	2764	2765	3016	
	1613	1614	1615	1673	1735	1736	1737	1745	1762	1843	1846	1647	1848	1875	1876
	1877	2004	2006	2042	2043	2044	2105	2106	2107	2108	2109	2894			
BEQ	774	833	850	867	982	1129	1373	1405	1458	1485	1627	1706	1712	1714	1721
	1778	1793	1810	1871	1929	2131	2151	2221	2611	2659	2661	2663	2667	2676	2708
	2711	2733	2736	2768	2773	2786	2817	2830	2865	2881	2885	2905	2907	2982	
BGE	1666	1747	1893	2679											
BGT	1555	1671	1968	1970	2058	2215	2551	2989							
BHI	2665														
BIC	758	840	923	994	1127	1387	1474	1503	1705	2100	2110	2212	2538	2544	2552
	2618	2979													
BICB	874														
BIS	871	980	1011	1282	1398	1600	1657	1662	1698	2088	2101	2111	2194	2984	2985
BISB	2037	2143	2144	2145	2755										
BIT	1388	1404	1928	2644	2658	2666	2673	2710	2717	2732					
BITB	832	2816	2821	2853	2884										
BLE	1754	1805	1839	1866	1896	2125									
BLOS	2564														
BLT	1265	1740	2097	2549	2844	2990									
BMI	917	1724													
BNE	760	766	772	798	822	838	844	847	852	876	879	882	885	888	891
	919	936	978	993	1111	1125	1142	1176	1263	1269	1280	1379	1389	1416	1418
	1491	1584	1604	1612	1653	1669	1674	1683	1694	1708	1726	1731	1750	1757	1772
	1821	1845	1882	1899	1904	1920	1925	1927	2002	2034	2041	2063	2081	2135	2139
	2155	2540	2546	2568	2574	2645	2674	2718	2723	2740	2756	2778	2815	2822	2824
	2832	2840	2854	2861	2883	2889	2892	2909	2980	3061					
BPL	1007	1019	1037	1061	1070	1090	1106	1522	1587	1717	1836	1949	1952	2055	2094
	2103	2113	2121	2161	2536	2542	2730	2809	2858	2978					
BR	787	790	824	894	902	921	1042	1064	1076	1079	1093	1337	1338	1340	1407
	1420	1426	1443	1453	1466	1495	1507	1527	1719	1727	1729	1742	1752	1780	1795
	1807	1868	2142	2153	2157	2547	2570	2620	2647	2653	2656	2669	2672	2728	2761
	2788	2811	2837	2847	2856	2863	2875	2897	2956	2971	2992	3053	3077		
CLR	753	789	796	810	811	831	836	872	898	915	974	1004	1033	1072	1073
	1133	1180	1247	1256	1377	1383	1385	1432	1449	1464	1479	1487	1519	1579	1605
	1647	1648	1678	1681	1686	1687	1688	1689	1690	1691	1692	1728	1818	1827	1828
	1922	1950	1953	1976	1977	1996	1997	2030	2089	2095	2209	2210	2608	2609	2671
	2586	2754	2969	3059											
CLRB	1476	2127	2128	2129	2136	2140	2575	2670	2836	2862	2913	2914	2915		
CMP	773	797	821	843	846	849	851	903	935	1043	1065	1077	1094	1128	1141
	1264	1268	1279	1484	1490	1493	1494	1554	1626	1665	1670	1713	1720	1739	1746
	1753	1804	1838	1844	1865	1892	1895	1902	1967	1969	2057	2096	2124	2539	2545
	2548	2550	2563	2654	2678	2739									
CMPB	759	765	771	875	878	881	884	887	890	2134	2138	2567	2573	2660	2664
	2722	2814	2829	2831	2839	2860	2864	2882							
DEC	910	918	977	992	1110	1124	1175	1415	1417	1465	1583	1603	1611	1652	1668
	1682	1693	1707	1725	1730	1771	1820	1881	1898	1903	1919	1924	1926	2001	2033
