

ADF11

LOGIC DIAGNOSTIC
MD-11-DZADG-A

EP-DZADG-A-DLA

NOV 1976

COPYRIGHT © 1976

digital

FICHE 1 OF 1

MADE IN U.S.A.

Frame 1	Frame 2	Frame 3	Frame 4	Frame 5	Frame 6
Frame 7	Frame 8	Frame 9	Frame 10	Frame 11	Frame 12
Frame 13	Frame 14	Frame 15	Frame 16	Frame 17	Frame 18
Frame 19	Frame 20	Frame 21	Frame 22	Frame 23	Frame 24
Frame 25	Frame 26	Frame 27	Frame 28	Frame 29	Frame 30
Frame 31	Frame 32	Frame 33	Frame 34	Frame 35	Frame 36
Frame 37	Frame 38	Frame 39	Frame 40	Frame 41	Frame 42
Frame 43	Frame 44	Frame 45	Frame 46	Frame 47	Frame 48

.REM X

DZADGA.CMB

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZADG-A-D
PRODUCT NAME:	ADF11 LOGIC DIAGNOSTIC TEST
DATE CREATED:	MARCH 4, 1974
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	EARL L. BOUSE

EO1

136
137
138

T'NNN

10.

-2-

133
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

B. LOGIC TEST

A. THE LOGIC TEST CONSISTS OF '106' SUBTESTS WHICH CHECK AND EXERCISE THE 'ADF11' EACH TEST IS LOOPED '50' TIMES TO TEST LOGIC RELIABILITY.

B. RESTRICTIONS

EXTERNAL SYNC MUST BE JUMPERED TO GROUND TO ENSURE THAT NO EXTRANEOUS CONVERSIONS TAKE PLACE. A PROVISION IS PROVIDED IN THE 'ANALOG' TEST TO TEST THIS FEATURE.

C. TEST TIME

IT TAKES APPROXIMATELY '2' MINUTES TO MAKE ONE COMPLETE PASS OF THE LOGIC TEST. THE MESSAGE 'LOGIC OK' IS TYPED ON PASS COMPLETION.

D. LOGIC ERRORS

ON ENCOUNTERING A LOGIC ERROR (ALL DATA SWITCHES DOWN) THE ERROR ADDRESS AND THE CONTENTS OF THE A/D REGISTERS ARE TYPED OUT.

E. CONTROL SWITCHES (TELETYPE)

1. ↑C (CONTROL C)

TYPING A '↑C' AT ANY TIME WHILE RUNNING THE LOGIC TEST WILL ENABLE THE PROGRAM TO RETURN TO THE MONITOR.

G. CONSOLE SWITCH SETTINGS FUNCTIONS

CONSOLE SW11=0	NORMAL RUN (2048 PASSES/TEST)
CONSOLE SW11=1	SURPRESS SUBPROGRAM ITERATIONS
CONSOLE SW12=0	NORMAL ERROR LOOP
CONSOLE SW12=1	ISSUE R-E-S-E-T (*) IN ERROR LOOP
CONSOLE SW13=0	PRINT ERROR MESSAGE
CONSOLE SW13=1	INHIBIT ERROR MESSAGES
CONSOLE SW14=0	INHIBIT LOOPING
CONSOLE SW14=1	LOOP ON CURRENT TEST
CONSOLE SW15=0	CONTINUE AFTER TYPING ERROR
CONSOLE SW15=1	HALT ON ERROR

* SOME SUBTESTS WILL REQUIRE A "R-E-S-E-T" INSTRUCTION TO ENABLE THE A/D TO BE INITIALIZED FOR SCOPING.

195
196

H. SUBROUTINE CALLS

1. SCOPE

THIS IS AN "ENT TRAP" TO THE SUBROUTINE CALLED "SCOPEC". THIS SUBROUTINE HAS TWO (2) FUNCTIONS. ONE, IT RECORDS THE "P.C." OF THE "SCOPE" CALL. THIS ENABLES THE PROGRAM TO ENTER THE NEXT LOGIC TEST OR TO LOOP BACK TO THE PREVIOUS "SCOPE" ADDRESS. THUS A SCOPE LOOP WILL CONTAIN ALL THE CODE BETWEEN TWO SCOPE CALLS. TWO, CHECKS THE ITERATIONS TO BE MADE ON THE CURRENT TEST BEFORE CONTINUING TO THE NEXT TEST.

2. ERROR

THIS IS A "T-R-A-P" TO A SUBROUTINE CALLED "LOGERR" (LOGIC ERROR). THIS SUBROUTINE ALSO HAS TWO FUNCTIONS. ONE, TO PRINT THE ADDRESS OF T-H-A-T ERROR DETECTED. TWO, TO HALT ON THE ERROR IF SWITCH "15" IS SET TO A "1".

9. DATA UPDATE TEST

A. THE "DATA UPDATE" IS AN OPERATOR INTERVENTION TEST USED TO THE A/D "DATA BUFFERS" AND "UPDATE" LOGIC. THE TEST REQUESTS FOUR (4) SPECIFIC VOLTAGES (2 POSITIVE AND 2 NEGATIVE) TO BE SUPPLIED TO CHANNEL "0". A CONVERSION IS TAKEN AT EACH OF THESE VOLTAGES AND THEN THE FOUR BUFFERED DATA WORDS ARE READ. THE PROGRAM CHECKS THE SIGN OF THE READ DATA TO DETERMINE IF THE DATA WAS BUFFERED CORRECTLY.

WHEN RUNNING THIS TEST WITH A UNI-POLAR A/D. THE USER SHOULD SUPPLY +1.00 VOLT FOR THE 2ND AND 3RD VOLTAGES. AN ERROR PRINTOUT WILL OCCUR ON THESE READINGS. THESE READINGS MAY THEN BE VERIFIED FOR BEING CLOSE TO THE 1 VOLT VALUE. IF THE READINGS ARE A 5 VOLT VALUE, THE DATA ISN'T GETTING BUFFERED CORRECTLY.

B. STARTING SEQUENCE

1. TYPE "D<CR>" TO RUN THE "DATA UPDATE TEST".
2. TYPE PROGRAM WILL TYPE THE TEST HEADER AND THEN ASK FOR FOUR (4) SPECIFIC VOLTAGES TO BE SUPPLIED.
3. TYPE A CARRIAGE RETURN <CR> AFTER SUPPLING THE REQUESTED VOLTAGE.

C. ERROR FORMAT

IF THE READ DATA IS NOT THE POLARITY EXPECTED, THE FOLLOWING TYPEOUT WILL OCCUR:

1ST RD NNNNN

197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250

I01

021100
021100
021100

A

B

-4-

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
 343
 344
 345

001000
 000400
 000200
 000100
 000040
 000020
 000010
 000004
 000002
 000001

 000000
 000001
 000002
 000003
 000004
 000005
 000006
 000007

 000003
 000004
 000005
 005746
 005726
 024646
 022626
 000240
 000002

SW09=1000
 SW08=400
 SW07=200
 SW06=100
 SW05=40
 SW04=20
 SW03=10
 SW02=4
 SW01=2
 SW00=1

;REGISTER DEFINITIONS

R0=%0
 R1=%1
 R2=%2
 R3=%3
 R4=%4
 R5=%5
 SP=%6
 PC=%7

;INSTRUCTIONS DEFINITIONS

ADDR3=%3
 ADCR4=%4
 ADCSR5=%5
 PUSH1SP=5746
 POP1SP=5726
 PUSH2SP=24646
 POP2SP=22626
 NOP=240
 X=2

;LOAD TRAP CATCHER INTO LOC'S 0-1000

346				
347				
348		000000		
349	000000	000002	.=0	
350	000002	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
351	000004	000006	4.+2	
352	000006	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
353	000010	000012	4.+2	
354	000012	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
355	000014	000016	4.+2	
356	000016	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
357	000020	000022	4.+2	
358	000022	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
359	000024	000026	4.+2	
360	000026	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
361	000030	000032	4.+2	
362	000032	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
363	000034	000036	4.+2	
364	000036	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
365	000040	000042	4.+2	
366	000042	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
367	000044	000046	4.+2	
368	000046	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
369	000050	000052	4.+2	
370	000052	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
371	000054	000056	4.+2	
372	000056	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
373	000060	000062	4.+2	
374	000062	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
375	000064	000066	4.+2	
376	000066	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
377	000070	000072	4.+2	
378	000072	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
379	000074	000076	4.+2	
380	000076	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
381	000100	000102	4.+2	
382	000102	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
383	000104	000106	4.+2	
384	000106	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
385	000110	000112	4.+2	
386	000112	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
387	000114	000116	4.+2	
388	000116	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
389	000120	000122	4.+2	
390	000122	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
391	000124	000126	4.+2	
392	000126	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
393	000130	000132	4.+2	
394	000132	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
395	000134	000136	4.+2	
396	000136	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
397	000140	000142	4.+2	
398	000142	000004	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
399	000144	000146	4.+2	
400	000146	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
401	000150	000152	.+2	

402	000152	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
403	000154	000156	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
404	000156	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
405	000160	000162	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
406	000162	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
407	000164	000166	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
408	000166	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
409	000170	000172	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
410	000172	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
411	000174	000176	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
412	000176	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
413	000200	000202	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
414	000202	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
415	000204	000206	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
416	000206	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
417	000210	000212	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
418	000212	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
419	000214	000216	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
420	000216	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
421	000220	000222	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
422	000222	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
423	000224	000226	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
424	000226	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
425	000230	000232	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
426	000232	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
427	000234	000236	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
428	000236	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
429	000240	000242	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
430	000242	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
431	000244	000246	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
432	000246	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
433	000250	000252	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
434	000252	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
435	000254	000256	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
436	000256	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
437	000260	000262	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
438	000262	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
439	000264	000266	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
440	000266	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
441	000270	000272	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
442	000272	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
443	000274	000276	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
444	000276	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
445	000300	000302	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
446	000302	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
447	000304	000306	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
448	000306	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
449	000310	000312	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
450	000312	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
451	000314	000316	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
452	000316	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
453	000320	000322	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
454	000322	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
455	000324	000326	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
456	000326	000004	4.	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
457	000330	000332	4.+2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.

458	000332	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
459	000334	000336	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
460	000336	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
461	000340	000342	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
462	000342	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
463	000344	000346	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
464	000346	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
465	000350	000352	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
466	000352	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
467	000354	000356	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
468	000356	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
469	000360	000362	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
470	000362	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
471	000364	000366	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
472	000366	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
473	000370	000372	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
474	000372	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
475	000374	000376	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
476	000376	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
477	000400	000402	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
478	000402	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
479	000404	000406	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
480	000406	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
481	000410	000412	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
482	000412	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
483	000414	000416	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
484	000416	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
485	000420	000422	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
486	000422	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
487	000424	000426	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
488	000426	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
489	000430	000432	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
490	000432	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
491	000434	000436	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
492	000436	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
493	000440	000442	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
494	000442	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
495	000444	000446	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
496	000446	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
497	000450	000452	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
498	000452	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
499	000454	000456	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
500	000456	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
501	000460	000462	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
502	000462	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
503	000464	000466	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
504	000466	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
505	000470	000472	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
506	000472	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
507	000474	000476	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
508	000476	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
509	000500	000502	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
510	000502	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
511	000504	000506	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
512	000506	000004	4	; TRAPPED OR INTERRUPTED TO PREV. ADDR.
513	000510	000512	4 +2	; TRAPPED OR INTERRUPTED TO PREV. ADDR.

104001
104002
104003
104004
104005
104006
104007
104010
104011
104012
104013

DECOCT=EMT+1
SCOPE=EMT+2
SPACE=EMT+3
PRTOCT=EMT+4
TTYIN=EMT+5
TSTTKS=EMT+6
CKDONE=EMT+7
NULL=EMT+10
INITAD=EMT+11
SAVREG=EMT+12
GETREG=EMT+13

;DECIMAL TO OCTAL CONVERSIN ROUTINE
;LOGIC TEST SCOPE SUBROUTINE
;TYPE 'N' SPACES
;OCTAL PRINT ROUTINE
;TELETYPE INPUT ROUTINE
;SUBROUTINE TO TEST FOR KEYBOARD FLAG
;ROUTINE TO CHECK FOR 'A/D READY'
;ROUTINE TO PRINT NULL CHAR.'S
;ROUTINE TO INITIALIZE THE A/D
;ROUTINE TO SAVE 'R1-R5' ON THE STACK
;ROUTINE TO GET 'R1-R5' FROM THE STACK

```

;EMT DISPATCH SERVICE ROUTINE
;ARGUMENT OF EMT IS EXTRACTED AND USED AS OFFSET TO OBTAIN POINTER
;TO THE SELECTED SUBROUTINE.
    
```

```

001200 011646
001202 162716 000002
001206 017616 000000
001212 005716
001214 001001
001216 000000
001220 006316
001222 042716 177001
001226 062716 001240
001232 017616 000000
001236 000136
    
```

```

ENTSRV: MOV (SP), -(SP) ;GET PC FOR TO RETURN
        SUB #2, (SP) ;PC OF EMT
        MOV 2(SP), (SP) ;GET EMT
        TST (SP) ;IS EMT VALID?
        BNE EMTOK
        HALT ;INVALID EMT
EMTOK: ASL (SP) ;MULTIPLY EMT ARG BY '2'
       BIC #177001, (SP) ;CLEAR UNWANTED BITS
       ADD #EMTTAB, (SP) ;POINTER TO SUBROUTINE ADDRESS
       MOV 2(SP), (SP) ;SUBROUTINE ADDRESS
       JMP 2(SP)+ ;GO TO SUBROUTINE
    
```

```

;EMT DISPATCH TABLE
    
```

```

001240 012116
001242 011304
001244 012622
001246 010724
001250 012224
001252 011020
001254 012712
001256 011574
001260 012724
001262 011756
001264 011766
001266 012042
    
```

```

EMTTAB: TYPMES ;MESSAGE PRINT ROUTINE
        BCD2BIN ;DECIMAL TO BINARY CONVERSION ROUTINE
        SCOPEC ;LOGIC TEST SCOPE ROUTINE
        XSPACE ;SUBROUTINE TO TYPE SPACES
        OCTPRT ;OCTAL PRINT ROUTINE
        TELEPRN ;TELEPRINTER SERVICE ROUTINE
        TSTKEY ;SUBROUTINE TO TEST FOR KEYBOARD FLAG
        A/DONE ;ROUTINE TO TEST FOR 'A/D DONE'
        PRINTN ;ROUTINE TO PRINT NULL CHAR.'S
        XINIT ;ROUTINE TO INITIALIZE THE A/D
        XSAVRG ;ROUTINE TO SAVE 'R1-R5' ON STACK
        XGETRC ;ROUTINE TO GET 'R1-R5' FROM STACK
    
```

```

;REGISTER ADDRESSES
    
```

```

001270 177776
001272 177560
001274 177562
001276 177564
001300 177566
001302 177570
001304 177571
001306 164006
001310 164010
001312 164004
001314 164000
001316 164012
001320 164002
001322 164014
001324 164016
001326 000274
001330 000276
    
```

```

PSW: 177776
TKS: 177560
TKB: 177562
TPS: 177564
TPB: 177566
SMR: 177570
SMRO: 177571
ADCR: 164006
ADCSR: 164010
ADWCR: 164004
ADSMR: 164000
ADOSR: 164012
ADWRA: 164002
ADWRB: 164014
ADAOR: 164016
ADINT: 0274
ADLVL: 0276
    
```

```

ADDRESS OF PROCESSOR STATUS REG.
ADDRESS OF KEYBOARD STATUS REG.
" " BUFFER
" " PRINTER STATUS REG.
" " PRINTER BUFFER REG.
" " SWITCH REG.
" " HIGH BYTE
" " A/D CONTROL REG.
" " A/D CONTROL & STATUS REG
" " A/D WORD COUNT REG.
" " A/D STATUS WORD REG.
" " A/D DATA BUFFER REG.
" " A/D WORD REG 'A'
" " A/D " " 'B'
" " A/D OFFSET REG.
" " A/D INTERRUPT VECTOR
ADDRESS OF A/D INTERRUPT LEVEL
    
```

:TEST INITIALIZATIN ROUTINE. PROGRAM IS SELF STARTING TO THIS ROUTINE.
:THE ROUTINE IS EXECUTED ON LOADING ONLY
:*****

001332	016706	012444		INIT:	MOV	STACK, SP		;INIT STACK POINTER=1000
001336	012777	000340	177724		MOV	#340, APSW		
001344	104000				PRINT			;CALL MESSAGE PRINTER VIA 'EMT'
001346	012755				TITLE			;TYPE PROGRAM HEADER.
001350	005067	012416		INIT1:	CLR	ADSIGN		;UNIPOLAR=0, BIPOLAR=1
001354	104000				PRINT			
001356	013063				MES2			;REQUEST THE A/D LENGTH
001360	104005				TTYIN			;WAIT FOR ENTRY
001362	104001				DECOCT			;CONVERT A/D LENGTH TO OCTAL
001364	012701	003776			MOV	#3776, R1		;INIT AS 'INC MEM' OFFSET
001370	012702	001000			MOV	#1000, R2		;= TO +5V VALUE FOR 10 BITS
001374	162767	000012	010056		SUB	#12, BCDTAB		;A/D LENGTH = TO 10 BITS?
001402	001414				BEG	CORSIZ		;YES, EXIT
001404	012703	000005			MOV	#5, R3		;NO, TEST UP TO 15 BITS
001410	006301			SIZE:	ASL	R1		;BUMP MEM. OFFSET
001412	006302				ASL	R2		;ALSO A/D SIZE
001414	005367	010040			DEC	BCDTAB		;DECREMENT COUNT
001420	001405				BEG	CORSIZ		;EXIT IF DONE
001422	005303				DEC	R3		
001424	100371				BPL	SIZE		;BRANCH UNTIL 15 IS REACHED
001426	104000				PRINT			;ILLEGAL ENTRY
001430	013316				QMARK			;PRINT '??'
001432	000746				BR	INIT1		;RETRY
001434	010167	012334		CORSIZ:	MOV	R1, SIGNBF		;SAVE A/D WORD LENGTH
001440	006267	012330			ASR	SIGNBF		;SET SIGN BITS
001444	005167	012324			COM	SIGNBF		;TEST FOR SIGN BIT
001450	005767	012316			TST	ADSIGN		;BRANCH IF NOT SET
001454	001401				BEG	.+4		;OTHERWISE ADD 1 BIT TO CONVERTER LENGTH.
001456	006301				ASL	R1		;SET ALL POSSIBLE SHIFTED BITS
001460	052701	000776			BIS	#776, R1		;INITIAL THE TIME OUT ADDRESS
001464	012737	001532	000004		MOV	#INITA, #4		
001472	012737	000340	000006		MOV	#340, #6		
001500	005067	012314			CLR	INCLG		;CLR INCREMENT MEMORY FLAG
001504	062701	020000			ADD	#20000, R1		;ADD 4K OFFSET TO A/D LENGTH
001510	010167	012306			MOV	R1, MEMSIZ		;SAVE MEMORY SIZE
001514	006302				ASL	R2		;SET UP OFFSET FOR AVERAGING ROUTINE
001516	010267	012306			MOV	R2, ADSIZE		;SAVE IT
001522	005737	037776		CORSZA:	TST	#37776		;TEST IF 8K MEMORY IS AVAILABLE
001526	005267	012266			INC	INCLG		;SET SOFTWARE SWITCH
001532	012737	001546	000004	INITA:	MOV	#INITB, #4		;SET RET FOR FIRST NON EX MEM
001540	005001				CLR	R1		;TEST FOR MAX CORE IN SYSTEM
001542	005721				TST	(R1)+		;TRAP & RET TO INITA:
001544	000776				BR	.-2		
001546	005741			INITB:	TST	-(R1)		;LAST CORE AVAILABLE
001550	162701	004000			SUB	#4000, R1		
001554	010167	012244			MOV	R1, CORMAX		;SAVE MAX CORE AVAILABLE
001560	012737	000006	000004		MOV	#6, #4		;RESTORE TIME OUT ADDRESS
001566	012737	000004	000006		MOV	#4, #6		

001574 104000
001576 013172
001600 000411

PRINT
MES4
BR INIT2

;PRINT THE TEST CALL LETTERS
;GO AND AWAIT COMMAND

;MONITOR SUBROUTINE. ENTER VIA 'IC' OR A RESTART AT LOCATION '200'.

001602 104010
001604 000005
001606 104010
001610 016706 012166
001614 004767 010034
001620 104000
001622 013302

MONITR: NULL
RESET
NULL
MOV STACK,SP
JSR PC,CLRINT
PRINT
CNTRLC

;INITIALIZE ON ENTRY
;RESET STACK POINTER
;CLR A/D INTR ADDR TO HALT
;CALL MESSAGE PRINTER
;TYPE 'IC'

001624 012767 001546 012152
001632 104000
001634 013313
001636 104005
001640 122767 000104 007416
001646 001002
001650 000167 005770
001654 122767 000114 007402
001662 001002
001664 000167 000106
001670 122767 000123 007366
001676 001002
001700 000167 006100
001704 022767 000124 007352
001712 001002
001714 000167 006670
001720 104000
001722 013316
001724 000737

INIT2: MOV #INITB,AVECTR
PRINT
DOT
TTYIN
CMPB #104,INBUF
BNE +6
JMP DATA
CMPB #114,INBUF
BNE +6
JMP LOGIC
CMPB #123,INBUF
BNE +6
JMP SYNC0
CMP #124,INBUF
BNE +6
JMP TESTX
INIT3: PRINT
QMARK
BR INIT2

;SET UP 'IA' VECTOR ADDRESS.
;PRINT ' ' TO INDICATE MONITOR READY
;WAIT FOR TTY ENTRY
;TEST FOR 'D'
;NOT 'D'
;YES, RUN 'DATA' TEST
;TEST FOR 'L'
;NOT 'L'
;YES, RUN 'LOGIC' TEST
;TEST FOR A "S"
;NOT AN S
;GO RUN SYNC TEST
;TEST FOR SUBTEST
;BRANCH IF NOT 'T'
;OTHERWISE RUN LOGIC SUBTEST
;ILLEGAL ENTRY
;TYPE '??'
;WAIT AGAIN

```

;*****
;ADF11 LOGIC TEST
;*****
    
```

```

001726 104011          SETUP:  INITAD          ;CALL ROUTINE TO INITIALIZE THE 'A/D'
001730 016703 177362   MOV      ADDR, ADDR3      ;LOAD R3 WITH ADDR ADDRESS
001734 016704 177346   MOV      ADCR, ADCR4      ;LOAD R4 WITH ADCR ADDRESS
001740 016705 177344   MOV      ADCSR, ADCSR5   ;LOAD R5 WITH ADCSR ADDRESS
001744 012767 002014 010736  MOV      #TST1, RETURN    ;SET UP RETURN ADDRESS FOR SCOPE
001752 012777 000340 177310  MOV      #340, PSM       ;SET PROCESSOR PRIORITY TO '7'
001760 005067 010722   CLR     SCOPEF          ;CLR SOFTWARE FLAG
001764 005067 012036   CLR     SOFLAG         ;CLR SOFTWARE FLAG
001770 005067 012062   CLR     MESPRT        ;CLR SOFTWARE FLAG
001774 000207   RTS     PC
    
```

```

001776 104000          LOGIC:  PRINT
002000 013321          MESS
002002 004767 177720  RESTRT: JSR     PC, SETUP    ;TEXT LOGIC
002006 012767 000030 010670  MOV      #30, ICOUNT    ;INITIALIZE LOGIC TEST
                                ;INITIALIZE LOGIC TEST
    
```

;TEST THAT THE 'CSR' WAS INITIALIZED CORRECTLY

```

002014 000240          TST1:   NOP
002016 104011          INITAD          ;CALL ROUTINE TO INITIALIZE THE 'A/D'
    
```

```

002020 104002          TST2:   SCOPE
002022 005715          TST      @ADCSR5
002024 001401          BEQ      .+4
002026 104400          ERROR
    
```

;TEST THAT THE 'STATUS WORD REGISTER' WAS CLEARED VIA INIT

```

002030 104002          TST3:   SCOPE
002032 005777 177256  TST      @ADSWR
002036 001401          BEQ      .+4
002040 104400          ERROR
    
```

;TEST THAT THE 'WORD REGISTER A' WAS CLEARED VIA INIT

```

002042 104002          TST4:   SCOPE
002044 005777 177250  TST      @ADWRA
002050 001401          BEQ      .+4
002052 104400          ERROR
    
```

;TEST THAT THE 'WORD REGISTER B' WAS CLEARED VIA INIT

```

002054 104002          TST5:   SCOPE
002056 005777 177240  TST      @ADWRB
002062 001401          BEQ      .+4
002064 104400          ERROR
    
```

;WRITE THE 'DATA WORD REGISTER A' WITH 1'S

```

002066 104002
002070 104011
002072 012777 177777 177220
002100 022777 177777 177212
002106 001401
002110 104400

```

```

TST6:  SCOPE
        INITAD           ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        MOV      #177777, @ADWRA ;WRITE ALL BITS
        CMP      #177777, @ADWRA
        BEQ      .+4
        ERROR

```

;WRITE THE 'DATA WORD REGISTER B' WITH 1'S

```

002112 104002
002114 104011
002116 012777 177777 177176
002124 022777 177777 177170
002132 001401
002134 104400

```

```

TST7:  SCOPE
        INITAD           ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        MOV      #177777, @ADWRB
        CMP      #177777, @ADWRB
        BEQ      .+4
        ERROR

```

;WRITE THE WORD COUNT REGISTER ALL 1'S

```

002136 104002
002140 104011
002142 012777 177777 177142
002150 022777 177777 177134
002156 001401
002160 104400

```

```

TST10: SCOPE
        INITAD           ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        MOV      #177777, @ADWCR
        CMP      #177777, @ADWCR
        BEQ      .+4
        ERROR

```

;WRITE THE STATUS WORD REGISTER ALL 1'S

```

002162 104002
002164 104011
002166 012777 177777 177120
002174 022777 177777 177112
002202 001401
002204 104400

```

```

TST11: SCOPE
        INITAD           ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        MOV      #177777, @ADSWR
        CMP      #177777, @ADSWR
        BEQ      .+4
        ERROR

```

;WRITE THE CONTROL & STATUS REGISTER WITH ALL 1'S EXCEPT 'GO' 'CLR FLAG' & INIT

```

002206 104002
002210 104011
002212 012715 153776
002216 022715 151774
002222 001401
002224 104400

```

```

TST12: SCOPE
        INITAD           ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        MOV      #153776, @ADCSRS
        CMP      #151774, @ADCSRS
        BEQ      .+4
        ERROR           ;*BEWARE* BIT '6' IS CLEARED VIA ERROR SUBROUTINE

```

;SELECT EXT SYN TO INHIBIT CONVERSION AND WRITE THE 'GO' BIT

002226 104002
002230 104011
002232 012714 004000
002236 012715 000001
002242 032715 000001
002246 001001
002250 104400

TST13: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
MOV #4000,ADCR4 ;SET 'EXT, SYNC'
MOV #1,ADCSRS ;SET 'GO'
BIT #1,ADCSRS ;TEST IF BIT SET.
BNE .+4
ERROR

;TEST THAT THE 'GO' IS CLEARED IN P.C. MODE WHEN THE DATA BUFFER IS READ.

002252 104002
002254 104011
002256 012714 005000
002262 005215
002264 005713
002266 032715 000001
002272 001401
002274 104400

TST14: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
MOV #5000,ADCR4 ;SET FINAL CH. & EXT. SYNC
INC ADCSR5 ;SET 'GO'
TST ADCBR3 ;READ DATA
BIT #1,ADCSRS ;TEST 'GO'
BEQ .+4 ;BRANCH IF CLR
ERROR ;READING DATA BUFFER IN P.C. MODE DIDN'T C. 'GO'

;TEST THAT 'A/D DONE' IS CLEARED WHEN DATA BUFFER IS READ

002276 104002
002300 104011
002302 052715 000200
002306 005713
002310 105715
002312 100001
002314 104400

TST15: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
BIS #200,ADCSRS ;SET DONE
TST ADCBR3 ;READ DATA BUFFER
TSTB ADCSR5 ;TEST DONE
BPL .+4 ;BRANCH IF CLR
ERROR ;READING DATA BUFFER DIDN'T CLR DONE

;SET XFER CHECK & WORDCOUNT CHECK (CSR BITS 3&8)
;AND CHECK THAT BIT 2 SETS.

002316 104002
002320 104011
002322 052715 000410
002326 032715 000004
002332 001001
002334 104400

TST16: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
BIS #410,ADCSRS ;SET WORDCOUNT CHECK AND XFER CHECK
BIT #4,ADCSRS ;TEST DONE
BNE .+4 ;BRANCH IF SET
ERROR ;SETTING W.C.CHECK & XFER CHECK DIDNT SET BIT 2

;SELECT EXT SYNC TO INHIBIT CONVERSION AND WRITE THE INITIAL CH.

002336 104002
002340 104011
002342 012714 004777
002346 022714 004777
002352 001401
002354 104400

TST17: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
MOV #4777,ADCR4
CMP #4777,ADCR4
BEQ .+4
ERROR

;TEST FOR WRITING THE 'CONTROL REG.' TO '0'

002356	104002	
002360	104011	
002362	052715	004000
002366	012714	175777
002372	042714	176777
002376	022714	001000
002402	001401	
002404	104400	
002406	005713	

```
TST20:  SCOPE
        INITAD                ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        BIS      #4000, @ADCSR5 ;CLR ALL A/D FLAGS
        MOV      #175777, @ADCR4
        BIC      #176777, @ADCR4 ;CLR ALL BITS EXCEPT FINAL
        CMP      #1000, @ADCR4
        BEQ      .+4
        ERROR
        TST      @ADDBR3      ;CAN'T CLR CONTROL REG.
                                ;CLR DONE
```

;INHIBIT A CONVERSION VIA SETTING EXT SYNC. & WRITE THE 'FINAL' BIT TO '0'

002410	104002	
002412	104011	
002414	052714	175777
002420	042714	173777
002424	022714	004000
002430	001401	
002432	104400	
002434	005713	

```
TST21:  SCOPE
        INITAD                ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        BIS      #175777, @ADCR4 ;WRITE ALL BITS BACK TO '1'
        BIC      #173777, @ADCR4
        CMP      #4000, @ADCR4
        BEQ      .+4
        ERROR
        TST      @ADDBR3      ;CLR DONE
```

;TEST FOR WRITING THE CSR TO '0'

002436 104002
002440 104011
002442 052714 004000
002446 052715 173777
002452 042715 173777
002456 011501
002460 042701 100000
002464 001401
002466 104400
002470 005713

TST22: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
BIS #4000, @ADCR4 ;SET EXTERNAL SYNC
BIS #173777, @ADCSR5 ;WRITE ALL BITS EXCEPT CLR FLAGS
BIC #173777, @ADCSR5
MOV @ADCSR5, R1
BIC #100000, R1 ;CLR ERROR BIT
BEQ .+4
ERROR
TST @ADDBR3 ;CLR DONE

;SET ALL WRITABLE 'CSR' BITS AND TEST CLEARING THEM WITH 'CLR FLAGS'

002472 104002
002474 104011
002476 052714 004000
002502 052715 153777
002506 052715 004000
002512 022715 000075
002516 001401
002520 104400
002522 005713

TST23: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
BIS #4000, @ADCR4 ;SET EXT, SYNC.
BIS #153777, @ADCSR5
BIS #4000, @ADCSR5 ;ISSUE CLR FLAGS
CMP #75, @ADCSR5 ;BITS '0,2,3,4,5' SHOULDN'T HAVE BEEN CLEARED
BEQ .+4
ERROR ;CLR FLAG DIDN'T CLR ALL 'CSR' FLAGS
TST @ADDBR3 ;CLR DONE

;TEST FOR SETTING THE 'ERROR' BIT VIA FORCING 'CONVERSION IN PROCESS'

002524 104002
002526 104011
002530 052715 040000
002534 005715
002536 100401
002540 104400

TST24: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
BIS #40000, @ADCSR5 ;SET 'CONV IN PRG'
TST @ADCSR5
BMI .+4
ERROR ;CONV IN PROGRESS DIDN'T SET 'ERROR BIT'

;TEST 'DATA OVERFLOW' FOR SETTING 'ERROR'

002542 104002
002544 104011
002546 052715 010000
002552 005715
002554 100401
002556 104400

TST25: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
BIS #10000, @ADCSR5 ;SET 'DATA OVFLOW'
TST @ADCSR5
BMI .+4
ERROR ;DATA OVFLOW DIDN'T SET 'ERROR BIT'

;TEST FOR SETTING THE 'ERROR' BIT IF IN P.C. SEQ. MODE W/ A/D DONE & FINAL SET

002560 104002
002562 104011
002564 052714 101000
002570 052715 001200
002574 005715
002576 100401
002600 104400
002602 005713

TST26: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
BIS #101000, @ADCR ;SELECT: PC, SEQ.
BIS #1200, @ADCSRS ;SET 'FINAL' & 'A/D DONE'
TST @ADCSRS ;TEST ERROR
BMI .+4 ;BRANCH IF SET
ERROR ;ERROR BIT DIDN'T SET FROM ABOVE SETUP
TST @ADDBR3 ;CLR DONE

;TEST THAT STATUS WORD ADDRESS REGISTER CAN BE CLEARED

002604 104002
002606 104011
002610 052777 177777 176476
002616 005077 176472
002622 005777 176466
002626 001401
002630 104400

TST27: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
BIS #177777, @ADSWR ;SET IT
CLR @ADSWR ;CLR IT
TST @ADSWR
BEQ .+4
ERROR ;CAN'T CLR A/D STATUS WORD REGISTER

;TEST THAT THE W.C. REGISTER CAN BE CLEARED

002632 104002
002634 052777 177777 176450
002642 005077 176444
002646 005777 176440
002652 001401
002654 104400
002656 052777 177777 176426
002664 052715 004000
002670 022777 177777 176414
002676 001401
002700 104400

TST30: SCOPE
BIS #177777, @ADWCR ;SET IT
CLR @ADWCR ;CLR IT
TST @ADWCR
BEQ .+4
ERROR ;CAN'T CLR A/D WORD COUNT REG.
BIS #177777, @ADWCR
BIS #4000, @ADCSRS
CMP #177777, @ADWCR
BEQ .+4
ERROR ;ISSUING 'CLR FLAGS' ALTERED WORD COUNT REG.

;TEST THAT DATA WORD ADDRESS REGISTER 'A' CAN BE CLEARED

002702 104002
002704 052777 177777 176406
002712 005077 176402
002716 005777 176376
002722 001401
002724 104400
002726 052777 177777 176364
002734 052715 004000
002740 022777 177777 176352
002746 001401
002750 104400

TST31: SCOPE
BIS #177777, @ADWRA ;SET IT
CLR @ADWRA ;CLEAR IT
TST @ADWRA
BEQ .+4
ERROR ;CAN'T CLR WORD ADDRESS REG 'A'
BIS #177777, @ADWRA
BIS #4000, @ADCSRS
CMP #177777, @ADWRA
BEQ .+4
ERROR ;ISSUING 'CLR FLAGS' ALTERED WORD REG. 'A'

;TEST THAT DATA WORD ADDRESS REGISTER 'B' CAN BE CLEARED

```
002752 104002
002754 052777 177777 176340
002762 005077 176334
002766 005777 176330
002772 001401
002774 104400
002776 052777 177777 176316
003004 052715 004000
003010 022777 177777 176304
003016 001401
003020 104400
```

```
TST32: SCOPE
BIS #177777, @ADWRB
CLR @ADWRB
TST @ADWRB
BEQ .+4 ;CAN'T CLR WORD ADDRESS REG. 'B'
ERROR
BIS #177777, @ADWRB
BIS #4000, @ADCSRS
CMP #177777, @ADWRB
BEQ .+4 ;ISSUING 'CLR FLAGS' ALTERED WORD REG. 'B'
ERROR
```

```
*****
;AT THIS POINT ALL REGISTERS HAVE BEEN TESTED TO 'READ/WRITE'
;THE NEXT SERIES OF TEST WILL TEST 'PROGRAM CONTROL' MODE
*****
```

```
003022 104002
003024 012777 177777 176260
003032 004767 005720
003036 012767 003044 007644
```

```
TST33: SCOPE
MOV #-1, @ADWCR ;LOAD W.C. '-1'
JSR PC, SETREG ;LOAD STATUS, WORD ADDRESSES A+B.
MOV @TSTXB, RETURN ;SET UP SCOPE ADDRESS
;TEST THAT AN INTERRUPT OCCURS IF READY IS SET
```

```
003044 000240
003046 052715 004000
003052 004767 006542
003056 003106
003060 012777 000140 176202
003066 052715 000300
003072 016701 010700
003076 005201
003100 001376
003102 104400
003104 000401
003106 022626
003110 004767 006540
```

```
TSTXB: NOP ;TEST STARTING ADDRESS
BIS #4000, @ADCSRS ;CLR ALL FLAGS
JSR PC, LDINTR ;LOAD INTERRUPT ADDRESS
TAGAA
MOV #140, @PSW ;SET PROCESSOR PRIORITY @3
BIS #300, @ADCSRS ;SET READY & INTR ENABLE
MOV DELAY1, R1
INC R1
BNE .-2 ;READY DIDN'T CAUSE INTR @ PRIORITY 3
ERROR
BR .+4
TAGAA: POP2SP ;RESTORE STACK POINTER
JSR PC, CLRINT ;CLR A/D INTR ADDRESS
```

;TEST THAT A/D DONE IS CLEARED ON THE INTERRUPT

003114	104002			TST34: SCOPE		
003116	104011			INITAD		;CALL ROUTINE TO INITIALIZE THE 'A/D'
003120	004767	006474		JSR PC,LDINTR		;LOAD INTERRUPT ADDRESS
003124	003154			TAGAAA		
003126	012777	000140	176134	MOV #140,2PSW		;SET PROCESSOR PRIORITY 23
003134	052715	000300		BIS #300,2ADCSRS		;SET A/D DONE & INTR ENABLE
003140	016701	010632		MOV DELAY1,R1		
003144	005201			INC R1		
003146	001376			BNE .-2		
003150	104400			ERROR		;A/D DONE DIDN'T CAUSE INTR 2 PRIORITY 3
003152	000401			BR .+4		
003154	022626			TAGAAA: POP2SP		;RESTORE STACK POINTER
003156	004767	006472		JSR PC,CLRINT		;CLR A/D INTR ADDRESS
003162	105715			TSTB 2ADCSRS		;TEST DONE
003164	100001			BPL .+4		;BRANCH IF CLR
003166	104400			ERROR		;A/D DONE WASN'T CLEARED VIA INTERRUPT

;TEST THAT SETTING THE 'ERROR BIT' WILL CAUSE AN INTERRUPT

003170	104002			TST35: SCOPE		
003172	104011			INITAD		;CALL ROUTINE TO INITIALIZE THE 'A/D'
003174	004767	006420		JSR PC,LDINTR		;LOAD INTERRUPT ADDRESS
003200	003230			TAGAB		
003202	052777	000140	176060	BIS #140,2PSW		;SET PROC. PRIORITY 23
003210	012715	040100		MOV #40100,2ADCSRS		;SET CONV. IN PROC. & INTR ENABLE
003214	016701	010556		MOV DELAY1,R1		
003220	005201			INC R1		
003222	001376			BNE .-2		
003224	104400			ERROR		;ERROR BIT DIDN'T CAUSE INTR
003226	000401			BR .+4		
003230	022626			TAGAB: POP2SP		;RESET STACK POINTER
003232	004767	006416		JSR PC,CLRINT		;CLR A/D INTR ADDRESS

;TEST THAT INTERRUPTS ARE INHIBITED WITH PROCESSOR PRIORITY 2 5

003236	104002			TST36: SCOPE		
003240	104011			INITAD		;CALL ROUTINE TO INITIALIZE THE 'A/D'
003242	004767	006352		JSR PC,LDINTR		;LOAD INTERRUPT ADDRESS
003246	003274			TAGAC		
003250	012777	000240	176012	MOV #240,2PSW		
003256	052715	000300		BIS #300,2ADCSRS		;SET DONE & INTR ENABLE
003262	016701	010510		MOV DELAY1,R1		
003266	005301			DEC R1		
003270	001376			BNE .-2		
003272	000402			BR .+6		
003274	022626			TAGAC: POP2SP		;RESET STACK POINTER
003276	104400			ERROR		;INTERRUPT OCCURRED 2 PROC. PRIORITY '5'
003300	004767	006350		JSR PC,CLRINT		;CLR A/D INTR ADDRESS

;TEST THAT INTERRUPTS ARE INHIBITED WITH PROCESSOR PRIORITY @ 4

003304	104002		TST37: SCOPE		
003306	104011		INITAD		:CALL ROUTINE TO INITIALIZE THE 'A/D'
003310	004767	006304	JSR	PC,LDINTR	:LOAD INTERRUPT ADDRESS
003314	003342		TAGAD		
003316	012777	000200	MOV	#200,@PSW	:SET PROC. PRIORITY @ 4
003324	052715	000300	BIS	#300,@ADCSRS	:SET A/D DONE & INTERRUPT ENABLE
003330	016701	010442	MOV	DELAY1,R1	
003334	005301		DEC	R1	
003336	001376		BNE	.-2	
003340	000402		BR	+.6	:OK, NO INTERRUPT OCCURRED
003342	022626		TAGAD: POP2SP		:RESET STACK
003344	104400		ERROR		:INTERRUPT OCCURRED W/ PROC. @ PRIORITY '4'
003346	004767	006302	JSR	PC,CLRINT	:CLR A/D INTR ADDRESS

;TEST FOR SETTING 'XFER ERROR' VIA SETTING 'XFER CHECK .' & 'WORD COUNT OFLO'

003352	104002		TST40: SCOPE		
003354	104011		INITAD		:CALL ROUTINE TO INITIALIZE THE 'A/D'
003356	052715	000411	BIS	#411,@ADCSRS	:SET W.C & XFER CHECK & GO
003362	032715	000004	BIT	#4,@ADCSRS	:TEST 'XFER ERROR'
003366	001001		BNE	+.4	:BRANCH IF SET
003370	104400		ERROR		: 'XFER ERR' DIDN'T SET W/XFER CHF & W.C SET