	2040	2132	2213	2762											
DECB	1378	2843	2846	2977	2988										
EMT	404														
HALT	532	1930	2731	2741	2810	3052	3076								
INC	751	853	900	990	1005	1266	1278	1401	1471	1582	1608	1667	1709	1716	1722
	1723	1741	1744	1748	1751	1755	1758	1859	2000	2211	2677	2713	2912	2983	2991

INCB	3060														
IOT	2133	2137	2141	2682	2707	2866									
JMP	405														
	536	537	538	764	770	868	877	880	883	886	889	892	1428	1445	1455
JSR	2195	2228													
	763	769	842	845	848	857	864	932	944	952	960	557	995	1009	1021
	1032	1041	1044	1055	1075	1095	1112	1155	1157	1177	1188	1190	1199	1201	1211
	1213	1224	1226	1236	1238	1248	1249	1252	1259	1260	1274	1283	1284	1286	1295
	1297	1306	1308	1320	1323	1332	1334	1348	1350	1352	1360	1364	1374	1423	1424
	1425	1433	1434	1441	1442	1450	1451	1452	1505	1518	1520	1521	1525	1528	1536
	1547	1565	1569	1578	1581	1591	1703	1769	1775	1783	1787	1790	1801	1811	1817
	1819	1823	1851	1908	1911	1913	1918	1936	1937	1939	1955	1958	1960	1978	1981
	1994	1987	1995	1999	2009	2223	2719	2725	2819	2838	2845	2852	2901		
MOV	750	756	757	762	768	776	783	784	791	795	799	801	802	803	804
	805	806	807	808	809	813	814	817	818	819	820	825	827	828	829
	834	858	859	860	861	862	863	873	895	896	897	904	911	912	922
	929	930	931	942	943	951	959	966	973	975	976	981	989	991	997
	1003	1017	1020	1031	1034	1035	1038	1039	1054	1056	1058	1059	1062	1066	1067
	1068	1071	1074	1085	1086	1087	1088	1091	1097	1093	1104	1107	1108	1109	1115
	1120	1121	1122	1126	1132	1139	1140	1151	1152	1153	1154	1169	1170	1171	1172
	1173	1179	1187	1198	1210	1223	1235	1246	1251	1257	1258	1267	1270	1271	1273
	1294	1305	1316	1317	1318	1319	1322	1327	1331	1346	1347	1359	1363	1371	1382
	1386	1391	1396	1399	1400	1403	1406	1409	1414	1422	1427	1431	1436	1444	1448
	1454	1468	1469	1470	1473	1477	1480	1482	1483	1486	1488	1499	1502	1504	1509
	1512	1515	1516	1517	1524	1525	1530	1538	1544	1546	1549	1564	1568	1574	1575
	1580	1589	1590	1597	1601	1602	1606	1607	1620	1621	1630	1641	1646	1650	1651
	1654	1656	1659	1660	1663	1679	1680	1684	1685	1695	1699	1700	1701	1702	1704
	1711	1718	1732	1733	1734	1760	1763	1774	1782	1785	1789	1797	1800	1814	1815
	1816	1830	1831	1834	1835	1840	1841	1850	1853	1860	1872	1873	1874	1880	1883
	1884	1895	1886	1887	1838	1889	1890	1891	1894	1897	1900	1901	1907	1910	1915
	1916	1923	1934	1935	1938	1954	1957	1962	1992	1994	1998	2006	2026	2028	2031
	2032	2035	2036	2049	2050	2051	2052	2053	2064	2065	2067	2073	2082	2083	2098
	2099	2126	2149	2152	2156	2158	2159	2216	2220	2533	2534	2561	2562	2577	2578
	2579	2580	2601	2602	2603	2604	2605	2607	2622	2623	2624	2625	2626	2649	2650
	2652	2655	2668	2670	2691	2684	2685	2688	2689	2709	2714	2734	2737	2753	2758
	2767	2772	2777	2779	2783	2812	2813	2818	2826	2841	2878	2879	2886	2890	2895