;TEST 'A/D DONE' TO SET VIA TAKING P.C CONVERSION

003372	104002		TST41: SCOPE		
003374	104011		INITAD		:CALL ROUTINE TO INITIALIZE THE 'A/D'
003376	012714	001777	MOV	#1777,@ADCR4	:LOAD FINAL CH
003402	012714	000005	MOV	#5,@ADCR4	:SELECT PC, INITIAL CH. '5'
003406	104007		CKDONE		:SUBROUTINE TO CHECK FOR 'A/D DONE'
003410	104400		ERROR		:A/D DONE FAILED TO SET (PC. MODE)
003412	005713		TST	@ADDBR3	:CLR DONE

;TEST 'A/D DONE' IS SET VIA STARTING A CONVERSION WITH 'GO'

003414	104002		TST42: SCOPE		
003416	104011		INITAD		:CALL ROUTINE TO INITIALIZE THE 'A/D'
003420	005215		INC	@ADCSRS	:SET 'GO'
003422	104007		CKDONE		:SUBROUTINE TO CHECK FOR 'A/D DONE'
003424	104400		ERROR		:SETTING GO DIDN'T SET DONE (PC. MODE)
003426	005713		TST	@ADDBR3	:CLR DONE

;TEST THAT LEAVING THE 'GO' BIT SET DOESN'T ALLOW CONTINUOUS CONVERSIONS

```
003430 104002
003432 104011
003434 005215
003436 105715
003440 100376
003442 005777 175650
003446 104007
003450 000401
003452 104400
003454 005713
```

```
TST43: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
INC @ADCSRS ;SET GO
TSTB @ADCSRS ;WAIT FOR DONE
BPL -2
TST @ADDBR ;CLR DONE
CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
BR .+4 ;OK, DONE DIDN'T SET
ERROR ;GO ENABLES CONTINUOUS P.C CONVERSIONS
TST @ADDBR3 ;CLR DONE
```

;TEST THAT A CONVERSION IS INHIBITED IF 'XFER CHK' & 'W.C' & 'GO' ARE SET

```
003456 104002
003460 104011
003462 052714 001000
003466 052715 000411
003472 042715 000200
003476 012714 000010
003502 104007
003504 000401
003506 104400
003510 005713
```

```
TST44: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
BIS #1000,@ADCR4 ;SET FINAL TO INHIBIT 'GO'
BIS #411,@ADCSRS ;SET W.C & XFER CHK & GO
BIC #200,@ADCSRS ;CLR DONE
MOV #10,@ADCR4 ;START CONVERSION
CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
BR .+4 ;OK, DONE DIDN'T SET
ERROR ;A/D DONE SET W/XFER CHK & W.C SET
TST @ADDBR3 ;CLR DONE
```

;TEST THAT THE CONVERTER CONTINUES IF 'XFER CHK & W.C & GO' WERE SET, THEN CLR 'XFER'

```
003512 104002
003514 104011
003516 052714 001000
003522 052715 000411
003526 042715 000200
003532 012714 000055
003536 000240
003540 000240
003542 042715 000010

003546 104007
003550 104400
003552 005713
```

```
TST45: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
BIS #1000,@ADCR4 ;SET FINAL TO INHIBIT 'GO'
BIS #411,@ADCSRS ;SET XFER CHK & W.C. & GO
BIC #200,@ADCSRS ;CLR DONE, (SET VIA SETTING W.C.)
MOV #55,@ADCR4 ;ISSUE FALSE START
NOP
NOP
BIC #10,@ADCSRS ;CLR 'XFER CHK' (A/D SHOULD CONTINUE)

CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
ERROR ;A/D DIDN'T CONTINUE AFTER CLR 'XFER CHK'
TST @ADDBR3 ;CLR DONE
```

;TEST THAT CONVERSIONS ARE INHIBITED IF 'W.C' IS SET AND 'GO' IS CLEARED.

```

003554 104002
003556 104011
003560 052715 000400
003564 042715 000200
003570 012714 040075
003574 104007
003576 000401
003600 104400
003602 104011

```

```

TST46: SCOPE
        INITAD
        BIS      #400, @ADCSRS
        BIC      #200, @ADCSRS
        MOV      #40075, @ADCR4
        CKDONE
        BR       .+4
        ERROR
        INITAD

```

```

:CALL ROUTINE TO INITIALIZE THE 'A/D'
:SET 'W.C' FLAG, GO IS CLR
:CLR DONE
:START CONVERSION
:SUBROUTINE TO CHECK FOR 'A/D DONE'
:OK, DONE DIDN'T SET
:A/D DONE SET W/'W.C' SET & 'GO' CLR.
:CALL ROUTINE TO INITIALIZE THE 'A/D'

```

;TEST THAT SETTING 'GO' IN 'DMA' MODE DOESN'T ENABLE CONVERSIONS

```

003604 104002
003606 104011
003610 052715 000400
003614 042715 000200
003620 012777 177777 175464
003626 004767 005124
003632 012714 111000
003636 005215
003640 104007
003642 000401
003644 104400
003646 005713

```

```

TST47: SCOPE
        INITAD
        BIS      #400, @ADCSRS
        BIC      #200, @ADCSRS
        MOV      #-1, @ADMCR
        JSR      PC, SETREG
        MOV      #111000, @ADCR4
        INC      @ADCSRS
        CKDONE
        BR       .+4
        ERROR
        TST      @ADDBR3

```

```

:CALL ROUTINE TO INITIALIZE THE 'A/D'
:SET W.C. TO INHIBIT CONVRT
:CLR DONE, SET VIA SETTING W.C.
:PRECAUTIONARY DMA SETUP
:SET UP A/D REG'S.
:SELECT; SEQ., DMA, FINAL CH.
:SET 'GO'
:SUBROUTINE TO CHECK FOR 'A/D DONE'
:OK, DONE DIDN'T SET
:SETTING 'GO' STARTED CONVRT IN 'DMA MODE'
:CLR DONE

```

;TEST THAT CONVERSIONS ARE INHIBITED WITH 'EXT. SYNC'. SELECTED

```

003650 104002
003652 104011
003654 012714 064000
003660 104007
003662 000401
003664 104400
003666 005713

```

```

TST50: SCOPE
        INITAD
        MOV      #64000, @ADCR4
        CKDONE
        BR       .+4
        ERROR
        TST      @ADDBR3

```

```

:CALL ROUTINE TO INITIALIZE THE 'A/D'
:SELECT P.C. EXT SYNC.
:SUBROUTINE TO CHECK FOR 'A/D DONE'
:OK, DONE DIDN'T SET
:A/D DONE SET WITH 'EXT. SYN' SELECTED
:CLR DONE

```

;TEST 'A/D WAIT' VIA TAKING 4 CONSECUTIVE CONVERSIONS WITHOUT READING DATA.

```
003670 104002
003672 104011
003674 012702 000004
003700 012714 060007
003704 104007
003706 104400
003710 032715 002000
003714 001401
003716 104400
003720 042715 000200
003724 005302
003726 001364
```

```
TST51: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
MOV #4,R2
TAGF: MOV #60007,ADCR4 ;START CONVERSION
CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
ERROR ;A/D DONE FAILED TO SET
BIT #2000,ADCSRS ;TEST A/D WAIT
BEQ .+4 ;BRANCH IF NOT SET
ERROR ;A/D WAIT SET BEFORE DATA BUFFER'S FULL
BIC #200,ADCSRS ;CLR 'A/D DONE'
DEC R2 ;DEC CNTR.
BNE TAGF ;BRANCH IF HAVEN'T TAKEN '4' CONVERSIONS
```

;TEST THAT 'A/D WAIT' WILL SET ON THE 5TH CONVERSION AND INHIBIT FUTHER CONVERSIONS

```
003730 012714 060007
003734 016701 010036
003740 005201
003742 001376
003744 032715 002000
003750 001001
003752 104400
```

```
TAGG: MOV #60007,ADCR4 ;START 5TH CONVERSION
MOV DELAY1,R1
INC R1
BNE .-2
BIT #2000,ADCSRS ;TEST A/D WAIT
BNE .+4 ;BRANCH IF SET
ERROR ;5 CONSECT P.C. CONVERSIONS DIDN'T SET A/D WAIT
```

;TEST THAT 'WAIT' INHIBITED 5TH CONVERSION

```
003754 105715
003756 100001
003760 104400
```

```
TSTB ADCSRS ;TEST DONE
BPL .+4 ;BRANCH IF CLR
ERROR ;A/D DONE SET WITH A/D WAIT SET.
```

;TEST THAT 'A/D WAIT' SET THE ERROR BIT

```
003762 005715
003764 100401
003766 104400
```

```
TST ADCSRS ;TEST ERROR
BMI .+4 ;BRANCH IF SET
ERROR ;A/D WAIT DIDN'T SET ERROR BIT
```

;TEST THAT 'A/D WAIT' CAN BE CLEARED VIA 'CLR FLAGS'.

```
003770 052715 004000
003774 032715 002000
004000 001401
004002 104400
```

```
BIS #4000,ADCSRS ;CLR FLAGS
BIT #2000,ADCSRS ;TEST A/D WAIT
BEQ .+4
ERROR ;CLR FLAGS DIDN'T CLR 'A/D WAIT'
```

;TEST THAT 'A/D WAIT' IS CLEARED VIA READING THE DATA BUFFER

004004	104002		TST52:	SCOPE		
004006	104011			INITAD		;CALL ROUTINE TO INITIALIZE THE 'A/D'
004010	012701	000005		MOV	#5,R1	;TAKE FIVE CONSECUTIVE CONVERSIONS
004014	013714	060007	TAGXF:	MOV	#60007, @ADCR4	;START CONVERSION
004020	016702	007752		MOV	DELAY1,R2	
004024	005202			INC	R2	;GIVE FLAG CHANCE TO SET
004026	001376			BNE	.-2	
004030	005301			DEC	R1	
004032	001370			BNE	TAGXF	
004034	032715	002000		BIT	#2000, @ADCSR5	;TEST A/D WAIT
004040	001001			BNE	.-4	;BRANCH IF SET
004042	104400			ERROR		;A/D WAIT FAILED TO SET ON 5TH CONVERSION
004044	005713			TST	@ADDBR3	;READING DATA SHOULD ALLOW 5TH CONVERSION
004046	032715	002000		BIT	#2000, @ADCSR5	;RE-TEST 'WAIT'
004052	001401			BEQ	.-4	
004054	104400			ERROR		;READING DATA BUFFER DIDN'T CLR WAIT

;TEST THAT A 5TH CONVERSION WILL BE TAKEN AFTER READING DATA

004056	016702	007714		MOV	DELAY1,R2	
004062	005202			INC	R2	;GIVE DONE A CHANCE TO SET
004064	001376			BNE	.-2	
004066	105715			TSTB	@ADCSR5	;TEST DONE
004070	100401			BMI	.-4	;BRANCH IF SET
004072	104400			ERROR		;COULDN'T TAKE CONVERSION AFTER READING DATA
004074	005713			TST	@ADDBR3	;CLR DONE

;TEST THE 'CR' UPDATE LOGIC VIA WRITTING '4' WORDS INTO THE 'CR'

004076	104002		TST53:	SCOPE		
004100	104011			INITAD		;CALL ROUTINE TO INITIALIZE THE 'A/D'
004102	012702	000004		MOV	#4,R2	;SET UP TO TAKE '4' CONVTS TO FILL DATA BUFFER
004106	005014		TAGXL:	CLR	@ADCR4	;ST. CONVERSION
004110	104007			CKDONE		;SUBROUTINE TO CHECK FOR 'A/D DONE'
004112	104400			ERROR		;A/D DONE DIDN'T SET
004114	042715	000200		BIC	#200, @ADCSR5	;CLR DONE
004120	005302			DEC	R2	
004122	001371			BNE	TAGXL	;ST. '4' CONVERSIONS

;THE NEXT '4' LOADS SHOULD RE-LOAD THE 'CR' BUFFERS

004124	012714	000525		MOV	#525, @ADCR4	;SELECT; SEQ. SYNC. CH. '525'
004130	012714	000252		MOV	#252, @ADCR4	;SELECT; SEQ. SYNC. CH. '252'
004134	012714	000000		MOV	#000, @ADCR4	;SELECT; SEQ. SYNC. CH. '000'
004140	012714	000777		MOV	#777, @ADCR4	;SELECT; SEQ. SYNC. CH. '777'

;ATTEMPT TO READ THE '4' STORED CONTROL WRD'S VIA UPDATING THE
;CONTROL BUFFER'S WITH A 'READ DATA' COMMAND

004144	022714	000525	CMP	#525, @ADCR4	; SHOULD = 1ST WORD WRITTEN
004150	001401		BEQ	.+4	
004152	104400		ERROR		; DATA IS NOT = TO 1ST WORD WRITTEN
004154	005713		TST	@ADDBR3	; READ DATA TO UPDATE BUFFER
004156	016702	007614	MOV	DELAY1, R2	
004162	005302		DEC	R2	
004164	001376		BNE	.-2	
004166	022714	000252	CMP	#252, @ADCR4	; SHOULD = 2ND WORD WRITTEN
004172	001401		BEQ	.+4	
004174	104400		ERROR		; DATA IS NOT = TO 2ND WORD WRITTEN
004176	005713		TST	@ADDBR3	; READ DATA TO UPDATA BUFFER
004200	016702	007572	MOV	DELAY1, R2	
004204	005302		DEC	R2	
004206	001376		BNE	.-2	
004210	005714		TST	@ADCR4	; SHOULD = 3RD WORD WRITTEN
004212	001401		BEQ	.+4	
004214	104400		ERROR		; DATA IS NOT = TO 3RD WORD WRITTEN
004216	005713		TST	@ADDBR3	; READ DATA TO UPDATE BUFFER
004220	016702	007552	MOV	DELAY1, R2	
004224	005302		DEC	R2	
004226	001376		BNE	.-2	
004230	022714	000777	CMP	#777, @ADCR4	; SHOULD = 4TH WORD WRITTEN
004234	001401		BEQ	.+4	
004236	104400		ERROR		; DATA IS NOT = TO 4TH WORD WRITTEN

;ATTEMPT TO WRITE '5' WORDS INTO THE CONTROL REG. WITHOUT READING
;DATA AND TEST THAT THE '5TH' WORD IS LOCKED OUT AND THAT IT DOESN'T
;WRITE OVER THE '4TH' WORD WRITTEN.

004240	104002		TST54:	SCOPE	
004242	104011			INITAD	
004244	012702	000004		MOV	#4 R2
004250	005014		TAGXK:	CLR	@ADCR4
004252	104007			CKDONE	
004254	104400			ERROR	
004256	042715	000200		BIC	#200, @ADCSR5
004262	005302			DEC	R2
004264	001371			BNE	TAGXK
004266	012714	000525		MOV	#525, @ADCR4
004272	012714	000252		MOV	#252, @ADCR4
004276	005014			CLR	@ADCR4
004300	012714	000777		MOV	#777, @ADCR4
004304	005014			CLR	@ADCR4

;CALL ROUTINE TO INITIALIZE THE 'A/D'
;SET UP TO TAKE '4' CONVTS TO FILL DATA BUFFER
;ST. CONVERSION
;SUBROUTINE TO CHECK FOR 'A/D DONE'
;A/D DONE DIDN'T SET
;CLR DONE

;ST. '4' CONVERSIONS
;SELECT: CH. '525'
;SELECT: CH. '252'
;SELECT: CH. '000'
;SELECT: CH. '777'
;SELECT: CH. '000'

004306 012700 000003
004312 005713
004314 016702 007456
004320 005302
004322 001376
004324 005300
004326 001371
004330 022714 000777
004334 001401
004336 104400

```

;ALL 5 WORDS HAVE BEEN WRITTEN
TAGXM:  MOV      #3,R0
        TST      @ADDBR3      ;ISSUE 3 READS TO UPDATE BUFFER
        MOV      DELAY1,R2
        DEC      R2
        BNE      #-2
        DEC      R0
        BNE      TAGXM
        CMP      #777,@ADCR4  ;SHOULD = 4TH WORD WRITTEN
        BEQ      .+4
        ERROR
;DATA IS NOT = TO 4TH WORD WRITTEN
    
```

;TEST FOR TAKING A CONVERSION WITH SINGLE CH. SELECTED.

004340 104002
004342 104011
004344 012714 101027
004350 012714 160027
004354 104007
004356 104400
004360 005713

```

TST55: SCOPE
        INITAD
        MOV      #101027,@ADCR4 ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        MOV      #160027,@ADCR4 ;SELECT FINAL
        CKDONE   ;LOAD INITIAL & ST.
        ERROR    ;SUBROUTINE TO CHECK FOR 'A/D DONE'
        TST      @ADDBR3        ;SINGLE CH. SELECTED, DIDN'T SET DONE
        ;CLR DONE
    
```

;TEST THAT 'FINAL CH.' FLAG WAS NOT SET WITH SINGLE CH. SELECTED

004362 032715 001000
004366 001401
004370 104400

```

        BIT      #1000,@ADCSRS ;TEST FINAL CH.
        BEQ      .+4
        ERROR    ;SINGLE CH SET FINAL CH. FLAG.
    
```

;TEST THAT INITIAL CH. DOESNT GET INCREMENTED IN SINGLE CH. MODE

004372 104002
004374 104011
004376 012714 101077
004402 012714 100077
004406 104007
004410 104400
004412 005713
004414 005215
004416 104007
004420 104400
004422 005713
004424 022714 100077
004430 001401
004432 104400

```

TST56: SCOPE
        INITAD
        MOV      #101077,@ADCR4 ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        MOV      #100077,@ADCR4 ;LOAD FINAL
        CKDONE   ;LOAD INITIAL CH.
        ERROR    ;SUBROUTINE TO CHECK FOR 'A/D DONE'
        TST      @ADDBR3        ;A/D DONE FAILED TO SET
        INC      @ADCSRS        ;CLR DONE
        CKDONE   ;SET 'GO', ST 2ND CONVERSION
        ERROR    ;SUBROUTINE TO CHECK FOR 'A/D DONE'
        TST      @ADDBR3        ;A/D DONE FAILED TO SET
        CMP      #100077,@ADCR4 ;CLR DONE
        BEQ      .+4            ;CH. CHANGED?
        ERROR    ;INITIAL CH. REG WAS UPDATED IN SINGLE CH
    
```

;TEST THAT THE FINAL CH. FLAG IS SET WHEN INITIAL = FINAL

```
004434 104002
004436 104011
004440 012714 101001
004444 012714 100000
004450 104007
004452 104400
004454 005713
004456 032715 001000
004462 001401
004464 104400
004466 005215
004470 104007
004472 104400
004474 032715 001000
004500 001001
004502 104400
004504 005713
```

```
TST57: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
MOV #101001, @ADCR4 ;LOAD FINAL CH. =1
MOV #100000, @ADCR4 ;LOAD INITIAL CH.=0
CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
ERROR ;DONE FAILED TO SET
TST @ADDBR3 ;CLR DONE
BIT #1000, @ADCSR5 ;TEST FINAL CH. FLAG
BEQ .+4 ;CONTINUE IF NOT SET
ERROR ;FINAL CH. FLAG SET BEFORE FINAL CH. WAS REACHED
INC @ADCSR5 ;ST 2ND CONVERSION
CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
ERROR ;DONE FAILED TO SET
BIT #1000, @ADCSR5 ;TEST FINAL CH.
BNE .+4 ;BR IF SET
ERROR ;FINAL CH. FLAG DIDN'T SET W/ INITIAL = FINAL
TST @ADDBR3 ;CLR DONE
```

;TEST THAT INITIAL CH. ADDRESS GETS UPDATED AFTER EACH NON SINGLE CH. CONVERSION

```
004506 104002
004510 104011
004512 012714 001777
004516 012714 100000
004522 012702 000776
004526 104007
004530 104400
004532 005713
004534 005302
004536 001402
004540 005215
004542 000771
```

```
TST60: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
MOV #1777, @ADCR4 ;LOAD FINAL CH: 777
MOV #100000, @ADCR4 ;LOAD INITIAL CH: '0'
MOV #776, R2 ;SET UP TO '776' CONVERSIONS
CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
ERROR ;A/D DONE FAILED TO SET
TST @ADDBR3 ;CLR DONE
DEC R2
BEQ TAGM ;BRANCH IF TAKEN ALL CONVERSIONS
INC @ADCSR5 ;OTHERWISE START NEXT CONVERSION
BR TAGL
```

```
004544 011501
004546 011402
004550 042702 177000
004554 022702 000776
004560 001401
004562 104400
```

```
TAGM:  MOV 2ADCSR5,R1 ;SAVE CONTENTS OF 'CSR'
        MOV 2ADCR4,R2
        BIC #177000,R2 ;CLR ALL BUT CH. BITS
        CMP #776,R2 ;DOES INITIAL = FINAL
        BEQ .+4 ;BRANCH IF YES
        ERROR ;INITIAL CH. REG WASN'T INCREMENTED
```

```
;TEST THAT THE 'INITIAL CH. REG. IS RESET AFTER REACHING THE FINAL CH.
```

```
004564 104002
004566 104011
004570 012714 001007
004574 012714 160005
004600 012702 000003
004604 104007
004606 104400
004610 005713
004612 005302
004614 001402
004616 005215
004620 000771
```

```
TST61: SCOPE
        INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        MOV #1007,2ADCR4 ;SET FINAL CH. = 7
        MOV #160005,2ADCR4 ;SET INITIAL CH = 5
        MOV #3,R2
```

```
TAG0:  CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
        ERROR ;A/D DONE FAILED TO SET
        TST 2ADDBR3 ;CLR 'GO' BIT
        DEC R2
        BEQ TAGP
        INC 2ADCSR5 ;SET 'GO'
        BR TAG0
```

```
004622 011401
004624 042701 177000
004630 022701 000005
004634 001401
004636 104400
```

```
TAGP:  MOV 2ADCR4,R1
        BIC #177000,R1 ;CLR UNWANTED BITS
        CMP #5,R1 ;SHOULD OF RESET TO '5'
        BEQ .+4 ;BRANCH IF YES
        ERROR ;INITIAL CH. REG. WASN'T RESET AFTER REACHING FINAL
```

```
;TEST THAT 'INITIAL CH.' REG IS NOT INCREMENTED IN RANDOM MODE
```

```
004640 104002
004642 104011
004644 012714 060005
004650 104007
004652 104400
004654 005713
004656 011401
004660 042701 177000
004664 022701 000005
004670 001401
004672 104400
```

```
TST62: SCOPE
        INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        MOV #60005,2ADCR4 ;LOAD INITIAL CH.
        CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
        ERROR ;A/D DONE FAILED TO SET
        TST 2ADDBR3 ;CLR DONE
        MOV 2ADCR4,R1
        BIC #177000,R1 ;CLR ALL BUT CH. BITS
        CMP #5,R1 ;CH. STILL SHOULD EQUAL '5'
        BEQ .+4 ;BRANCH IF YES
        ERROR ;INITIAL CH. REG WAS UPDATED IN RANDOM MODE
```

:TEST THE 'SEQUENTIAL DMA' MODE WITH WITH 'XFER CHK' SET TO STOP ON WC.
:*****

004674 104002
004676 104011
004700 012777 177777 174404
004706 004767 004044
004712 012714 101000
004716 012714 110000
004722 104007
004724 104400

TST63: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
MOV #-1,ADWCR ;SET UP TO TAKE '1' CONVERSIONS
JSR PC,SETREG ;LOAD A/D REG.S
MOV #101000,ADCR4 ;LOAD FINAL CH.=0
MOV #110000,ADCR4 ;SELECT: SEQ. DMA, CH.0
CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
ERROR ;W.C. OVRFLO DIDN'T SET DONE IN SEQ. MODE

;TEST THAT THE 'W.C.' REG. IS RESET TO '-1' ON THE W.C. OVFLO

004726 104002
004730 104011
004732 012777 177777 174352
004740 004767 004012
004744 012715 000010
004750 012714 001007
004754 012714 110000
004760 104007
004762 104400
004764 022777 177777 174320
004772 001401
004774 104400

TST64: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
MOV #-1,ADWCR ;SET UP TO TAKE '1' CONVERSIONS
JSR PC,SETREG ;LOAD A/D REG.S
MOV #10,ADCSR5 ;SET 'XFER CHK' TO HALT ON WC. OVRFLO
MOV #1007,ADCR4 ;LOAD FINAL CH.=7
MOV #110000,ADCR4 ;SELECT: SEQ. DMA, CH.0
CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
ERROR ;W.C. OVRFLO DIDN'T SET DONE IN SEQ. MODE
CMP #-1,ADWCR ;WAS THE 'W.C.' REG RESET
BEQ .+4
ERROR ;W.C. OVRFLO DIDN'T RESET W.C. REG.