	2896	2898	2900	2910	2916	2917	2952	2960	2961	2962	2968	2973	2993	2994	2995
	2996	2997	3012	3013	3017	3041	3042	3043	3044	3045	3046	3047	3048	3049	3050
	3051	3057	3058	3052	3063	3064	3065	3066	3067	3068	3069	3070	3073		
MOV8	812	1123	1475	1558	1655	1773	1802	2104	2114	2537	2543	2558	2571	2610	2683
	2687	2716	2724	2823	2651	2859	2873	2874	2876	2953	2954	2957	2958	2959	2963
	2966	2967	2986	3015											
NEG	1523	1588	1837	2056	2123	2964									
NOP	1150	2208	2224	2225	2226										
RESET	792	856	865	979	1008	1281	2087	2222							
ROL	934	2613	2615	2617	2970	2972	2973	2974	2976						
RTI	754	785	826	2553	2581	2627	2690	2743	2828	2998	3075				
RTS	913	924	1135	1144	1375	1380	1492	1557	1559	1572	1592	1617	1635	1675	1914
	1921	1932	1940	1972	1974	1990	2010	2024	2046	2060	2078	2085	2090	2115	2147
	2162	2781	2868	2918	3018										
SUB	1057	1272	1326	1329	1545	1571	1585	1672	1761	1832	1842	1983	1989	2003	2054
	2715	2893													
SLAB	1397	1599	1661	1697	2029										
TRAP	3020	3029	3030	3031	3034	3035	3036								
TST	837	866	899	920	1262	1372	1457	1489	1586	1749	1756	1777	1792	1809	1870
	1917	1948	1951	2059	2062	2080	2093	2120	2130	2150	2154	2160	2621	2651	2675

TSTB	2729	2735	2785	2825	2833	2855	2888	2906	2908	2981	3014	2541	2662	2808	2857
	916	1006	1018	1036	1060	1069	1089	1105	2102	2112	2535				
WAIT	2880	2891	2904												
.ASCII	1402	1609	1658	1664	1710	2038									
.ASCIZ	612	613	2322	2332	2345	2353	2359	2396	2435	2442	2445	2448	2451	2454	2273
	611	614	2231	2235	2238	2257	2259	2260	2263	2264	2265	2257	2270	2271	2273
	2276	2278	2279	2285	2287	2290	2293	2299	2305	2310	2317	2319	2326	2329	2336
	2340	2371	2382	2389	2393	2399	2405	2406	2407	2408	2409	2411	2412	2413	2457
	2461	2465	2469	2474	2479	2484	2497	2505	2586	2587	2589	2789	3079		
.BLKB	2584														
.BLKW	3083	3084													
.BYTE	581	582	587	588	596	597	605	606	607	608	630	631	641	642	649
	650	652	653	655	656	779	780	907	908	1394	1395	1412	1413	1438	1439
	1533	1524	1541	1542	1552	1553	1624	1625	1633	1634	1766	1767	1856	1857	1863
	1864	1965	1966	2070	2071	2076	2077	2230	2247	2248	2249	2251	2253	2417	2425
	2431	2498	2499	2500	2501	2582	2583	2726	7727	2919	2920	2921	2999	3000	3001
.DSABL	3002														
.ENABL	2522														
.END	1	390	2520												
.ENDC	3086														
	395	404	496	510	518	520	521	522	537	542	546	548	553	555	562
	575	579	581	609	610	611	612	616	619	641	649	652	655	658	659
	660	661	662	665	680	751	780	781	799	800	803	805	807	809	810
	811	813	815	836	877	880	883	886	889	892	903	908	909	918	920
	922	927	928	929	930	937	939	940	941	942	948	949	950	951	956
	957	958	959	963	964	965	966	970	971	972	973	974	983	986	987
	988	989	1000	1001	1002	1003	1004	1014	1015	1016	1017	1024	1025	1026	1027
	1043	1047	1048	1049	1050	1077	1082	1083	1084	1085	1086	1094	1101	1102	1103
	1104	1117	1118	1119	1120	1130	1143	1148	1149	1150	1151	1152	1166	1167	1168
	1169	1170	1184	1185	1186	1187	1188	1195	1196	1197	1198	1199	1207	1208	1209
	1210	1211	1220	1221	1222	1223	1224	1232	1233	1234	1235	1236	1243	1244	1245
	1246	1247	1291	1292	1293	1294	1295	1302	1303	1304	1305	1306	1313	1314	1315
	1316	1317	1341	1343	1344	1345	1346	1347	1356	1357	1358	1359	1360	1368	1369
	1370	1371	1372	1390	1395	1396	1413	1414	1534	1535	1542	1543	1553	1554	1625
	1626	1634	1635	1767	1768	1857	1858	1864	1865	1930	1966	1967	2071	2072	2077
	2078	2152	2154	2200	2201	2202	2204	2206	2209	2215	2218	2219	2220	2222	2228
	2230	2233	2520	2521	2522	2526	2554	2555	2562	2564	2567	2569	2585	2591	2594
	2596	2629	2632	2635	2640	2644	2646	2657	2660	2661	2662	2664	2666	2673	2677
	2682	2684	2688	2691	2692	2695	2698	2707	2714	2719	2720	2721	2729	2739	2743
	2744	2747	2762	2791	2794	2823	2873	2874	2877	2904	2919	2930	3007	3013	3016
	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3040	3049	3050	3056	3062
	3063	3073	3075	3082											
.EQUIV	404	405	413	428	429	458	459	460	461	462	463	464	465	466	467
	486	487	488	489	490	491	492	493	494	495					
.EVEN	619	2339	2502	2790	2922	3081	3082								
.IF	391	402	468	496	517	519	520	521	522	535	541	544	546	552	554
	561	574	578	580	609	610	611	615	616	618	641	649	652	655	658
	659	660	661	662	663	665	680	750	779	780	794	799	801	803	805
	807	809	810	811	813	831	876	879	882	885	888	891	902	907	908
	917	919	921	926	928	930	936	938	940	942	947	949	951	955	957
	959	962	964	966	969	971	973	974	982	985	987	989	999	1001	1003
	1004	1013	1015	1017	1023	1025	1027	1042	1046	1048	1050	1076	1081	1083	1085
	1086	1093	1100	1102	1104	1116	1118	1120	1129	1142	1147	1149	1151	1152	1165
	1167	1169	1170	1183	1185	1187	1188	1194	1196	1198	1199	1206	1208	1210	1211
	1219	1221	1223	1224	1231	1233	1235	1236	1242	1244	1246	1247	1290	1292	1294
	1295	1301	1303	1305	1306	1312	1314	1316	1317	1340	1342	1344	1346	1347	1355

	1357	1359	1360	1367	1369	1371	1372	1389	1394	1395	1412	1413	1533	1534	1541
	1542	1552	1553	1624	1625	1633	1634	1766	1767	1856	1857	1863	1864	1929	1965
	1966	2070	2071	2076	2077	2151	2153	2199	2200	2201	2202	2203	2204	2208	2214
	2217	2219	2220	2222	2228	2230	2231	2233	2519	2521	2522	2525	2526	2554	2562
	2563	2567	2568	2584	2585	2591	2593	2596	2608	2631	2634	2639	2644	2656	2658
	2659	2660	2662	2663	2664	2673	2675	2683	2685	2690	2691	2692	2694	2697	2707
	2710	2717	2719	2720	2722	2729	2732	2739	2743	2744	2746	2761	2777	2793	2814
	2872	2874	2877	2904	2919	2929	3006	3012	3016	3020	3029	3030	3031	3032	3033
	3034	3035	3036	3037	3039	3049	3050	3055	3062	3063	3071	3073	3075	3079	