;TEST THAT 'DATA WORD REG. A' IS RESET ON W.C. OVFLO

004776 104002
005000 104011
005002 012777 177777 174302
005010 004767 003742
005014 012715 000010
005020 012714 001007
005024 012714 110000
005030 104007
005032 104400
005034 022777 015072 174256
005042 001401
005044 104400

TST65: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
MOV #-1,ADWCR ;SET UP TO TAKE '1' CONVERSIONS
JSR PC,SETREG ;LOAD A/D REG.S
MOV #10,ADCSR5 ;SET 'XFER CHK' TO HALT ON WC. OVRFLO
MOV #1007,ADCR4 ;LOAD FINAL CH.=7
MOV #110000,ADCR4 ;SELECT: SEQ. DMA, CH.0
CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
ERROR ;W.C. OVRFLO DIDN'T SET DONE IN SEQ. MODE
CMP #ADBUFF,ADWRA ;WAS WORD REG. A RESET
BEQ .+4
ERROR ;DATA WRD REG. 'A' WASN'T RESET ON 'W.C.' OVRFLO

;TEST THAT 'DATA WORD REG. B' IS RESET ON W.C. OVRFLO

005046	104002			TST66:	SCOPE		
005050	104011				INITAD		;CALL ROUTINE TO INITIALIZE THE 'A/D'
005052	012777	177777	174232		MOV	#-1, @ADWCR	;SET UP TO TAKE '1' CONVERSIONS
005060	004767	003672			JSR	PC, SETREG	;LOAD A/D REG.S
005064	012715	000010			MOV	#10, @ADCSR5	;SET 'XFER CHK' TO HALT ON WC. OVRFLO
005070	012714	001007			MOV	#1007, @ADCR4	;LOAD FINAL CH.=7
005074	012714	110000			MOV	#110000, @ADCR4	;SELECT: SEQ., DMA, CH.0
005100	104007				CKDONE		;SUBROUTINE TO CHECK FOR 'A/D DONE'
005102	104400				ERROR		;W.C. OVRFLO DIDN'T SET DONE IN SEQ. MODE
005104	022777	015074	174210		CMP	#ADBUF+2, @ADWRB	;WAS WORD REG. B RESET
005112	001401				BEQ	.+4	
005114	104400				ERROR		;DATA WRD REG. 'B' WASN'T RESET ON 'W.C.' OVRFLO

;TEST THAT THE 'STATUS WORD REG.' IS RESET ON W.C. OVRFLO

005116	104002			TST67:	SCOPE		
005120	104011				INITAD		;CALL ROUTINE TO INITIALIZE THE 'A/D'
005122	012777	177777	174162		MOV	#-1, @ADWCR	;SET UP TO TAKE '1' CONVERSIONS
005130	004767	003622			JSR	PC, SETREG	;LOAD A/D REG.S
005134	012715	000010			MOV	#10, @ADCSR5	;SET 'XFER CHK' TO HALT ON WC. OVRFLO
005140	012714	001007			MOV	#1007, @ADCR4	;LOAD FINAL CH.=7
005144	012714	110000			MOV	#110000, @ADCR4	;SELECT: SEQ., DMA, CH.0
005150	104007				CKDONE		;SUBROUTINE TO CHECK FOR 'A/D DONE'
005152	104400				ERROR		;W.C. OVRFLO DIDN'T SET DONE IN SEQ. MODE
005154	022777	014070	174132		CMP	#RANBUF, @ADSWR	;WAS STATUS REG. RESET+++++
005162	001401				BEQ	.+4	
005164	104400				ERROR		;STATUS WRDREG. WASN'T RESET VIA W.C. OVRFLD

;TEST THAT DATA WORD ADDRESS 'A' GETS MODIFIED VIA THE 'DMA' TRANSFER

005166	104002			TST70:	SCOPE		
005170	104011				INITAD		;CALL ROUTINE TO INITIALIZE THE 'A/D'
005172	012777	177770	174112		MOV	#-10, @ADWCR	;SET UP TO TAKE '8' CONVERSIONS
005200	004767	003552			JSR	PC, SETREG	;LOAD A/D REG.S
005204	012700	177760			MOV	#-20, R0	
005210	012702	015072			MOV	#ADBUF, R2	;PRE-LOAD DATA BUFFERS 'A&B'
005214	012722	125252		TAGAF:	MOV	#125252, (R2)+	
005220	005200				INC	R0	
005222	001374				BNE	TAGAF	
005224	012715	000010			MOV	#10, @ADCSR5	;SET 'XFER CHK' TO HALT ON WC. OVRFLO
005230	012714	001007			MOV	#1007, @ADCR4	;LOAD FINAL CH.=7
005234	012714	110000			MOV	#110000, @ADCR4	;SELECT: SEQ., DMA, CH.0
005240	104007				CKDONE		;SUBROUTINE TO CHECK FOR 'A/D DONE'
005242	104400				ERROR		;W.C. OVRFLO DIDN'T SET IN SEQ. MODE

;CHECK THAT DATA BUFFER 'A' WAS MODIFIED VIA DMA

005244 012700 177770
 005250 012701 015072
 005254 022721 125252
 005260 001002
 005262 104400
 005264 000402
 005266 005200
 005270 001371

TAGAG: MOV #-10,R0
 MOV #ADBUFF,R1
 CMP #125252,(R1)+ ;CHECK IF ADDRESS WAS MODIFIED
 BNE .+6
 ERROR ;MEMORY WASN'T MODIFIED VIA SEQ. DMA
 BR .+6 ;EXIT ON ERROR
 INC R0
 BNE TAGAG

;TEST THAT ONLY 'B' LOCATIONS WERE MODIFIED WITH WC.=-10 &'XFER CHK' SET

005272 012700 177770
 005276 012701 015112
 005302 022721 125252
 005306 001402
 005310 104400
 005312 000402
 005314 005200
 005316 001371

TAGAH: MOV #-10,R0
 MOV #ADBUFF+20,R1
 CMP #125252,(R1)+ ;CHECK BUFFER 'B'
 BEQ .+6 ;BRANCH IF UNMODIFIED
 ERROR ;DATA BUFFER 'B' WAS MODIFIED W/ 'XFER CHK' SET
 BR .+6 ;EXIT ON ERROR
 INC R0
 BNE TAGAH

;TEST THAT BOTH BUFFERS 'A&B' ARE MODIFIED IF 'XFER CHK' IS CLR & 'GO' IS SET

```

005320 104002
005322 104011
005324 012777 177700 173760
005332 004767 003420
005336 062777 000100 173756
005344 012701 015072
005350 012700 177500
005354 012721 152525
005360 005200
005362 001374
005364 004767 004230
005370 005424
005372 012714 001777
005376 005215
005400 012714 110000
005404 012700 177740
005410 005200
005412 001376
005414 012767 125252 007450
005422 000001
    
```

```

TST71: SCOPE
        INITAD
        MOV #100,ADWCR ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        JSR PC,SETREG ;SET UP FOR '200' CONVERSION; A-100,B-100
        ADD #100,ADWAB ;LOAD A/D REG.S
        MOV #A0BUFF,R1 ;OFFSET 'B' 200 LOCATION FROM 'A'
        MOV #300,R0 ;THIS WILL GIVE A '100' BYTE (32 WRD) GAP
        TAGAI: MOV #152525,(R1)+ ;BETWEEN THE END OF 'A' & ST. OF 'B'
        INC R0 ;PRE-LOAD BUFFER AREA
        BNE TAGAI
        JSR PC,LDINTR ;LOAD A/D INTR. ADDR.
        TAGAJ ;INTR HERE
        MOV #1777,ADCR4 ;LOAD FINAL CH.
        INC ADCSRS ;SET 'GO'
        MOV #110000,ADCR4 ;SELECT: SEQ.,DMA, INITIAL CH.
        MOV #40,R0 ;LET A/D COMPLETE A FEW CONVERSIONS
        INC R0 ;THEN RELOAD BUFFER 'A'
        BNE .-2 ;TO CHK THAT IT IS NOT RE-MODIFIED
        MOV #125252,A0BUFF ;RELOAD 'A' BUFFER WITH KNOWN VALUE
        WAIT ;WAIT FOR INTERRUPT
    
```

;ENTER HERE ON INTERRUPT

```

005424 052715 004000
005430 105715
005432 100376
005434 004767 004214
005440 022626
    
```

```

TAGAJ: BIS #4000,ADCSRS ;CLR ALL FLAGS
        TSTB ADCSRS ;TEST FOR 2ND W.C. OVRFLO
        BPL .-2
        JSR PC,CLRINT ;CLR OUT INTERRUPT
        POP2SP ;RESET STACK POINTER
    
```

;TEST THAT BUFFER 'A' WASN'T REMODIFIED AFTER CLEARING 'GO'

```

005442 022767 125252 007422
005450 001401
005452 104400
    
```

```

        CMP #125252,A0BUFF ;EXAMINE BUFFER
        BEQ .+4 ;BRANCH IF NOT MODIFIED
        ERROR ;'STOP' DIDN'T STOP A/D
    
```

;TEST THAT THE '32' WRD GAP WASN'T MODIFIED

```

005454 012700 177741
005460 012701 015272
005464 022721 152525
005470 001402
005472 104400
005474 000402
005476 005200
005500 001371
    
```

```

TAGXJ: MOV #37,R0 ;LOAD ST. ADDR OF GAP
        MOV #A0BUFF+200,R1 ;TEST LOCATION
        CMP #152525,(R1)+
        SEQ .+6
        ERROR ;THE '32' WRD GAP AFTER WRD 'A' WAS MODIFIED
        BR .+6 ;EXIT ON ERROR
        INC R0
        BNE TAGXJ
    
```

;TEST THAT BUFFER 'B' WAS MODIFIED

005502 012700 000100
005506 012701 015372
005512 022721 152525
005516 001002
005520 104400
005522 000402
005524 005300
005526 001371

TAGAK: MOV #100,RO ;LOAD BUFFER 'B' STARTING ADDRESS
MOV #A0B(FF+303),R1 ;WAS ADDRESS MODIFIED?
CMP #152525,(R1)+ ;BRANCH IF YES
BNE .+6 ;BUFFER 'B' WASN'T MODIFIED
ERP,OR ;EXIT ON ERROR
BR .+6
DEC RO
BNE TAGAK

;TEST THAT THE INITIAL CH. REG. CONTAINS '177' AFTER TAKING '200' CONVERSIONS

005530 104002
005532 104011
005534 012777 177700 173550
005542 004767 003210
005546 004767 004046
005552 005570
005554 012714 001777
005560 005215
005562 012714 110000
005566 000001

TST72: SCOPE ;CALL ROUTINE TO INITIALIZE THE 'A/D'
INITAD ;SET UP FOR '200' CONVERSION; A-100,B-100
MOV #100,2ADWCR ;LOAD A/D REG
JSR PC,SETREG ;LOAD A/D INTR. ADDR.
JSR PC,LDINTR ;INTR HERE
TAGYK ;LOAD FINAL CH.
MOV #1777,2ADCR4 ;SET 'GO'
INC 2ADCSRS ;SELECT: SEQ, DMA, INITIAL CH.
MOV #110000,2ADCR4 ;WAIT FOR INTERRUPT
WAIT

;ENTER HERE ON INTERRUPT

005570 052715 004000
005574 022626
005576 004767 004052
005602 105715
005604 001776
005606 022714 110177
005612 001401
005614 104400

TAGYK: BIS #4000,2ADCSRS ;CLR ALL A/D FLAGS
POP2SP ;RESET STACK POINTER
JSR PC,CLRINT ;CLR OUT INTR ADDRESS
TSTB 2ADCSRS ;WAIT FOR 2ND W.C OVRFLO
BEQ -2 ;READ THE 'CR'
CMP #110177,2ADCR4 ;'CR' DOESN'T CONTAIN FINAL CH.
BEQ .+4

:NOW IT HAS BEEN CONFIRMED THAT WE CAN DO D.M.A. TRANSFERS, ATTEMPT A
: 'D.M.A.' TRANSFER TO A NON EXISTANT MEMORY LOCATION TO TEST THAT THE
: A/D WILL RELEASE THE BUS AND SET THE 'MFX' FLAG.
:*****

005616 104002
005620 104011
005622 012777 177766 173462
005630 004767 003122
005634 052715 000060
005640 012777 173000 173452
005646 004767 003746
005652 005666
005654 012714 001000
005660 012714 110000
005664 000001

TST73: SCOPE ;CALL ROUTINE TO INITIALIZE THE 'A/D'
INITAD ;SET UP FOR 8 CONVERSIONS
MOV #10,2ADWCR ;SET UP A/D REG.
JSR PC,SETREG ;SET EA BITS '16 & 17'
BIS #60,2ADCSRS ;SET WORD ADDR 'A' TO AN ILLEGAL ADDR.
MOV #173000,2ADWRA ;LOAD INTERRUPT VECTOR ADDRESS
JSR PC,LDINTR
TAGAL
MOV #1000,2ADCR4 ;LOAD FINAL CH. OF '0'
MOV #110000,2ADCR4 ;LOAD INITIAL CH. & START
WAIT ;WILL IT EVER COME BACK?

;ENTER HERE ON INTERRUPT

005666 022626
005670 004767 003760
005674 005715
005676 100401
005700 104400

TAGAL: POP2SP ;RESET STACK POINTER
JSR PC, CLRINT ;RESTORE INTERRUPT ADDRESS
TST @ADCSRS ;TEST FOR ERROR BIT
BNI .+4 ;BRANCH IF SET
ERROR ;NMX DIDN'T SET 'ERROR' BIT

;TEST THAT A 'D.M.A' RANDOM MODE TRANSFER CAN BE MADE.

005702 104002
005704 104011
005706 012701 030000
005712 012702 014070
005716 010122
005720 005201
005722 022701 030311
005726 001373

005730 012777 177777 173354
005736 004767 003014
005742 012714 030125
005746 104007
005750 104400

TST74: SCOPE
INITAD ;CALL ROUTINE TO INITIALIZE THE 'A/D'
MOV @30000,R1 ;SELECT: RAN DMA CH. '0'
MOV @RANBUF,R2 ;SETUP TO LOAD STATUS WORD BUFFER
TAGYN: MOV R1,(R2)+ ;LOAD CH.S '0-310'
INC R1 ;LOADED ALL CH.S?
CMP @30311,R1 ;BRANCH IF NO
BNE TAGYN

MOV @-1,@ADWCR ;LOAD W.C
JSR PC,SETREG ;LOAD A/D REGS.
MOV @30125,@ADCR4 ;START CONVERSION
CKDONE ;SUBROUTINE TO CHECK FOR 'A/D DONE'
ERROR ;RANDOM DIDN'T SET DONE

:TEST THAT THE CORRECT STATUS WORD IS LOADED FROM THE STATUS BUFFER

005752	104002			TST75:	SCOPE		
005754	104011				INITAD		:CALL ROUTINE TO INITIALIZE THE 'A/D'
005756	012701	030000			MOV #30000,R1		:SELECT: RAN,DMA,CH. '0'
005762	012702	014070			MOV #RANBUF,R2		:SETUP TO LOAD STATUS WORD BUFFER
005766	010122			TAGXN:	MOV R1,(R2)↓		:LOAD CH.S '0-310'
005770	005201				INC R1		
005772	022701	030311			CMP #30311,R1		:LOADED ALL CH.S?
005776	001373				BNE TAGXN		:BRANCH IF NO
006000	012777	177777	173304		MOV #-1,ADWCR		:LOAD W.C
006006	004767	002744			JSR PC,SETREG		:LOAD A/D REGS.
006012	012714	030125			MOV #30125,ADCR4		:START CONVERSION
006016	105715				TSTB ADCSR5		:WAIT FOR DONE (SET VIA W.C.)
006020	100376				BPL .-2		
006022	022714	030000			CMP #30000,ADCR4		:CMP 1ST STATUS WRD TO 'CR'
006026	001401				BEQ .+4		
006030	104400				ERROR		:RANDOM DIDN'T PICK UP CORRECT STATUS WRD

:TEST THAT THE A/D REG'S. ARE RESET ON W.C. OVRFLO USING 'RANDOM' MODE OPERATION
:NOTE: THIS IS THE FIRST TIME MORE THAN 1 CONVERSION IS TAKEN UNDER RANDOM

006032	104002			TST76:	SCOPE		
006034	104011				INITAD		:CALL ROUTINE TO INITIALIZE THE 'A/D'
006036	012701	030000			MOV #30000,R1		:SELECT: RAN,DMA,CH. '0'
006042	012702	014070			MOV #RANBUF,R2		:SETUP TO LOAD STATUS WORD BUFFER
006046	010122			TAGAN:	MOV R1,(R2)↓		:LOAD CH.S '0-310'
006050	005201				INC R1		
006052	022701	030311			CMP #30311,R1		:LOADED ALL CH.S?
006056	001373				BNE TAGAN		:BRANCH IF NO
006060	012777	177500	173224		MOV #-300,ADWCR		:LOAD W.C FOR 300 CONVERSIONS
006066	004767	002664			JSR PC,SETREG		:LOAD A/D REGS.
006072	012714	030125			MOV #30125,ADCR4		:START CONVERSION
006076	105715				TSTB ADCSR5		:WAIT FOR DONE (SET VIA W.C.)
006100	100376				BPL .-2		
006102	022777	014676	173204		CMP #RANBUF+606,ADSMR		:CHECK THAT STATUS WORD WAS UPDATED
006110	001401				BEQ .+4		
006112	104400				ERROR		:STATUS WRD BUFFER SHOULD = RANBUF+606
006114	022777	015072	173176		CMP #ADBUF,ADWRA		:TEST THAT WRD REG. 'A' WAS RESET
006122	001401				BEQ .+4		
006124	104400				ERROR		:WORD REG. 'A' WASN'T RESET VIA OVRFLO
006126	022777	015672	173166		CMP #ADBUF+600,ADWRB		:TEST THAT WRD REG 'B' WAS RESET
006134	001401				BEQ .+4		
006136	104400				ERROR		:WORD REG. 'B' WASN'T RESET VIA OVRFLO
006140	022714	030277			CMP #30277,ADCR4		:SHOULD = LAST CH.
006144	001401				BEQ .+4		
006146	104400				ERROR		:INITIAL CH. REG DOESN'T = LAST CH. CONVERTED

:TEST THAT ONLY '8' LOCATIONS ARE MODIFIED IN RANDOM WITH W.C. 2-10

```

006150 104002
006152 104011
006154 012701 030000
006160 012702 014070
006164 010122
006166 005201
006170 022701 030015
006174 001373
006176 012700 177763
006202 012701 015072
006206 012721 125252
006212 005200
006214 001374
006216 012777 177770 173056
006224 004767 002526
006230 012714 030125
006234 105715
006236 100376
006240 022767 125252 006644
006246 001401
006250 104400
    
```

```

TST77: SCOPE
        INITAD          ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        MOV             #30000,R1      ;SELECT :RAN DMA CH. '0'
        MOV             #RANBUF,R2    ;SETUP TO LOAD STATUS WORD BUFFER
TAGYA:  MOV             R1,(R2)+      ;LOAD CH.S'0-15'
        INC             R1
        CMP             #30015,R1     ;LOADED ALL CH.S?
        BNE             TAGYA
        MOV             #15,R0
        MOV             #ADBUF,R1     ;PRELOAD DATA AREA
TAGYB:  MOV             #125252,(R1)+
        INC             R0
        BNE             TAGYB
        MOV             #10,ADWCR     ;LOAD W.C. FOR '8' CONVERSIONS
        JSR             PC,SETREG     ;LOAD A/D REGS.
        MOV             #30125,ADCR4  ;START CONVERSIONS
        TSTB            ADCSR5       ;WAIT FOR DONE (SET VIA W.C.)
        BPL             #-2
        CMP             #125252,ADBUF+20 ;WERE ONLY '8' LOCATIONS MODIFIED?
        BEQ             #+4
        ERROR           ;A/D TOOK MORE THAN '8' CONVERSIONS W/WC
    
```

:TEST THAT BIT 09 SET IN A STATUS WORD WILL RESET THE SWAR

```

006252 104002
006254 104011
006256 012701 030000
006262 012702 014070
006266 010122
006270 005201
006272 022701 030010
006276 001373
006300 012701 014100
006304 052711 001000
006310 012777 177767 172774
006316 004767 002434
006322 012714 030125
006326 105715
006330 100376
006332 022777 014074 172754
006340 001401
006342 104400
    
```

```

TST100: SCOPE
        INITAD          ;CALL ROUTINE TO INITIALIZE THE AD
        MOV             #30000,R1     ;SEL RAM DMACH0
        MOV             #RANBUF,R2    ;SET UP TO LOAD STATUS AND BUFF
TAGYC:  MOV             R1,(R2)+      ;LOAD CHS 0-10
        INC             R1
        CMP             #30010,R1     ;LOADED 10 CHANNELS
        BNE             TAGYC        ;NOP NOT YET
        MOV             #RANBUF+10,R1
        BIS             #1000,R1      ;SET BIT 09 IN FINAL STATUS WORD
        MOV             #11,ADWCR     ;LOAD WORD COUNT FOR 7 CONV.
        JSR             PC,SETREG     ;LOAD A/D REGS
        MOV             #30125,ADCR4  ;START CONV
        TSTB            ADCSR5       ;WAIT FOR DONE
        BPL             #-2
        CMP             #RANBUF+4,ADSWR ;WAS STATUS WORD ADDRESS REG RESET
        BEQ             #+4
        ERROR           ;SWAR DIDNT RESET WITH BIT 09 SET
    
```

```

*****
: THIS TEST WILL CHECK THE DOUBLE BUFFER FEATURES OF
: THE RANDOM MODE BY PRELOADING THE DATA STORAGE
: AREA WITH A KNOWN VALUE, A WINDOW OF 32 WORDS IS
: PLACED BETWEEN THE BUFFERS, THIS WINDOW IS TESTED
: TO VERIFY THAT NO DATA HAS BEEN STORED IN THIS AREA.
: ON THE FIRST WORD COUNT OVERFLOW THE "GO" BIT IS
: SET AND THE "XFER CHEK" BIT IS CLEAR ENABLING BUFFER
: B TO BE FILLED. THE "XFER CHEK" BIT IS THEN SET PREVENTING
: THE WORD COUNT OVERFLOW FROM BUFFER B TO START FILLING
: BUFFER A AGAIN.
: WHILE BUFFER B IS BEING FILLED THE CONTENTS OF
: BUFFER A IS TESTED FOR BEING MODIFIED.
: BUFFER A IS THEN REFILLED WITH A KNOWN VALUE, THEN THE
: WORD COUNT OVERFLOW FROM BUFFER B IS WAITED FOR. WHEN THE
: WORD COUNT IS DETECTED A STALL LOOP IS RUN. THEN BUFFER B
: IS CHECKED TO VERIFY THAT THE "GO" BIT PREVENTED
: THE OVERFLOW FROM BUFFER A FROM LOADING DATA INTO BUFFER B.
*****

```

006344	104002			TST101: SCOPE	
006346	104011			INITAD	: CALL ROUTINE TO INITIALIZE THE A/D
006350	104012			SAVREG	: SAVE ALL REG. ON STACK
006352	012700	177500		MOV #300,R0	: PRE LOAD COUNTER
006356	012701	015072		MOV #ADBUFF,R1	: GET DATA BUFFER TABLE
006362	012721	152525		15: MOV #152525,(R1)+	: LOAD BUFFER AREA
006366	005200			INC R0	
006370	001374			BNE 15	
006372	012777	177700	172712	MOV #100,ADWCR	: SET W.C. FOR 100 OCTAL
006400	004767	002352		JSR PC,SETREG	: LOAD A/D REGISTERS
006404	062777	000100	172710	ADD #100,ADWRB	: OFFSET REGISTER B FROM A BY 100 OCTAL
006412	012701	014070		MOV #RANBUF,R1	: GET STATUS WORD TABLE POINTER
006416	012700	177700		MOV #100,R0	: PRE LOAD COUNTER FOR 100 STATUS WORDS.
006422	005003			25: CLR R3	: START EACH TIME WITH CH 0
006424	010304			35: MOV R3,R4	: PUT
006426	062704	010000		ADD #10000,R4	: RANDOM.DMA (BASE VALUE OF STATUS)
006432	010421			MOV R4,(R1)+	
006434	005200			INC R0	
006436	001405			BEQ 45	
006440	005203			INC R3	: THE RANDOM CHANNELPARE ONLY 0,1,2,3
006442	022703	000004		CMP #4,R3	: IF NOW A 4 WE'RE TOO HIGH
006446	001366			BNE 35	: RETURN AND LOAD NEXT CH.
006450	000764			BR 25	: RETURN AND START FROM CH0.
006452	052741	001000		45: BIS #1000,-(R1)	: SET FINAL CHANNEL FLAG BIT 9
006456	104013			GETREG	: RESTORE ALL REGS.
006460	004767	003134		JSR PC,LDINTR	: LOAD THE INT. VECTOR
006464	006500			TAGRAN	: INT RETURN TAG
006466	012714	010000		MOV #10000,ADCR4	: START THE CONV. BY LOADING THE CR.
006472	012715	000101		MOV #101,ADCSR5	: SET THE INT. ENABLE AND GO BIT
006476	000001			WAIT	: WAIT FOR INT. FROM A/D
				: RETURN HERE FROM 1ST INTERRUPT AND TEST THAT BUFFER A HAS BEEN MODIFIED	
006500	022626			TAGRAN: POP2SP	: RESET THE STACK
006502	012715	000001		MOV #1,ADCSR5	: INC GO BIT
006506	012701	015072		MOV #ADBUFF,R1	: POINT TO TABLE A

```

006512 012700 177700
006516 022721 152525
006522 001403
006524 005200
006526 001373
006530 000401
006532 104400
1S:    MOV    #-100,R0      ;LOAD COUNTER TO VERIFY TABLE A
        CMP    #152525,(R1)+ ;TEST THAT DATA IN TABLE A HAS BEEN MOD.
        BEQ    2S          ;IF EQUAL REPORT AN ERROR
        INC    R0          ;AT END OF TABLE?
        BNE   1S          ;NOP TRY AGAIN
        BR    3S          ;YES!
2S:    ERROR              ;REPORT TABLE A NOT MODIFIED IN RANDOM
;NOW SET UP INTERRUPT FROM WORD COUNT OVERFLOW OF TABLE B

006534 012777 006562 172564
006542 012777 000340 172560
006550 052715 000100
006554 005077 172510
006560 000001
3S:    MOV    #TAGRAB,2ADINT ;LOAD THE INTERRUPT VECTOR
        MOV    #340,2ADLVL   ;INT. RETURN TAG
        BIS    #100,2ADCSRS  ;ENABLE THE INTERRUPT
        CLR    2PSW
        WAIT
;WAIT FOR INTERRUPT

;ENTER HERE FROM INTERRUPT FROM TABLE B
;TABLE A IS NOW BEING FILLED AGAIN

006562 022626
TAGRAB: POP2SP ;RESET THE STACK
;NOW VERIFY THAT THE WINDOW GAP HAS NOT BEEN WRITTEN OVER

006564 042715 000100
006570 012701 015272
006574 012700 177740
006600 022721 152525
006604 001003
006606 005200
006610 001373
006612 000401
006614 104400
1S:    BIC    #100,2ADCSRS   ;CLR INT. ENABLE
        MOV    #ADBUFF+200,R1 ;LOAD POINTER TO WINDOW AREA
        MOV    #-40,R0      ;PRESET COUNTER
        CMP    #152525,(R1)+ ;CHECK DATA IN WINDOW AREA
        BNE   2S          ;REPORT ERROR DATA IS WRITTEN IN WINDOW AREA
        INC    R0          ;UP COUNTER
        BNE   1S          ;GO BACK IF NOT DONE YET
        BR    3S          ;DATA WAS WRITTEN IN WINDOW AREA
2S:    ERROR
;REFILL TABLE B WITH KNOWN DATA
3S:    MOV    #ADBUFF+300,R1 ;POINT AT TABLE B
        MOV    #-100,R0     ;PRELOAD COUNTER
4S:    MOV    #152525,(R1)+ ;LOAD TABLE B
        INC    R0
        BNE   4S
        MOV    #SYNRET,2ADINT ;LOAD INT VECTOR
        MOV    #340,2ADLVL   ;LOAD THE NEW PROCESSOR PRI.
        BIS    #100,2ADCSRS  ;ENABLE THE INTERRUPT
        CLR    2PSW         ;CLEAR PROCESSOR STATUS
        WAIT              ;NOW WAIT FOR INT. FROM TABLE A

;TEST THAT THE A/D STOPPED BY TESTING THAT NO DATA WAS WRITTEN INTO BUFFER B

006664 022626
006666 042715 000100
006672 012700 177700
006676 005200
006700 001376
006702 012701 015372
006706 022711 152525
006712 001401
006714 104400
SYNRET: POP2SP ;RESTORE STACK
        BIC    #100,2ADCSRS  ;DISABLE THE INTERRUPT
        MOV    #-100,R0
        INC    R0
        BNE   -2
        MOV    #ADBUFF+300,R1 ;LETS LOOK AT FIRST WORD IN TABLE B
        CMP    #152525,(R1)
        BEQ    6S
        ERROR
;A/D FAILED TO STOP WITH GO=0

```

ADF11 PART I, LOGIC DIAGNOSTIC TEST
DZADGA.CMB

MACY11 27(732) 26-OCT-76 16:52 PAGE 48

104

006716 000240

65: NOP

```

:*****
:TEST 'INCREMENT MEMORY' MODE
:THIS ROUTINE EXAMINES THE LOCATION 'INCFLG' (A SOFTWARE FLAG INDICATING
:SUFFICIENT CORE). IF CLEARED, THIS TEST IS SKIPPED
:INCREMENT MEMORY WILL BE TESTED IF THE POP11 HAS AT LEAST
:8K OF CORE. THE (IOR) IS PRESET WITH 034000
:WHICH WILL PLACE THE INCREMENTED LOCATION AT HIGH END OF
:8K. THE (SWAR) IS READ AFTER TAKING ONE INCREMENT MEMORY CONVERT
:TO DETERMINE THE INCREMENTED LOCATION. THE LOCATION IS THEN
:TESTED FOR HAVING BEEN MODIFIED.

```

:*****

```

006720 104002
006722 005767 005072
006726 001431
006730 104011
006732 012701 020000
006736 005021
006740 022701 040000
006744 001374
006746 012777 034000 172350
006754 012777 177777 172330
006762 004767 001770
006766 012714 001000
006772 012714 112000
006776 104007
007000 104400
007002 017701 172306
007006 022711 000001
007012 001401
007014 104400

```

```

TST102: SCOPE
          TST      INCFLG      ;TEST IF CORE IS AVAILABLE
          BEQ      TAGAM      ;EXIT IF NOT
          INITAD
          MOV      #20000,R1    ;CALL ROUTINE TO INITIALIZE THE 'A/D'
          CLRCOR: CLR      (R1)+ ;CLEAR THE LOAD ZONE
          CMP      #40000,R1   ;CLR CORE
          BNE      CLRCOR      ;DONE
          MOV      #34000,2ADADR ;BRANCH IF NOT
          JSR      PC,SETREG   ;LOAD OFFSET IN IOR
          MOV      #1,2ADWCR   ;LOAD W.C FOR 1 CONVERSION
          JSR      PC,SETREG   ;SET UP A/D REG'S.
          MOV      #1000,2ADCR4 ;LOAD FINAL CH.
          MOV      #112000,2ADCR4 ;SELECT: SEQ, DMA, INC.MEM & START
          CKDONE
          ERROR
          MOV      2ADSWR,R1   ;SUBROUTINE TO CHECK FOR 'A/D DONE'
          CMP      #1,(R1)    ;DONE FAILED TO SET
          BEQ      .+4        ;OBTAIN ADDRESS OF DATA FROM STATUS
          TAGAM: BEQ      .+4  ;WAS LOCATION INCREMENTED?
          ERROR              ;BRANCH IF YES
                              ;SELECTED MEMORY ADDR. WASN'T INCREMENTED

```

```

:*****
:TEST INCREMENT MEMORY 'DATA OVRFLO'
:THIS ROUTINE EXAMINES THE LOCATION 'INCFLG' (A SOFTWARE FLAG INDICATING
:SUFFICIENT CORE). IF CLEARED, THIS TEST IS SKIPPED OTHERWISE 'DATA
:OVERFLOW' IS TESTED. THIS IS DONE VIA LOADING ALL POSSIBLE ADDRESSES
:TO BE INCREMENTED WITH '-1' AND TAKING ONE 'DMA' CONVERSION AT A TIME.
:A TEST IS THEN MADE TO SEE IF 'DATA OVRFLO' WAS SET.
:*****
    
```

007016	104002		TST103: SCOPE		
007020	005767	004774	TST	INCFLG	:TEST IF CORE IS AVAILABLE
007024	001435		BEQ	TAGYF	:EXIT IF NOT
007026	104011		INITAD		:CALL ROUTINE TO INITIALIZE THE 'A/D'
007030	012777	020000	MOV	#20000, @ADADR	:LOAD 'A/D OFFSET' REG.
007036	012701	020000	MOV	#20000, R1	:START AT BOTTOM OF BK
007042	012721	177777	LODCOR: MOV	#-1, (R1)+	:LOAD MEM. WITH '-1'
007046	020127	040000	CMP	R1, #40000	:DONE
007052	001373		BNE	LODCOR	:BRANCH IF NOT
007054	012777	177777	MOV	#-1, @ADWCR	:LOAD W.C FOR 1 CONVERSION
007062	004767	001670	JSR	PC, SETREG	:SET UP A/D REG'S.
007066	012714	112000	MOV	#112000, @ADCR4	:SELECT: SEQ DMA, INC. MEM & START
007072	104007		CKDONE		:SUBROUTINE TO CHECK FOR 'A/D DONE'
007074	104400		ERROR		:DONE FAILED TO SET
007076	017701	172212	MOV	@ADSMR, R1	:OBTAIN ADDRESS FROM STATUS REG.
007102	005711		TST	(R1)	:CHECK IF ADDRESS WAS INCREMENTED TO '0'
007104	001401		BEQ	.+4	
007106	104400		ERROR		:SELECTED ADDR WASN'T CLEARED
007110	032715	010000	BIT	#10000, @ADCSR5	:TEST IF OVRFLO SET
007114	001001		BNE	.+4	
007116	104400		ERROR		:DATA OVRFLO FAILED TO SET
007120	000240		TAGYF: NOP		:DATA OVRFLO FAILED TO SET

;TEST THAT DATA CAN BE STORED IN THE HIGHEST 1K OF MEMORY USING SEQ. MODE

007122	104002			TST104: SCOPE		
007124	104011			INITAD		;CALL ROUTINE TO INITIALIZE THE 'A/D'
007126	012777	177300	172156	MOV	#-500,ADWCR	;SET UP FOR '640' CONVERSIONS
007134	016777	004664	172156	MOV	CORMAX,ADWRA	
007142	016777	004656	172152	MOV	CORMAX,ADWRB	
007150	062777	001200	172144	ADD	#1200,ADWRB	;OFFSET WRD REG.'B' VIA '1200' WRDS
007156	016700	004642		MOV	CORMAX,R0	
007162	012701	001000		MOV	#1000,R1	
007166	012720	152525		MOV	#152525,(R0)+	;PRE-LOAD DATA AREA
007172	005301			DEC	R1	
007174	001374			BNE	-6	
007176	004767	002416		JSR	PC,LDINTR	;SET UP INTERRUPT ADDRESS
007202	007220			TAGAO		
007204	012714	111007		MOV	#111007,ADCR4	;SELECT; DMA, FINAL CH. '7'
007210	005215			INC	ADCSRS	;SET 'GO'
007212	012714	130007		MOV	#130007,ADCR4	;SELECT;SEQ. DMA,SINGLE CH.
007216	000001			WAIT		;WAIT FOR INTERRUPT
007220	052715	004000		TAGAO: BIS	#4000,ADCSRS	;CLR ALL FLAGS+'GO BIT'
007224	022626			POP2SP		;RESET STACK POINTER
007226	004767	002366		JSR	PC,LDINTR	;RE-ENABLE INTERRUPT
007232	007236			TAGAAQ		
007234	000001			WAIT		;WAIT FOR 2ND INTERRUPT
007236	052715	004000		TAGAAQ: BIS	#4000,ADCSRS	;CLR ALL FLAGS
007242	022626			POP2SP		;RESET STACK POINTER
007244	004767	002404		JSR	PC,CLRINT	;CLR INTERRUPT
007250	016700	004550		MOV	CORMAX,R0	
007254	012701	001000		MOV	#1000,R1	
007260	022720	152525		TAGAAP: CMP	#152525,(R0)+	;SEE IF MEMORY WAS MODIFIED
007264	001002			BNE	+6	
007266	104400			ERROR		;MEMORY WASN'T MODIFIED ON HIGH MEM. TRANS.
007270	000402			BR	+6	;EXIT ON ERROR
007272	005301			DEC	R1	
007274	001371			BNE	TAGAAP	
007276	000240			TAGAP: NOP		;EXIT ADDRESS IF RUN VIA ACT11

;TEST THAT THE 'A/D' WILL OPERATE IN THE HIGHEST 1K OF MEMORY IN RANDOM MODE

```

007300 104002
007302 104011
007304 012777 177500 172000
007312 016777 004506 171774
007320 016777 004500 171772
007326 062777 001000 171764
007334 012777 000600 171760
007342 067777 171752 171752
007350 017700 171740
007354 012701 000310
007360 012720 030125
007364 005301
007366 001374
007370 017700 171724
007374 012701 000600
007400 012720 152525
007404 005301
007406 001374
007410 004767 002204
007414 007424
007416 012714 030007
007422 000001

007424 052715 004000
007430 022626
007432 004767 002216
007436 017700 171656
007442 012701 000300
007446 022720 152525
007452 001002
007454 104400
007456 000402
007460 005301
007462 001371

007464 016700 004334
007470 062700 001600
007474 012701 000300
007500 022720 152525
007504 001401
007506 104400
007510 005301
007512 001372

TST105: SCOPE
INITAD
MOV #-300, JADWCR ;CALL ROUTINE TO INITIALIZE THE 'A/D'
MOV CORMAX, JADSWR ;SET UP FOR '192' CONVERSIONS
MOV CORMAX, JADWRA ;LOAD STATUS WORD REG.
ADD #1000, JADWRA ;OFFSET DATA BUFFER '500' WRDS FROM STATUS BUFFER
MOV #600, JADWRB ;LOAD REG WITH DOUBLE THE W.C.
ADD JADWRA, JADWRB ;OFFSET 'B' DOUBLE THE W.C. FROM 'A'
MOV JADSWR, R0
MOV #310, R1 ;LOAD '310' STATUS WORDS
MOV #30125, (R0)+
DEC R1
BNE .-6
MOV JADWRA, R0 ;PRE-LOAD DATA AREA.
MOV #600, R1
MOV #152525, (R0)+
DEC R1
BNE .-6
JSR PC, LDINTR ;SET UP INTERRUPT ADDRESS
TAGAQ
MOV #30007, JADCR4 ;SELECT; RANDOM, DMA
WAIT

TAGAQ: BIS #4000, JADCSR5 ;CLR ALL FLAGS
POP2SP ;RESET STACK POINTER
JSR PC, CLRINT ;CLR INTERRUPT
MOV JADWRA, R0
MOV #300, R1
TAGAR: CMP #152525, (R0)+ ;SEE IF MEMORY WAS MODIFIED
BNE .+6
ERROR ;LAST '1K' OF MEM. WASN'T MODIFIED
BR .+6 ;EXIT ON ERROR
DEC R1
BNE TAGAR

;EXAMINE THAT 'WORD REG.B' WASN'T MODIFIED

MOV CORMAX, R0 ;LOAD 'B' STARTING ADDR.
ADD #1600, R0 ;SET UP ADDRESS OF BUFFER 'B'
MOV #300, R1
TAGAS: CMP #152525, (R0)+
BEQ .+4
ERROR ;BUFFER 'B' WAS MODIFIED WITH 'GO' CLEARED.
DEC R1
BNE TAGAS