.IFF	402	517	520	521	522	542	546	548	553	555	562	575	578	581	609
	616	619	750	779	781	799	876	879	882	885	888	891	902	908	917
	919	921	926	927	928	929	930	936	939	940	941	942	948	949	950
	951	956	957	958	959	963	964	965	966	970	971	972	973	983	986
	987	988	989	1000	1001	1002	1003	1014	1015	1016	1017	1024	1025	1026	1027
	1043	1047	1048	1049	1050	1077	1082	1083	1084	1085	1094	1101	1102	1103	1104
	1117	1118	1119	1120	1129	1142	1147	1148	1149	1150	1151	1166	1167	1168	1169
	1184	1185	1186	1187	1195	1196	1197	1198	1207	1208	1209	1210	1220	1221	1222
	1223	1232	1233	1234	1235	1243	1244	1245	1246	1247	1291	1292	1293	1294	1302
	1303	1304	1305	1313	1314	1315	1316	1317	1341	1343	1344	1345	1346	1347	1356
	1357	1358	1359	1360	1368	1369	1370	1371	1372	1389	1395	1413	1414	1534	1542
	1553	1554	1625	1626	1634	1635	1767	1768	1857	1858	1864	1865	1929	1966	2071
	2076	2078	2151	2153	2200	2203	2208	2215	2218	2230	2520	2522	2526	2528	2533
	2554	2555	2564	2568	2585	2594	2632	2657	2660	2661	2664	2691	2692	2695	2697
	2710	2739	2744	2747	2762	2791	2794	2873	2930	3007	3013	3040	3056	3073	
.IFT	2528	2533	2612	2628	2629	2672	2720								
.IFTF	2522	2526	2529	2608	2612	2628	2670	2719							
.IIF	390	395	400	514	515	516	518	521	522	532	615	619	777	800	803
	809	810	811	813	814	905	1392	1410	1731	1539	1550	1622	1631	1764	1854
	1861	1963	2068	2074	2201	2202	2209	2210	2230	2233	2520	2577	2585	2591	2635
	2636	2637	2638	2639	2640	2644	2671	2672	2688	2691	2692	2698	2699	2700	2701
.IRP	2702	2707	2732	2739	2744	2759	2784	2870	3028	3029	3030	3031	3034	3035	3036
	680	926	938	947	955	962	969	985	999	1013	1023	1046	1081	1100	1116
	1147	1165	1183	1194	1206	1219	1231	1242	1290	1301	1312	1342	1355	1367	2208
	2603	2624	2878	2879	2900	2916	2917	3043	3049	3062	3063				
.LIST	1	390	510	521	532	609	616	619	680	815	926	930	938	942	947
	951	955	959	962	966	969	973	985	989	999	1003	1013	1017	1023	1027
	1046	1050	1081	1085	1100	1104	1116	1120	1147	1151	1165	1169	1183	1187	1194
	1198	1206	1210	1219	1223	1231	1235	1242	1246	1290	1294	1301	1305	1312	1316
	1342	1346	1355	1359	1367	1371	2209	2222	2554	2639	2739	3020	3028	3029	3030
	3031	3032	3034	3035	3036	3037									
.MACRO	1	522	572	683	831	3020									
.MCALL	390	510	616	815											
.MEXIT	664														
.NLIST	1	390	510	521	532	609	616	619	680	815	926	930	938	942	947
	951	955	959	962	966	969	973	985	989	999	1003	1013	1017	1023	1027
	1046	1050	1081	1085	1100	1104	1116	1120	1147	1151	1165	1169	1183	1187	1194
	1198	1206	1210	1219	1223	1231	1235	1242	1246	1290	1294	1301	1305	1312	1316
	1342	1346	1355	1359	1367	1371	2209	2222	2554	2639	2739	3020	3028	3029	3030
	3031	3032	3034	3035	3036	3037									
.PAGE	572														
.REM	1														
.REPT	532														