```

```

;*****
;LAST BUT IN NO WAY LEAST IS THE 'BURST MODE' TEST.
;THIS TEST PRE-LOADS THE DATA BUFFER WITH KNOWN DATA THAN SETS UP TO
;TAKE '500' SEQUENTIAL CONVERSIONS USING THE BURST MODE FEATURE.
;*****
    
```

```

007514 104002
007516 104011
007520 012701 015072
007524 012721 125252
007530 022701 016056
007534 001373
007536 012777 177014 171546
007544 004767 001206
007550 004767 002044
007554 007574
007556 012714 001777
007562 012715 000102
007566 012714 110000
007572 000001
    
```

```

TST106: SCOPE
        INITAD          ;CALL ROUTINE TO INITIALIZE THE 'A/D'
        MOV             #ADBUFF,R1      ;SET UP TO PRE-LOAD DATA BUFFER
        MOV             #125252,(R1)+
        CMP             #ADBUFF+500.,R1 ;DONE?
        BNE             .-10            ;NO
        MOV             #-500.,R1      ;SET UP TO TAKE '500' CONVERSIONS
        JSR             PC,SETREG       ;SET UP THE A/D REG.'S
        JSR             PC,LDINTR       ;LOAD THE A/D INTERRUPT ADDRESS
        TAG1B           ;TO INTERRUPT TO HERE
        MOV             #1777,R1        ;LOAD FINAL CH.
        MOV             #102,R1        ;SET 'BURST' ENABLE BIT & INTR. ENB.
        MOV             #110000,R1     ;SELECT: SEQ.,DMA, INITIAL CH. '0'
        WAIT           ;NOW HOPE FOR THE BEST
    
```

;ENTER HERE ON THE A/D INTERRUPT

```

007574 052715 004000
007600 022626
007602 004767 002046
007606 012701 015072
007612 022721 125252
007616 001002
007620 104400
007622 000403
007624 022701 015572
007630 001370
    
```

```

TAG1B: BIS             #4000,R1        ;CLR ALL FLAGS
        POP2SP          ;RESET STACK
        JSR             PC,CLRINT      ;CLR OUT INTERRUPT ADDRESS
        MOV             #ADBUFF,R1    ;SET UP TO EXAMINE THE BUFFER
TAG1C: CMP             #125252,(R1)+  ;WAS LOCATION MODIFIED?
        BNE             .+6           ;YES
        ERROR          ;DATA BUFFER WASN'T MODIFIED?
        BR              .+10          ;EXIT ON ERROR
        CMP             #ADBUFF+500.,R1 ;EXAMINED ENTIRE BUFFER?
        BNE             TAG1C         ;NO
    
```

```

;*****
;TEST COMPLETE
;*****
    
```

```

007632 104002
007634 104000
007636 013340
007640 000167 172136
    
```

```

TST107: SCOPE
        PRINT          ;TYPE 'LOGIC' TO INDICATE PASS COMPLETION
        MES6
        JMP            RESTAT         ;RESTART TEST
    
```

: DATA UPDATE TEST
:*****

: THE DATA 'UPDATE' TEST IS AN OPERATOR INTERVENTION TEST USED TO TEST
: THE A/D 'DATA BUFFER' & UPDATE LOGIC. THE TEST REQUESTS '4' SPECIFIC
: VOLTAGES (2 POS. & 2 NEG) TO SUPPLIED TO CH. '0'. A CONVERSION IS THEN
: TAKEN AT EACH OF THESE VOLTAGES AND THEN THE '4' BUFFERED DATA WORDS ARE
: READ. THE PROGRAM CHECKS THE SIGN OF THE READ DATA TO DETERMINE IF THE
: DATA WAS BUFFERED CORRECTLY.

007644	004767	172056		DATA:	JSR	PC, SETUP		: INITIALIZE TEST
007650	012767	007662	004126		MOV	#DATA1, AVECTR		: LOAD THE '1A' VECTOR ADDRESS
007656	104000				PRINT			
007660	013361				MES7			: TEXT 'DATA UPDATE TEST'
007662	104011			DATA1:	INITAD			: CALL ROUTINE TO INITIALIZE THE 'A/D'
007664	104000				PRINT			
007666	013311				CRLF			
007670	012700	000002		DATA2:	MOV	#2, R0		
007674	104000				PRINT			
007676	013510				MES8			: TEXT 'SUPPLY +5V TO CH. '0''
007700	004767	000062			JSR	PC, WATINP		: WAIT FOR INPUT & ST. CNVRT
007704	104000				PRINT			
007706	013530				MES9			: TEXT 'SUPPLY -5V TO CH. '0''
007710	004767	000052			JSR	PC, WATINP		: WAIT FOR INPUT & ST. CNVRT
007714	005300				DEC	R0		
007716	001366				BNE	DATA2		
007720	012700	000002			MOV	#2, R0		
007724	005001				CLR	R1		
007726	011367	004114		DATA3:	MOV	#ADDR3, KSTOR4		: READ & STORE DATA
007732	100001				BPL	.+4		: BRANCH IF POS.
007734	104400				ERROR			: 1ST OR 3RD READ DATA WASN'T POS.
007736	005201				INC	R1		
007740	011367	004102			MOV	#ADDR3, KSTOR4		: READ & STORE DATA
007744	100401				BMI	.+4		: BRANCH IF MINUS
007746	104400				ERROR			: 2ND OR 4TH READ DATA WASN'T NEG.
007750	005201				INC	R1		
007752	005300				DEC	R0		
007754	001364				BNE	DATA3		
007756	105777	171314			TSTB	#TPS		
007762	100375				BPL	-.4		
007764	000736				BR	DATA1		
007766	104005			WATINP:	TTYIN			: WAIT FOR 'CR'
007770	005014				CLR	#ADCR4		: START CONVERSION
007772	105715				TSTB	#ADCSRS		: WAIT FOR DONE
007774	100376				BPL	-.2		
007776	052715	004000			BIS	#4000, #ADCSRS		: CLR ALL A/D FLAGS
010002	000207				RTS	PC		


```

010042 104002          SYNC:  SCOPE
010044 104011          INITAD
010046 012777 177777 171236  MOV      #-1, @ADWCR      ; INITIALIZE A/D
010054 004767 000676      JSR      PC, SETREG      ; SET WORD COUNT FOR 1
010060 004767 001534      JSR      PC, LDINTR      ; LOAD REGISTERS OF THE A/D
010064 010114          SYNC1
010066 012714 115000      MOV      @115000, @ADCR4  ; SEQ, DMA, EXTERNAL SYNC FINAL CH. + CH 0.
010072 012714 114000      MOV      @114000, @ADCR4  ; SEQ, DMA, EXTERNAL SYNC + CH 0
010076 005000          CLR      R0              ; LOAD REG 0 FOR WAIT LOOP
010100 005200          15:  INC      R0              ; WAIT LONG DURATION FOR CONVERSION TO BE DONE
010102 001376          BNE     15
010104 042715 000100      BIC     @100, @ADCSRS
010110 104400          ERROR
010112 000401          BR      SYNC1+2      ; EXTERNAL TRIGGER DIDN'T START A/D CONV.

;*****
;TAKE (100 DEC.) CONVERSIONS IN THE SINGLE CHANNEL MODE INITIATED BY
;AN EXTERNAL TRIGGER PULSE.
;*****
010114 022626          SYNC1: POP2SP      ; RESET THE STACK
010116 005715          TST     @ADCSRS          ; WAS INTERRUPT FROM ERROR?
010120 100001          BPL     .+4
010122 104400          ERROR
010124 104002          SCOPE
010126 104011          INITAD
010130 012701 015072      MOV      @ADBUFF, R1      ; INITIALIZE THE A/D
010134 012721 152525      15:  MOV      @152525, (R1)+    ; PRELOAD DATA AREA
010140 022701 015274      CMP     @ADBUFF+202, R1   ; LOAD CONSTANT IN DATA AREA
010144 001373          BNE     15              ; LOADED 100 LOCATIONS YET
010146 004767 001446      JSR      PC, LDINTR      ; IF NOT DO-IT
010152 010220          SYNCRE1
010154 012777 177634 171130  MOV      #-144, @ADWCR    ; LOAD INT VECTOR
010162 004267 000570      JSR      R2, SETREG      ; THIS IS THE RETURNING ADDRESS
010166 012700 177634      MOV      #-144, R0       ; YES 100 DONE NOW SET THE WORD COUNT
010172 012714 115000      MOV      @115000, @ADCR4  ; LOAD A/D REGISTERS
010176 012714 114000      MOV      @114000, @ADCR4  ; # OF CONVERSIONS.
010202 005001          25:  CLR      R1              ; SEQ, DMA, EXTERNAL, FINAL + CH 0
010204 005201          INC      R1              ; SEQ, DMA, EXTERNAL, + CH 0 (START CONV.)
010206 001376          BNE     .-2            ; LOAD VALUE FOR EACH CONVERSION
010210 005200          INC      R0              ; WAIT FOR INDIVIDUAL CONVERSION
010212 001373          BNE     25            ; WAIT FOR INDIVIDUAL CONVERSION
010214 104400          ERROR
010216 000417          BR      SYNC2          ; WAIT 100 TIMES
010220 022626          SYNCRE1: POP2SP      ; IF NOT 100 TIMES GO BACK
010222 005715          TST     @ADCSRS          ; DONE FAILED TO SET (100 CONVERTS) STARTED EXTERNAL.
010224 100001          BPL     .+4            ; GET TO SCOPE ROUTINE AFTER AN ERROR
010226 104400          ERROR
010230 012701 015072      MOV      @ADBUFF, R1      ; RESTORE STACK
010234 012700 177634      MOV      #-144, R0       ; TEST FOR A STATUS ERROR
010240 022721 152525      45:  CMP     @152525, (R1)+    ; INTERRUPT WAS FROM ADF STATUS
010244 001403          BEQ     55            ; VERIFY DATA WAS REALLY READ-IN
010246 005200          INC      R0              ; # OF CONVERTS TO BE VERIFIED
010250 001373          BNE     45            ; VERIFY DATA WAS READ-IN
010252 000401          BR      SYNC2
010254 104400          55:  ERROR

```

```

;*****
;TAKE 100 (OCTAL) CONVERSIONS IN THE RANDOM-DMA MODE ALL SYNCHRONIZED
;INDIVIDUALLY BY EXTERNAL TRIGGER PULSES.
;*****

```

```

010256 104002
010260 104011
010262 012701 015072
010266 012721 152525
010272 022701 015274
010276 001373
010300 012701 014070
010304 012721 014000
010310 022701 014272
010314 001373
010316 004767 001276
010322 010364
010324 012777 177634 170760
010332 004767 000420
010336 012714 014000
010342 012701 177634
010346 005000
010350 005200
010352 001376
010354 005201
010356 001373
010360 104400
010362 000416
010364 022626
010366 005715
010370 100001
010372 104400
010374 012701 015072
010400 022721 152525
010404 001404
010406 022701 015272
010412 001372
010414 000401
010416 104400

```

```

SYNC2: SCOPE
        INITAD
        MOV #ADBUFF,R1 ;INITIALIZE THE ADF
        MOV #152525,(R1)+ ;PRELOAD DATA AREA
1S:     CMP #ADBUFF+202,R1 ;LOAD CONSTANT INTO DATA AREA
        BNE J% ;IS AREA LOADED YET?
        MOV #RANBUF,R1 ;YES NOW LOAD STATUS WORDS
2S:     MOV #014000,(R1)+ ;RANDOM, DMA, EXT, CH=0
        CMP #RANBUF+202,R1 ;100 WORDS YET?
        BNE JSR
        JSR PC,LDINTR
        SYNRE2
        MOV #8-144,ADWCR ;SET WORD COUNT FOR 100 CONVERSIONS
        JSR PC,SETREG ;SET UP A/D REGISTERS
        MOV #014000,ADCR4 ;LOAD CONTROL REGISTER RAN, DMA, EXT. TRIG.
        MOV #8-144,R1 ;NUMBER OF CONVERSIONS WAITED FOR
3S:     CLR R0 ;DURATION OF EACH CONVERSION
4S:     INC R0 ;DO A SINGLE DURATION HERE
        BNE 4S ;LOOP TILL DONE
        INC R1 ;COUNT DURATIONS
        BNE 3S ;LOOP TILL 100 DURATIONS DONE
        ERROR ;DONE FAILED TO SET ON 100 RANDOM, (EXT.TRIG) CONVERTS
        BR SYNC3 ;WITH ERROR
SYNRE2: POP2SP ;RESTORE STACK
        TST ADCSR5 ;TEST FOR ASTATUS ERROR
        BPL .+4
        ERROR ;INTERRUPT WAS FROM STATUS ERROR
        MOV #ADBUFF,R1 ;VERIFY THAT DATA WAS MODIFIED
6S:     CMP #152525,(R1)+
        BEQ 7S
        CMP #ADBUFF+200,R1 ;HAVE WE DONE 100 VERIFICATIONS YET?
        BNE 6S ;IF NOT LOOP AGAIN
        BR SYNC3 ;YES, NO ERROR
7S:     ERROR ;FAILED TO MODIFY BUFFER AREA (DONE SET)

```

```

;*****
;TAKE 100 (OCTAL) CONVERSIONS IN THE RANDOM-DMA MODE. THE CONVERTS ARE
;SYNCHRONIZED ALTERNATELY "INTERNAL" AND "EXTERNAL"
;*****

```

```

010420 104002
010422 104011
010424 012701 015072
010430 012721 152525
010434 022701 015274
010440 001373
010442 004767 001152
010446 010542
010450 012701 014070
010454 012721 014000
010460 012721 010000

```

```

SYNC3: SCOPE
        INITAD
        MOV #ADBUFF,R1 ;INITIALIZE THE ADF
1S:     MOV #152525,(R1)+ ;GET POINTER TO DATA AREA
        CMP #ADBUFF+202,R1 ;LOAD DATA AREA
        BNE 1S ;DONE YET
        JSR PC,LDINTR ;LOOP TILL DONE
        SYNRE3
        MOV #RANBUF,R1 ;GET STATUS WORD TABLE POINTER
2S:     MOV #014000,(R1)+ ;RAN, DMA, EXT, TRIG AND CH=0
        MOV #010000,(R1)+ ;RAN, DMA, CH=0

```

```

010464 022701 014404          CMP      #RANBUF+314,R1 ;LOAD 100 STATUS WORDS
010470 001371          BNE      2$
010472 052711 001000          BIS      #1000,(R1) ;SET THE FINAL WORD STATUS BIT
010476 012777 177634 170606    MOV      #-144,ADWCR ;SET WORD COUNT FOR 100
010504 004767 000246          JSR      PC,SETREG ;SET UP A/D REG.
010510 012714 014000          MOV      #014000,ADCR4 ;LOAD CONTROL REG
010514 012701 177634          MOV      #-144,R1 ;# OF CONVERTS
010520 005000          3$: CLR      R0 ;DURATION OF EACH CONVERT
010522 005200          4$: INC      R0 ;LOOP TIME FOR EACH CONVERT HERE
010524 001376          BNE      4$
010526 005201          INC      R1 ;COUNT # OF CONVERTS HERE
010530 001373          BNE      3$
010532 052715 020000          BIS      #20000,ADCSRS ;REINIT THE A/D
010536 104400          ERROR ;DONE BIT NOT SET
010540 000401          BR      SYNRE3+2
;TEST THAT THE DATA BUFFER HAS BEEN MODIFIED

010542 022626          SYNRE3: POP2SP ;POP 2 FROM STACK
010544 005715          TST      ADCSRS ;TEST FOR A STATUS ERROR
010546 100001          BPL      .+4
010550 104400          ERROR ;INTERRUPT WAS FROM ADF STATUS ERROR
010552 012701 015072          MOV      #ADBLFF,R1 ;GET DATA BUFFER POINTER
010556 022721 152525          6$: CMP      #152525,(R1)+ ;VERIFY DATA WAS MODIFIED
010562 001404          BEQ      7$ ;IF SAME DATA WASN'T MODIFIED
010564 022701 015272          CMP      #ADBLFF+200,R1 ;DONE YET
010570 001372          BNE      6$
010572 000401          BR      8$
010574 104400          7$: ERROR ;DATA BUFFER WASNT MODIFIED (DONE BIT SET)
010576 104002          8$: SCOPE
010600 104000          PRINT
010602 013710          MES42
010604 000167 177224          JMP      CYC ;RECYCLE TEST

```

; ROUTINE TO LOOP THRU A SINGLE LOGIC SUBTEST. ENTERED FROM THE 'MONITOR'
; VIA TYPING 'TNN' WHERE 'NN' IS EQUATED TO THE 'PC' OF A SUBTEST.
; NOTE THAT 'SW11' MUST BE '0' (DOWN) TO RUN THIS TEST.

010610	004767	171112		TESTX:	JSR	PC, SETUP	; MAIN INITIALIZATION FOR LOGIC TEST
010614	005067	000444			CLR	INBUF	
010620	012701	011264			MOV	#INBUF, R1	
010624	005067	003210			CLR	KSTOR1	
010630	042711	177770		TSTA:	BIC	#177770, (R1)	; STRIPE NO. TO OCTAL
010634	062167	003200			ADD	(R1)+, KSTOR1	; ADD TO LAST RESULT
010640	005367	003170			DEC	CHRCNT	
010644	001407				BEQ	TSTB	
010646	006367	003166			ASL	KSTOR1	
010652	006367	003162			ASL	KSTOR1	
010656	006367	003156			ASL	KSTOR1	
010662	000762				BR	TSTA	
010664	022767	010610	003146	TSTB:	CMPL	#TESTX, KSTOR1	; IS NO. WITHIN LIMITS OF THE LOGIC TEST
010672	100002				BPL	+6	; CONTINUE IF YES
010674	000167	171020			JMP	INIT3	; OTHERWISE RETURN TO MONITOR
010700	062767	000002	003132		ADD	#2, KSTOR1	; ADD '2' TO POINT TO INSTRUCTION AFTER SCOPE
010706	005067	001774		XLOOP:	CLR	SCOPEF	; KEEP COUNT AT ZERO
010712	012767	010706	001770		MOV	#XLOOP, RETURN	; LOAD SCOPE LOOP RETURN POINTER
010720	000177	003114			JMP	#KSTOR1	; JUMP TO TEST

; SUBROUTINE TO ISSUE N SPACES
; N IS ONE PLUS VALUE CONTAINED IN SPACEX
; SPACEX IS CLEARED WITHIN THE SUBROUTINE, SO THAT A CALL ON
; SPACE WITHOUT LOADING SPACEX ISSUES ONLY ONE SPACE

010724	105777	170346		XSPACE:	TSTB	@TPS	; WAIT FOR TTY READY
010730	100375				BPL	-4	
010732	012777	000240	170340		MOV	#240, @TPB	; OUTPUT A SPACE
010740	005367	000010			DEC	SPACEX	; DECREMENT COUNT
010744	003367				BGT	XSPACE	; LOOP IF NOT DONE
010746	005067	000002			CLR	SPACEX	; RESET COUNT TO ZERO
010752	000002				RTI		; RETURN
010754	000000			SPACEX:	0		

; SUBROUTINE TO SET UP ALL A/D REGISTERS

010756	012777	014070	170330	SETREG:	MOV	#RANBUF, @ADSWR	
010764	012777	015072	170326		MOV	#ADBUFF, @ADMRA	
010772	017777	170314	170322		MOV	@ADMCR, @ADMWB	
011000	005477	170316			NEG	@ADMWB	
011004	006377	170312			ASL	@ADMWB	
011010	062777	015072	170304		ADD	#ADBUFF, @ADMWB	
011016	000207				RTS	PC	

;KEYBOARD SERVICE ROUTINE

011020	104012		XTTYIN: SAVREG		
011022	012704	011264	MOV	#INBUF,R4	; SETUP CHARACTER BUFFER
011026	005067	003002	CLR	CHRCNT	; CLEAR CHARACTER COUNTER
011032	005067	000230	CLR	INBUF+2	
011036	105777	170230	INPUTA: TSTB	@TKS	; CHARACTER READY?
011042	100375		BPL	INPUTA	; NO, WAIT IT OUT
011044	017701	170224	MOV	@TKB,R1	; SAVE CHARACTER
011050	042701	000200	BIC	#200,R1	; STRIPE PARITY BIT
011054	120127	000060	CMPS	R1,#60	; IS IT A SPECIAL CHARACTER
011060	100420		BMI	SPCHR1	; YES, TEST IT
011062	122701	000137	CMPS	#137,R1	
011066	100415		BMI	SPCHR1	
011070	010124		INPUTB: MOV	R1,(R4)+	; SAVE CHARACTER
011072	005267	002736	INC	CHRCNT	; INCREMENT THE CHARACTER COUNT.
011076	022767	000007	002730	CMPS	#7,CHRCNT
011104	100461		BMI	SPCHR5	; TYPE '?' IF TOO MANY CHAR.
011106	105777	170164	OUTPTA: TSTB	@TPS	; ECHO CHARACTER
011112	100375		BPL	OUTPTA	
011114	110177	170160	MOV	R1,@TPB	
011120	000746		BR	INPUTA	; WAIT FOR NEXT CHARACTER
			;SUBROUTINE TO TEST FOR SPECIAL CHARACTERS : 'IC', '+', 'CR', ',', ' OR 'RUBOUT'		
011122	122701	000003	SPCHR1: CMPS	#3,R1	; CHAR. = 'IC'
011126	001002		BNE	#+6	; NO, NOT 'IC'
011130	000167	170446	JMP	MONITR	; YES, EXIT TO MONITOR
011134	122701	000001	CMPS	#1,R1	; CHAR. = 'IA'?
011140	001006		BNE	#+16	; NOT 'IA'
011142	022626		POP2SP		; YES, RESTORE STACK
011144	104000		PRINT		
011146	013306		CNTRLA		; TEXT 'IA'
011150	104013		GETREG		; RESTORE THE REG.'S
011152	000177	002626	JMP	@AVECTR	
011156	122701	000177	CMPS	#177,R1	; CHAR. = 'RUBOUT'
011162	001011		BNE	SPCHR2	; IGNORE CHAR. & EXIT
011164	005767	002644	TST	CHRCNT	; IS RUBOUT LEGAL?
011170	001722		BEQ	INPUTA	; NO, IGNORE IT
011172	005367	002636	DEC	CHRCNT	
011176	012701	000134	MOV	#134,R1	; TYPE '\ ' TO INDICATE RUBOUT
011202	005744		TST	-(R4)	; POP OFF LAST CHARACTER
011204	000740		BR	OUTPTA	; WAIT FOR NEXT CHARACTER
011206	122701	000053	SPCHR2: CMPS	#53,R1	; TEST FOR '+'
011212	001004		BNE	SPCHR3	; BRANCH IF NO
011214	012767	000177	002550	MOV	#177,ADSIGN
011222	000731		BR	OUTPTA	; YES, INDICATES UNIPOLAR
011224	122701	000054	SPCHR3: CMPS	#54,R1	; WAIT NEXT CHAR.
011230	001717		BEQ	INPUTB	; TEST FOR ','
011232	122701	000015	SPCHR4: CMPS	#15,R1	; LEGAL CHAR., SAVE IT
011236	001004		BNE	SPCHR5	; =TO 'CARRIAGE RETURN' TO TERMINATE?
011240	104000		PRINT		; NO, CONTINUE
011242	013311		CRLF		; YES, TYPE 'CR-LF'
011244	104013		EXTTY: GETREG		; RESTORE THE REG.'S
011246	000002		RTI		; EXIT

011250 122701 000040
011254 001714
011256 104000
011260 013316
011262 000657
011264 000000
011304 011304

SPCHRS: CMPB #40 R1 ; TEST FOR SPACE
BEQ OUTPTA ; ECHO BUT DON'T SAVE
PRINT ; OTHERWISE TYPE '?'
QMARK
BR XTTYIN+2 ; WAIT FOR NEW ENTRY
INBUF: 0 ; CHARACTER STORAGE BUFFER
 ; = +16
; SUBROUTINE WILL CONVERT 'N' BCD WORDS (SEPARATED VIA COMMA'S)
; WHICH WERE STORED IN A TABLE VIA 'TTYIN' TO OCTAL AND STORE THEM.

011304 104012
011306 012704 011264
011312 012703 011460
011316 005067 000140
011322 005001
011324 005002
011326 005767 002502
011332 003426
011334 005367 002474
011340 122714 000054
011344 001421
011346 121427 000060
011352 002435
011354 021427 000071
011360 003032
011362 042714 177760
011366 012400
011370 010102
011372 006301
011374 006301
011376 006301
011400 060201
011402 060201
011404 060001
011406 000747
011410 005724
011412 010123
011414 005767 002414
011420 001340
011422 026727 000032 000777
011430 100006
011432 026727 000024 000777
011440 100002
011442 104013
011444 000002
011446 104000
011450 013316
011452 104005
011454 000167 177624
011460 000000
011462 000000
011464 000000
011466 000000

BCDBIN: SAVREG ; SAVE THE REG.'S
MOV #INBUF,R4 ; SETUP ASCII STORAGE TABLE
MOV #BCDTAB,R3 ; TABLE FOR STORAGE OF CONVERTED WORDS
CLR BCDTAB+2
BCDBN1: CLR R1 ; REG. TO STORE RUNNING TOTAL
CLR R2 ; TEMP. STORAGE FOR 'R1'
BCDBN2: TST CHRCNT ; END OF DATA?
BLE BCDEND ; YES, EXIT
DEC CHRCNT ; DECREMENT CHARACTER COUNTER
CMPB #54,(R4) ; IS CHARACTER = TO ' '?
BEQ BCDEND ; YES, DECODE NEW WORD
CMPB (R4),#60 ; TEST FOR LEGAL NO.
BLT BCDERR
CMP (R4),#71
BGT BCDERR
BIC #177760,(R4) ; STRIPE NO. TO BCD
MOV (R4)+,R0 ; SAVE NO. IN R0.
MOV R1,R2 ; SAVE CURRENT TOTAL
ASL R1 ; NX2
ASL R1 ; NX4
ASL R1 ; NX8
ADD R2,R1 ; NX9
ADD R2,R1 ; NX10
ADD R0,R1 ; N+NEW NO.
BR BCDEN2
BCDEND: TST (R4)+ ; UPDATE BUFFER
MOV R1,(R3)+ ; SAVE CONVERTED VALUE & SETUP TO SAVE NEXT
TST CHRCNT ; FINISHED?
BNE BCDEN1 ; NO, CONVERT NEXT WORD
CMP BCDTAB,#777 ; TEST IF NO. <511
BPL BCDERR ; REPORT ERROR IF NOT
CMP BCDTAB+2,#777 ; TEST IF 2ND. NO. <511
BPL BCDERR ; BRANCH IF NOT
GETREG ; RESTORE THE REG.'S
RTI ; YES, EXIT
BCDERR: PRINT ; TYPE '??'
QMARK ; TO BE TYPED ON QUESTIONABLE ENTRIES.
TTYIN
JMP BCDBIN ; OCTAL STORAGE TABLE
BCDTAB: 0
0
0
0

```

011470 010046
011472 010146
011474 010246
011476 010346
011500 010446
011502 010546
011504 016746 166314
011510 010667 002316
011514 012767 011524 166302
011522 000000

:POWER FAIL HANDLER
PWFAL: MOV R0,-(SP)
      MOV R1,-(SP)
      MOV R2,-(SP)
      MOV R3,-(SP)
      MOV R4,-(SP)
      MOV R5,-(SP)
      MOV 24,-(SP)
      MOV SP,PROC
      MOV #PWRUP,24
      HALT

:POWER UP HANDLER
PWRUP: MOV #340,2PSW
      MOV PROC,SP
      MOV (SP)+,24
      MOV (SP)+,R5
      MOV (SP)+,R4
      MOV (SP)+,R3
      MOV (SP)+,R2
      MOV (SP)+,R1
      MOV (SP)+,R0
      CLR R1
      INC R1
      BNE .-2
      PRINT
      MES21
      JMP MONITR

;POWER UP DELAY

```

```

:*****
:THIS SUBROUTINE IS CALLED VIA A 'IOT' TRAP. IT IS USED BY MANY LOGIC
:SUBTESTS TO CHECK FOR 'A/D DONE'. THE ROUTINE CHECKS FOR 'A/D DONE'
:WHILE INCREMENTING A WAIT LOOP. IF 'DONE' SETS PROGRAM CONTROL IS THEN
:TRANSFERRED BACK TO THE 'IOT' CALL +4. IF THE LOOP TIMES OUT (NO DONE
:SETS), PROGRAM CONTROL IS TRANSFERRED BACK TO THE 'IOT' CALL+2
:*****

```

```

011574 016701 002176
011600 105715
011602 100403
011604 005201
011606 001374
011610 000002
011612 062716 000002
011616 000002

XCKDON: MOV DELAY1,R1 ;INITIALIZE WAIT LOOP
XCHK1: TSTB 2ADCSRS ;TEST FOR DONE
      BMI EXTCHK ;BRANCH IF SET
      INC R1 ;INCREMENT WAIT LOOP
      BNE XCHK1
      RTI ;RETURN TO CALL+2

EXTCHK: ADD #2,(SP) ;RETURN TO CALL +4
      RTI

```

;SUBROUTINE TO SET UP THE A/D VECTOR ADDR TO ENABLE INTERRUPTS.

```

011620 017677 000000 167500 LDINTR: MOV    @ (SP), @ADINT    ;LOAD INTERRUPT SERVICE ADDRESS
011626 062716 000002          ADD    #2, (SP)          ;SET UP STACK TO EXIT
011632 012777 000340 167470          MOV    #340, @ADLVL      ;SET A/D LEVEL @ 7
011640 052777 000100 167442          BIS    #100, @ADCSR     ;SET INTERRUPT ENABLE
011646 005077 167416          CLR    @PSW            ;SET PROC. @ 0
011652 000207          RTS    PC
    
```

;SUBROUTINE TO RESET A/D VECTOR ADDRESS TO HALT ON INTERRUPTS

```

011654 012777 000340 167406 CLRINT: MOV    #340, @PSW      ;RE-SET PROC. PRIORITY @ 7
011662 042777 000100 167420          BIC    #100, @ADCSR     ;CLR INTR ENABLE
011670 016777 167434 167430          MOV    ADLVL, @ADINT
011676 005077 167426          CLR    @ADLVL
011702 000207          RTS    PC
    
```

;SUBROUTINE ENTERED ON AN ILLEGAL TRAP. THE ROUTINE REPORTS WHERE IT
;TRAPPED 'FROM' AND WHERE IT TRAPPED 'TO'.

```

011704 011667 002140          ERTRAP: MOV    (SP), TEMP1    ;SAVE LOCATION WHERE IT TRAPPED 'TO'
011710 022626          POP2SP
011712 011667 002134          MOV    (SP), TEMP2    ;SAVE WHERE IT TRAPPED FROM.
011716 104000          PRINT
011720 013657          MES40              ;TEXT 'ILLEGAL TRAP TO'
011722 162767 000004 002120          SUB    #4, TEMP1
011730 104004          PRTOCT
011732 014050          TEMP1              ;TYPE 'PC' TRAPPED TO
011734 104000          PRINT
011736 013701          MES41              ;TEXT 'FROM'
011740 162767 000002 002104          SUB    #2, TEMP2
011746 104004          PRTOCT
011750 014052          TEMP2              ;TYPE WHERE IT TRAPPED FROM
011752 000167 167624          JMP    MONITR        ;RETURN TO MONITOR
    
```

;ROUTINE TO INITIALIZE THE A/D

```

011756 052777 020000 167324 XINIT: BIS    #20000, @ADCSR
011764 000002          RTI
    
```

: SUBROUTINE TO SAVE 'R1-R5' ON STACK
:*****

011766	012667	002016	XSAVRG: MOV	(SP)+,SAVEPC
011772	012667	002014	MOV	(SP)+,SAVPSW
011776	012667	002012	MOV	(SP)+,SAV2PC
012002	012667	002010	MOV	(SP)+,SAV2SW
012006	010146		MOV	R1,-(SP)
012010	010246		MOV	R2,-(SP)
012012	010346		MOV	R3,-(SP)
012014	010446		MOV	R4,-(SP)
012016	010546		MOV	R5,-(SP)
012020	016746	001772	MOV	SAV2SW,-(SP)
012024	016746	001764	MOV	SAV2PC,-(SP)
012030	016746	001756	MOV	SAVPSW,-(SP)
012034	016746	001750	MOV	SAVEPC,-(SP)
012040	000002		RTI	

: SUBROUTINE TO RESTORE 'R1-R5' FROM THE STACK
:*****

012042	012667	001742	XGETRG: MOV	(SP)+,SAVEPC
012046	012667	001740	MOV	(SP)+,SAVPSW
012052	012667	001736	MOV	(SP)+,SAV2PC
012056	012667	001734	MOV	(SP)+,SAV2SW
012062	012605		MOV	(SP)+,R5
012064	012604		MOV	(SP)+,R4
012066	012603		MOV	(SP)+,R3
012070	012602		MOV	(SP)+,R2
012072	012601		MOV	(SP)+,R1
012074	016746	001716	MOV	SAV2SW,-(SP)
012100	016746	001710	MOV	SAV2PC,-(SP)
012104	016746	001702	MOV	SAVPSW,-(SP)
012110	016746	001674	MOV	SAVEPC,-(SP)
012114	000002		RTI	

;MESSAGE PRINT ROUTINE, ENTERED VIA EMT DISPATCH HANDLER.
;ROUTINE PICKS UP CONTENTS OF THE 'PC' AND USES THIS AS
;THE ADDRESS OF MESSAGE TO BE TYPED.

012116	104012			TYPMES: SAVREG		;SAVE THE REGISTERS
012120	017602	000000		MOV	2(SP),R2	;GET THE MESSAGE ADDRESS FROM START
012124	062716	000002		ADD	2,(SP)	;SET UP STACK TO EXIT
012130	012777	000100	167134	MOV	100,2TKS	
012136	005077	167126		CLR	2PSW	;ENABLE KEYBOARD INTR.
012142	105777	167130		TYPERA: TSTB	2TPS	
012146	100375			BPL	TYPERA	;WAIT FOR TTY DONE
012150	122712	000100		CMPB	100,(R2)	;TEST FOR '2'
012154	001002			BNE	TYPER1	;BRANCH IF NO EQUAL
012156	104013			GETREG		
012160	000002			RTI		;OTHERWISE EXIT
012162	122712	000045		TYPER1: CMPB	45,(R2)	;TEST FOR '%'
012166	001403			BEG	TYPECL	;IF = TYPE 'CR-LF'
012170	112277	167104		TYPER2: MOVB	(R2)+,2TPB	;OUTPUT CHAR.
012174	000762			BR	TYPERA	
012176	012777	000015	167074	TYPECL: MOV	15,2TPB	;TYPE 'CR'
012204	105777	167066		TSTB	2TPS	
012210	100375			BPL	-4	
012212	012777	000012	167060	MOV	12,2TPB	
012220	105722			TSTB	(R2)+	;INCREMENT BUFFER
012222	000747			BR	TYPERA	

;SUBROUTINE TO TYPEOUT A '6' DIGIT OCTAL NO. THE 'PC' CONTAINS
;THE ADDRESS OF 'WORD' TO BE TYPED

012224	104012			OCTPRT: SAVREG		
012226	012777	000100	167036	MOV	100,2TKS	
012234	005077	167030		CLR	2PSW	;ENABLE KEYBOARD INTR.
012240	017600	000000		MOV	2(SP),R0	;THE ADDRESS OF WORD TO BE TYPED
012244	062716	000002		ADD	2,(SP)	;SET UP STACK TO EXIT
012250	012767	000006	001564	MOV	6,KSTOR2	
012256	012767	000376	000056	MOV	376,MASK	;MASK FOR FIRST BIT
012264	000401			BR	+4	
012266	006110			SHIFT: ROL	(R0)	
012270	006110			ROL	(R0)	
012272	006110			ROL	(R0)	
012274	111002			MOVB	(R0),R2	
012276	146702	000040		BICB	MASK,R2	
012302	052702	000260		BIS	260,R2	
012306	132777	000200	166762	BITB	200,2TPS	
012314	100374			BPL	-6	;WAIT FOR PRINTER READY
012316	110277	166756		MOVB	R2,2TPB	;PRINT CHAR.
012322	012767	000370	000012	MOV	370,MASK	;MASK FOR NEXT '5' DIGITS
012330	005367	001506		DEC	KSTOR2	
012334	001354			BNE	SHIFT	
012336	104013			GETREG		
012340	000002			RTI		
012342	000376			MASK:	376	

; ENTERED WITH SYSTEM TRAP CALL (ERROR)
; PRINT OUT THE ERROR ADDRESS AND ALL A/D STATUS REGISTERS

012344	104006			LOGERR: TSTTKS					
012346	037727	166730	020000	BIT	%SWR, #20000				; TEST FOR KEYBOARD INTERRUPT
012354	001110			BNE	CK				; TEST SW-13 FOR INHIBIT PRINT OUT
012356	011667	001462		MOV	(SP), KSTOR3				; INHIBIT, CHECK FOR HALT
012362	162767	000002	001454	SUB	#2, KSTOR3				; PC OF FAILING ROUTINE
012370	042777	000100	166712	BIC	#100, %ADCSR				; CLR INTERRUPT ENABLE
012376	010046			MOV	RD, -(SP)				; SAVE REGISTERS 0, 1 & 2
012400	010146			MOV	R1, -(SP)				
012402	010246			MOV	R2, -(SP)				
012404	022767	007662	001372	CMP	%DATA1, AVECTR				; RUNNING DATA TEST ?
012412	001034			BNE	ERR1				; NO, LOGIC ERROR
012414	005701			TST	R1				
012416	001003			BNE	.+10				
012420	012767	013547	000046	MOV	%MES10, PRNTIT				
012426	022701	000001		CMP	#1, R1				
012432	001003			BNE	.+10				
012434	012767	013561	000032	MOV	%MES11, PRNTIT				
012442	022701	000002		CMP	#2, R1				
012446	001003			BNE	.+10				
012450	012767	013573	000016	MOV	%MES12, PRNTIT				
012456	022701	000003		CMP	#3, R1				
012462	001003			BNE	.+10				
012464	012767	013605	000002	MOV	%MES13, PRNTIT				
012472	104000			PRINT					
012474	000000			PRNTIT: 0					
012476	104004			PRTOCT					
012500	014046			KSTOR4					; PRINT READ DATA
012502	000432			BR	XPRT1				; EXIT
012504	005767	001346		TST	MESPRT				; HAS HEADER BEEN TYPED
012510	001004			BNE	ERR2				; BRANCH IF YES
012512	104000			PRINT					
012514	013101			MES3					; PRINT LOGIC ERROR HEADER
012516	005267	001334		INC	MESPRT				; SET PRINT INHIBIT SW.
012522	104000			PRINT					; OUTPUT CARRIAGE RETURN AND LINE FEED
012524	013311			CRLF					
012526	104004			PRTOCT					; PRINT FAILING PC+2
012530	014044			KSTOR3					
012532	104003			SPACE					; OUTPUT A SPACE
012534	012767	000007	001306	MOV	#7, TEMP1				
012542	012701	001306		MOV	%ADCR, R1				
012546	012100			MOV	(R1)+, RD				
012550	011067	001270		MOV	(RD), KSTOR3				
012554	104004			PRTOCT					
012556	014044			KSTOR3					
012560	104003			SPACE					
012562	005367	001262		DEC	TEMP1				
012566	001367			BNE	XPRT				

```

012570 012602      XPRT1:  MOV      (SP)+,R2      ;RESTORE REGISTERS
012572 012601      MOV      (SP)+,R1
012574 012600      MOV      (SP)+,R0
012576 005777 166500  CK:      TST      @SWR      ;CHECK SW-15 FOR HALT SWITCH
012602 100001      BPL      .+4      ;BRANCH IF NOT SET
012604 000000      HALT     ;HALT ON ERROR UP
012606 032777 010000 166466  BIT      @SW12,@SWR ;TEST SW 12
012614 001401      BEQ     .+4      ;BRANCH IF NOT SET
012616 104011      INIAD   ;CALL ROUTINE TO INITIALIZE THE 'A/D'
012620 000002      RTI     ;RETURN TO MAIN LINE

```

;SCOPE AND/OR ITERATION LOOP FOR EACH TEST 200 TIMES

```

012622 104006      SCOPEC:  TSTTKS   ;TEST FOR KEYBOARD INIT
012624 032777 040000 166450  BIT      @40000,@SWR ;TEST SW-14 FOR SCOPE
012632 001012      BNE     SCOPEB   ;YES SCOPE
012634 032777 004000 166440  BIT      @4000,@SWR ;NO-TEST SW-11 FOR ITERATION
012642 001013      BNE     SCOPEG   ;INHIBIT ITERATION
012644 026767 000036 000032  CMP     SCOPEF,ICOUNT ;COMPARE CURRENT COUNT TO MAX NUMBER
012652 100007      BPL     SCOPEG   ;EXIT-DONE
012654 005267 000026      INC     SCOPEF   ;INCREMENT COUNT
012660 022606      SCOPEB:  CMP     (6)+,SP ;REPOSITION STACK
012662 012677 166402      MOV     (6)+,@PSW ;RESTORE PREVIOUS PROCESSOR STATUS
012666 000177 000016      JMP     @RETURN  ;REPEAT TEST
012672 005067 000010      SCOPEG:  CLR     SCOPEF ;CLEAR COUNT
012676 011667 000006      MOV     @SP,@RETURN ;SAVE SCOPE RETURN POINTER
012702 000002      RTI     ;RETURN INLINE-NEXT TEST
012704 000200      ICOUNT: 200      ;ITERATION COUNT
012706 000000      SCOPEF: 0        ;COUNT LOCATION FOR ITERATION LOOP
012710 002014      RETURN:  TST1