.SBTTL	400	510	526	535	539	550	572	616	665	709	755	788	793	839	854
	926	938	947	955	962	969	985	999	1013	1023	1046	1081	1100	1116	1145
	1147	1165	1183	1194	1206	1219	1231	1242	1290	1301	1312	1342	1355	1367	1381
	1421	1430	1447	1456	1498	1508	2197	2234	2517	2591	2629	2692	2744	2791	2870

.TITLE	2927	3004	3020	3037											
.WORD	390														
	532	533	534	547	566	567	568	569	570	571	580	583	584	585	586
	589	590	591	592	593	594	595	596	599	600	621	622	623	624	625
	626	627	628	632	633	634	647	651	654	657	658	659	660	661	662
	2214	2217	2229	2628	2770	2775	2820	2867	2902	3003	3072	3074			

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

* DZADLA.SEG/SOL/CRF/PAGNUM/NL:TOC=DZADLA.SML,DZADLA.P11
 RUN-TIME: 39 51 7 SECONDS
 RUN-TIME RATIO: 159/98=1.6
 CORE USED: 34K (67 PAGES)

10			...B1	2243	012074	023440	...B5
63			...C1	2296	012456	020051	...C5
116			...D1	2348	013064	020040	...D5
172			...E1	2404	013560	000	...E5
213			...F1				...F5
241			...G1	2463	014150	020056	...G5
297			...H1	2507	014500	001124	...H5
352			...I1	2526			...I5
408	000011		...J1	2582	014746	000	...J5
464			...K1	2600			...K5
520			...L1	2638			...L5
548	000214		...M1	2694			...M5
581	001102	000	...N1	2750			...N5
637			...B2	2800			...B6
674			...C2	2856	016134	000724	...C6
718	001336	000000	...D2	2912	016422	005237	...D6
764	001510	000137	...E2	2936			...E6
797	001644	022706	...F2	2992	016652	000744	...F6
845	002144	004737	...G2	3013	016704	016600	...G6
863	002244	012737	...H2	3046	016764	010346	...H6
912	002516	012637	...I2	ADDW11=	000000		...I6
934	002606	006137	...J2	BEGINA	005260		...J6
978	002754	001375	...K2	CONV	006376		...K6
1032	003146	004737	...L2	GNS =	***** U		...L6
1088	003430	012777	...M2	NXTCMP	010102		...M6
1125	003610	001376	...N2	RETERR	003704		...N6
1154	003732	012737	...B3	SETB	006220		...B7
1203	004114	011612	...C3	TADDR	001370		...C7
1251	004256	013704	...D3	TYPON =	104403		...D7
1299	004500	011620	...E3	\$CRLF	001171		...E7
1321	004572	000062	...F3	\$MSGTY	001174		...F7
1351	004734	000007	...G3	\$TPB	001152		...G7
1390	005100	104400	...H3	NEWTST	1#	510#	...H7
1430			...I3	.SERRT	1#	390#	...I7
1465	005460	005337	...J3		2711	2733	...J7
1517	005756	013737	...K3		1425	1433	...K7
1567	006172	063702	...L3	.BLK8	2584	2465	...L7
1602	006350	012700	...M3	.IFF	402	517	...M7
1646	006530	012737	...N3	RUN-TIME:	39 51 7 SECON		...N7
1685	006744	012701	...B4	**END**	USER DAVIES, TOM		...B8
1741	007216	005237	...C4				
1791	007454	104400	...D4				
1834	007662	010120	...E4				
1883	010044	012700	...F4				
1931	010272	104400	...G4				
1984	010532	004537	...H4				
2034	011024	001376	...I4				
2070	011204	002	...J4				
2095	011276	005002	...K4				
2127	011420	105037	...L4				
2172	011630	000062	...M4				
2206			...N4				