```

;SUBROUTINE TO TEST FOR THE KEYBOARD FLAG BEING SET

```

012712 105777 166354  TKSFLG:  TSTB   @TKS      ;FLAG SET?
012716 100001      BPL     .+4      ;NO EXIT
012720 104005      TTYIN  ;YES, INQUIRE
012722 000002      RTI

```

```

;*****
;ROUTINE TO TRANSMIT A 'NULL' CHAR. TO THE PRINTER
;*****

```

```

012724 012767 000002 176022 XNULL:  MOV      @2,SPACEX
012732 105777 166340      TSTB   @TPS
012736 100375      BPL     .-4
012740 005077 166334      CLR    @TPB      ;TRANSMIT NULL CHAR.
012744 005367 176004      DEC    SPACEX
012750 001370      BNE    XNULL+6
012752 000002      RTI

```

;MESSAGES

012754	000		
012755	045	040445	043104
012762	030461	050040	051101
012770	020124	026111	046040
012776	043517	041511	042040
013004	040511	047107	051517
013012	044524	020103	042524
013020	052123	020054	033461
013026	045055	046125	033455
013034	065		
013035	040	046450	044501
013042	042116	041505	030455
013050	026461	055104	042101
013056	026507	024501	100

TITLE: .BYTE
.ASCII '%ADF11 PART I, LOGIC DIAGNOSTIC TEST, 17-JUL-75'

.ASCII '(MAINDEC-11-DZADG-A)@'

013063	045	027501	020104
013070	042514	043516	044124
013076	020077	100	
013101	045	020040	041520
013106	020040	020040	041440
013114	020122	020040	041440
013122	051123	020040	020040
013130	053440	020103	020040
013136	052123	052101	051525
013144	020040	040504	040524
013152	020040		
013154	042101	051104	040440
013162	040440	042104	020122
013170	040102		

MES2: .ASCII '%A/D LENGTH? @'

MES3: .ASCII '% PC CR CSR WC STATUS DATA '

.ASCII 'ADDR A ADDR B@'

013172	052045	050131	020105
013200	042514	052124	051105
013206	024040	024440	052040
013214	020117	052522	020116
013222	042504	044523	042522
013230	020104	042524	052123
013236	020072		
013240	046050	047451	044507
013246	026103	024040	024504
013254	040504	040524	020054
013262	051450	051451	047131
013270	020103	052050	047051
013276	047116	040045	

MES4: .ASCII '"XTYPE LETTER () TO RUN DESIRED TEST: "'

.ASCII '"(L)OGIC, (D)DATA, (S)SYNC (T)NNX@"'

013302	041536	040045	
013306	040536	100	

CNTRLC: .ASCII 'tC@'

CNTRLA: .ASCII 'tA@'

013311 045 100

CRLF: .ASCII 'x@'

013313 045 040056

DOT: .ASCII '%.@'

013316 020077 100

QMARK: .ASCII '?@'

013321 045 046042 043517

MES5: .ASCII 'x"LOGIC TEST"x@'

013326	041511	052040	051505		
013334	021124	040045			
013340	021045	042524	052123	MES6:	.ASCII ;%"TEST COMPLETE"@";
013346	041440	046517	046120		
013354	052105	021105	100		
013361	045	042042	052101	MES7:	.ASCII '%"DATA UPDATE TEST"@"'
013366	020101	050125	040504		
013374	042524	052040	051505		
013402	021124	100			
013405	123	050125	046120		.ASCII ;SUPPLY THE FOLLOWING VOLTAGES TO CH. '0'.;
013412	020131	044124	020105		
013420	047506	046114	053517		
013426	047111	020107	047526		
013434	052114	043501	051505		
013442	052040	020117	044103		
013450	020056	030047	027047		
013456	054524	042520	023440	MES44:	.ASCII ;TYPE 'CR' TO START TEST.%@";
013464	051103	020047	047524		
013472	051440	040524	052122		
013500	052040	051505	027124		
013506	040045				
013510	051445	050125	046120	MES8:	.ASCII "%SUPPLY '+5V' !@"
013516	020131	025447	053065		
013524	020047	040041			
013530	052523	050120	054514	MES9:	.ASCII "%SUPPLY '-5V' !@"
013536	023440	032455	023526		
013544	020440	100			
013547	045	051461	020124	MES10:	.ASCII '%"1ST WRD'@"'
013554	051127	020104	100		
013561	045	047062	020104	MES11:	.ASCII '%"2ND WRD'@"'
013566	051127	020104	100		
013573	045	051063	020104	MES12:	.ASCII '%"3RD WRD'@"'
013600	051127	020104	100		
013605	045	052064	020110	MES13:	.ASCII '%"4TH WRD'@"'
013612	051127	020104	100		
013617	045	051045	041505	MES21:	.ASCII '%"RECOVERED FROM POWER FAILURE'@"'
013624	053117	051105	042105		
013632	043040	047522	020115		
013640	047520	042527	020122		
013646	040506	046111	051125		
013654	020105	100			
013657	045	046111	042514	MES40:	.ASCII ;%ILLEGAL TRAP TO @;
013664	040507	020114	051124		
013672	050101	052040	020117		
013700	100				
013701	040	051106	046517	MES41:	.ASCII ; FROM @;
013706	040040				
013710	054105	027124	044455	MES42:	.ASCII ;EXT.-INT. TEST COMPLETE %@";
013716	052116	020056	042524		
013724	052123	041440	046517		
013732	046120	052105	020105		
013740	040045				
013742	047503	053116	051105	MES43:	.ASCII ;CONVERSION SYNC TEST %@";
013750	044524	047117	051440		

E06

ADF11 PART I, LOGIC DIAGNOSTIC TEST
DZADGA.CMB

MACY11 27(732) 26-OCT-76 16:52 PAGE 70

013756	047131	020103	042524
013764	052123	022440	100

013772

.EVEN

;ADDRESS AND CONSTANTS TABLE

013772	000000	ADSIGN: 0	: UNIPOLAR=0, BIPOLAR=1
013774	000000	SIGNBF: 0	: A/D WORD LENGTH
013776	177700	DELAY1: -100	: DELAY COUNT FOR LOGIC TEST
014000	177634	DELAY2: -100.	: DELAY COUNT FOR LOGIC TEST
014002	001000	STACK: 1000	: INITIAL SP. ADDRESS
014004	001546	AVECTR: INITB	: 'IA' VECTOR ADDRESS
014006	001720	PVECTR: INIT3	: 'IP' VECTOR ADDRESS
014010	000000	SAVEPC: 0	
014012	000000	SAVPSM: 0	
014014	000000	SAV2PC: 0	
014016	000000	SAV2SM: 0	
014020	000000	INCFLG: 0	: SOFTWARE FLAG; 0=NO INC. MEM.
014022	000000	MEMSIZ: 0	: CALCULATED MEM SIZE TO SUPPORT INC. MEM.
014024	000000	CORMAX: 0	: CALCULATED MEMORY SIZE
014026	000000	SOFLAG: 0	: SOFTWARE 'FLAG'
014030	000000	ADSIZE: 0	: OCTAL STORAGE OF A/D LENGTH
014032	000000	PROC: 0	: TEMP STORAGE FOR 'PSM'
014034	000000	CHRCNT: 0	: TEMP STORAGE
014036	000000	COUNT: 0	: TEMP STORAGE
014040	000000	KSTOR1: 0	: PERMANENT STORAGE
014042	000000	KSTOR2: 0	: PERMANENT STORAGE
014044	000000	KSTOR3: 0	: PERMANENT STORAGE
014046	000000	KSTOR4: 0	: PERMANENT STORAGE
014050	000000	TEMP1: 0	: TEMPORARY STORAGE
014052	000000	TEMP2: 0	: TEMPORARY STORAGE
014054	000000	TEMP3: 0	: TEMPORARY STORAGE
014056	000000	MESPRT: 0	
014060	000000	HIGH: 0	
014062	000000	LOW: 0	
014064	000000	GMD7XB: 0	
014066	000000	GMD3XB: 0	
:HERE STARTS THE '1000' WORD STATUS WORD BUFFER			
014070	000000	RANBUF: 0	
	015072		: = +1000
:HERE STARTS THE WORD A/D DATA BUFFER.			
015072	000000	ADBUF: 0	
	001332		
.END INIT			

TAGAN	007012	1968	1983#						
TAGAN	006046	1768#	1771						
TAGAO	007220	2030	2036#						
TAGAP	007276	2053#							
TAGAQ	007424	2076	2080#						
TAGAR	007446	2085#	2090						
TAGAS	007500	2097#	2101						
TAGF	003700	1251#	1259						
TAGG	003730	1263#							
TAGL	004526	1454#	1460						
TAGM	004544	1458	1461#						
TAGO	004604	1475#	1481						
TAGP	004622	1479	1483#						
TAGRAB	006562	1905	1915#						
TAGRAN	006500	1886	1893#						
TAGXF	004014	1295#	1300						
TAGXJ	005464	1653#	1658						
TAGXK	004250	1372#	1377						
TAGXL	004106	1324#	1329						
TAGXM	004312	1385#	1390						
TAGXN	005766	1746#	1749						
TAGYA	006164	1801#	1804						
TAGYB	006206	1807#	1809						
TAGYC	006266	1826#	1829						
TAGYF	007120	1996	2015#						
TAGYK	005570	1677	1685#						
TAGYN	005716	1728#	1731						
TAG1B	007574	2117	2126#						
TAG1C	007612	2130#	2135						
TEMP1	014050	2617#	2622#	2624	2768#	2775#	2969#		
TEMP2	014052	2619#	2627#	2629	2970#				
TEMP3	014054	2971#							
TESTX	010610	776	2399#	2412					
TITLE	012755	698	2829#						
TKB	001274	673#	2451						
TKS	001272	672#	2449	2682#	2705#	2810			
TKSFLG	012712	663	2810#						
TPB	001300	675#	2427#	2463#	2692#	2694#	2697#	2720#	2822#
TPS	001276	674#	2185	2425	2461	2684	2695	2718	2820
TSTA	010630	2403#	2410						
TSTB	010664	2406	2412#						
TSTTKS=	104006	631#	2730	2790					
TSTXB	003044	1059	1062#						
TST1	002014	789	803#	2806					
TST10	002136	851#							
TST100	006252	1822#							
TST101	006344	1860#							
TST102	006720	1966#							
TST103	007016	1994#							
TST104	007122	2018#							
TST105	007300	2057#							
TST106	007514	2108#							
TST107	007632	2141#							
TST11	002162	860#							
TST12	002206	869#							
TST13	002226	877#							

TST14	002252	887#
TST15	002276	898#
TST16	002316	909#
TST17	002336	918#
TST2	002020	806#
TST20	002356	927#
TST21	002410	939#
TST22	002436	949#
TST23	002472	962#
TST24	002524	975#
TST25	002542	984#
TST26	002560	992#
TST27	002604	1003#
TST3	002030	813#
TST30	002632	1012#
TST31	002702	1026#
TST32	002752	1039#
TST33	003022	1056#
TST34	003114	1078#
TST35	003170	1097#
TST36	003236	1113#
TST37	003304	1128#
TST4	002042	820#
TST40	003352	1144#
TST41	003372	1153#
TST42	003414	1163#
TST43	003430	1171#
TST44	003456	1184#
TST45	003512	1197#
TST46	003554	1212#
TST47	003604	1224#
TST5	002054	827#
TST50	003650	1238#
TST51	003670	1246#
TST52	004004	1292#
TST53	004076	1321#
TST54	004240	1369#
TST55	004340	1397#
TST56	004372	1413#
TST57	004434	1429#
TST6	002066	833#
TST60	004506	1449#
TST61	004564	1470#
TST62	004640	1491#
TST63	004674	1506#
TST64	004726	1517#
TST65	004776	1531#
TST66	005046	1545#
TST67	005116	1560#
TST7	002112	842#
TST70	005166	1576#
TST71	005320	1614#
TST72	005530	1672#
TST73	005616	1700#
TST74	005702	1724#
TST75	005752	1742#

.SRAND	18
.SRDOE	18
.SRDOC	18
.SREAO	18
.SR2A2	18
.SSAVE	18
.SSB2D	18
.SSB2O	18
.SSCOP	18
.SSIZE	18
.SSUPR	18
.STRAP	18
.STYPB	18
.STYPO	18
.STYPE	18
.STYPO	18
.SHOCA	18
.1170	18

E07

JMP	620	621	653	767	770	773	776	2145	2394	2414	2418	2469	2476	2545	2577
JSR	2630	2800													
	757	798	1058	1065	1075	1080	1090	1099	1109	1115	1125	1130	1140	1229	1509
	1520	1534	1548	1563	1579	1617	1624	1640	1675	1676	1687	1703	1706	1715	1734
	1751	1773	1811	1833	1869	1885	1976	2004	2029	2038	2044	2075	2082	2115	2116
MOV	2128	2157	2167	2170	2249	2256	2257	2281	2284	2323	2326	2358	2367	2399	
	643	645	652	695	696	704	705	708	719	726	727	730	732	735	741
	742	743	756	761	786	787	788	789	790	799	835	844	853	862	871
	879	880	889	920	930	954	1057	1059	1067	1069	1082	1084	1102	1103	1117
	1119	1133	1134	1155	1156	1189	1202	1216	1228	1230	1240	1250	1251	1263	1264
	1294	1295	1296	1311	1323	1333	1334	1335	1336	1344	1351	1358	1371	1378	1379
	1381	1384	1386	1399	1400	1415	1416	1431	1432	1451	1452	1453	1461	1462	1472
	1473	1474	1483	1493	1497	1508	1510	1511	1519	1521	1522	1523	1533	1535	1536
	1537	1547	1549	1550	1551	1562	1564	1565	1566	1578	1580	1581	1582	1585	1586
	1587	1593	1594	1604	1605	1616	1619	1620	1621	1626	1628	1629	1632	1651	1652
	1661	1662	1674	1678	1680	1702	1705	1708	1709	1726	1727	1728	1733	1735	1744
	1745	1746	1750	1753	1766	1767	1768	1772	1774	1799	1800	1801	1805	1806	1807
	1810	1812	1824	1825	1826	1830	1832	1834	1863	1864	1868	1871	1872	1873	1874
	1876	1887	1888	1894	1896	1896	1905	1906	1920	1921	1931	1932	1935	1936	1938
	1946	1948	1970	1974	1975	1977	1978	1986	1998	1999	2000	2003	2005	2008	2020
	2030	2031	2034	2035	2036	2031	2033	2041	2046	2059	2060	2061	2063	2065	2066
	2070	2071	2071	2072	2077	2083	2084	2094	2098	2110	2111	2114	2118	2119	2120
	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143	2144
	2158	2164	2164	2165	2171	2179	2184	2190	2191	2192	2193	2194	2195	2196	2197
	2200	2201	2202	2203	2204	2205	2206	2207	2208	2209	2210	2211	2212	2213	2214
	2220	2221	2222	2223	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234
	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254
	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271	2272	2273	2274
	2280	2281	2282	2283	2284	2285	2286	2287	2288	2289	2290	2291	2292	2293	2294
	2300	2301	2302	2303	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314
	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334
	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351	2352	2353	2354
	2360	2361	2362	2363	2364	2365	2366	2367	2368	2369	2370	2371	2372	2373	2374
	2380	2381	2382	2383	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394
	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414
	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431	2432	2433	2434
	2440	2441	2442	2443	2444	2445	2446	2447	2448	2449	2450	2451	2452	2453	2454
	2460	2461	2462	2463	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474
	2480	2481	2482	2483	2484	2485	2486	2487	2488	2489	2490	2491	2492	2493	2494
	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511	2512	2513	2514
	2520	2521	2522	2523	2524	2525	2526	2527	2528	2529	2530	2531	2532	2533	2534
	2540	2541	2542	2543	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554
	2560	2561	2562	2563	2564	2565	2566	2567	2568	2569	2570	2571	2572	2573	2574
	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591	2592	2593	2594
	2600	2601	2602	2603	2604	2605	2606	2607	2608	2609	2610	2611	2612	2613	2614
	2620	2621	2622	2623	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634
	2640	2641	2642	2643	2644	2645	2646	2647	2648	2649	2650	2651	2652	2653	2654
	2660	2661	2662	2663	2664	2665	2666	2667	2668	2669	2670	2671	2672	2673	2674
	2680	2681	2682	2683	2684	2685	2686	2687	2688	2689	2690	2691	2692	2693	2694
	2700	2701	2702	2703	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714
	2720	2721	2722	2723	2724	2725	2726	2727	2728	2729	2730	2731	2732	2733	2734
	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751	2752	2753	2754
	2760	2761	2762	2763	2764	2765	2766	2767	2768	2769	2770	2771	2772	2773	2774
	2780	2781	2782	2783	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794
	2800	2801	2802	2803	2804	2805	2806	2807	2808	2809	2810	2811	2812	2813	2814
	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831	2832	2833	2834
	2840	2841	2842	2843	2844	2845	2846	2847	2848	2849	2850	2851	2852	2853	2854
	2860	2861	2862	2863	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874
	2880	2881	2882	2883	2884	2885	2886	2887	2888	2889	2890	2891	2892	2893	2894
	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911	2912	2913	2914
	2920	2921	2922	2923	2924	2925	2926	2927	2928	2929	2930	2931	2932	2933	2934
	2940	2941	2942	2943	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954
	2960	2961	2962	2963	2964	2965	2966	2967	2968	2969	2970	2971	2972	2973	2974
	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991	2992	2993	2994
	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007	3008	3009
MOV8															
NEG															
NOB															
RESET															
ROL															
RTI															
RTS															
SUB															
TRAP															
TST															
	646	722	733	737	739	807	814	821	828	891	901	935	946	958	970
	978	987	996	999	1007	1015	1029	1042	1159	1168	1176	1180	1193	1209	1235
	1244	1280	1304	1317	1343	1350	1354	1357	1385	1403	1419	1423	1435	1445	1456
	1477	1496	1716	1967	1995	2009	2272	2296	2337	2381	2479	2483	2513	2532	2534
	2741	2758	2780												
TSTB	902	1091	1174	1274	1314	1638	1688	1753	1775	1813	1835	2185	2191	2425	2

	919	928	940	950	963	976	985	993	1004	1013	1027	1040	1057	1079	1098
	1114	1129	1145	1154	1164	1172	1185	1198	1213	1225	1239	1249	1293	1322	1370
	1398	1414	1430	1450	1471	1492	1507	1518	1532	1546	1561	1577	1615	1673	1701
	1725	1743	1765	1798	1823	1861	1967	1995	2019	2058	2109	2142			
.MACRO	1	622													
.NLIST	1	622	807	814	821	828	834	843	852	861	870	878	888	899	910
	919	928	940	950	963	976	985	993	1004	1013	1027	1040	1057	1079	1098
	1114	1129	1145	1154	1164	1172	1185	1198	1213	1225	1239	1249	1293	1322	1370
	1398	1414	1430	1450	1471	1492	1507	1518	1532	1546	1561	1577	1615	1673	1701
	1725	1743	1765	1798	1823	1861	1967	1995	2019	2058	2109	2142			
.REM	1														
.REPT	349														
.TITLE	297														

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

* DZADGA.SEG=SYSMAC.CO,DZADGA.CMB
RUN-TIME: 27 37 4 SECONDS
RUN-TIME RATIO: 128/70=1.8
CORE USED: 33K (65 PAGES)

10			...B1	007644	004767	...B5
35			...C1			...C5
89			...D1	010100	005200	...D5
			...E1	010276	001373	...E5
148			...F1	010524	001376	...F5
			...G1	010640	005367	...G5
206			...H1	011054	120127	...H5
			...I1	011304	104012	...I5
266			...J1	011514	012767	...J5
322		000001	...K1	011662	042777	...K5
355	000014	000016	...L1	012012	010346	...L5
411	000174	000176	...M1	012146	100375	...M5
467	000354	000356	...N1	012400	010146	...N5
523	000534	000536	...B2	012620	000002	...B6
579	000714	000716	...C2	013026	045055	...C6
		104012	...D2	013405	123	...D6
	001216	000000	...E2			...E6
	001354	104000	...F2	014012	000000	...F6
	001610	016706	...G2			...G6
	001760	005067	...H2	DATA3	007726	...H6
	002114	104011	...I2	MEMSIZ	014022	...I6
	002252	104002	...J2			...J6
	002406	005713	...K2	SPCHR1	011122	...K6
	002466	104400	...L2	TAGL	004526	...L6
			...M2	TST24	002524	...M6
	003010	022777	...N2	X =	000110	...N6
	003146	001376	...B3	REPORT	1#	CROSS RE ...B7
	003336	001376	...C3	SSIZE	1#	CROSS RE ...C7
	003452	104400	...D3	BGT	2429	2521 ...D7
	003602	104011	...E3		1119	1132 ...E7
	003716	104400	...F3		1725	1743 ...F7
	004032	001370	...G3	**END**	USER DAVIES, TOM	...G7
	004166	022714	...H3			
	004334	001401	...I3			
	004462	001401	...J3			
	004570	012714	...K3			
	004722	104007	...L3			
	005102	104400	...M3			
			...N3			
	005360	005200	...B4			
			...C4			
	005702	104002	...D4			
	005772	022701	...E4			
	006170	022701	...F4			
			...G4			
	006542	012777	...H4			
			...I4			
			...J4			
	007020	005767	...K4			
	007166	012720	...L4			
	007350	017700	...M4			
	007530	022701	...N4			