

IP11, IP300

PROCESS CONTROL SUBSYSTEM
CVPCAB0

AH-A961B-MC
COPYRIGHT 77-78
FICHE 1 OF 1

JAN 1979
digital
MADE IN USA

This microfiche contains 100 frames of data, arranged in a 10x10 grid. Each frame displays a page of text, likely technical specifications or code, with some frames containing diagrams or tables. The text is too small to be legible in this image.

13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55

.SBTTL DOCUMENT
:*****
.REM !

IDENTIFICATION

PRODUCT CODE: AC-A959B-MC
PRODUCT NAME: CVPCABO PROCESS CTRL SS
DATE: 28-AUG-78
MAINTENANCE: DIAGNOSTIC GROUP
AUTHOR: H. SZEJNWALD

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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1977,1978 BY DIGITAL EQUIPMENT CORPORATION.

57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113

1.0 ABSTRACT

THIS PROGRAM IS A DIAGNOSTIC TOOL FOR TESTING THE ENTIRE FAMILY OF PCS MODULES. THE PROGRAM IS INTENDED TO BE USE BY FIELD ENGINEERING AND MANUFACTURING . BY USING THIS PROGRAM AN OPERATOR IS ABLE TO CHECK THE IOCM, AND ANY DIGITAL OR ANALOG MODULE. SINCE THE ASSUMPTION IS MADE THAT ALL OUTPUT MODULES ARE CONNECTED TO THE CUSTOMERS WIRING DURING THE TEST OF OUTPUT MODULES THE DISABLE BIT IS SET. THIS DIAGNOSTIC WILL NOT CHECK THE PART OF HARDWARE THAT IS INTERFACING DIGITAL OR ANALOG MODULES WITH CUSTOMERS WIRING (DRIVING TRANSISTOR, ISOLATING TRANSFORMERS, PHOTO-COUPPLERS ECT). THIS DIAGNOSTIC DOES NOT REQUIRE ANY EXTERNAL DEVICES OR SPECIAL CONNECTIONS.

2.0 HARDWARE REQUIREMENTS

1. LSI 11 WITH 16K OF MEMORY OR ANY PDP11 CPU WITH UNIBUS BRIDGE INTERFACE.
2. CONSOLE TERMINAL
3. FLOPPY DISC OR SOME OTHER INPUT DEVICE
4. IOCM (M7958)

3.0 PROGRAM CONSIDERATION

3.1 CPU COMPATIBILITY

THIS PROGRAM CAN BE USED BY THE LSI 11 OR BY THE PDP-11 /04,05,10,20,34,35,40,45,50 & 70 IF UNIBUS BRIDGE MODULE IS USED.

3.2 XXDP

THIS PROGRAM CAN BE CHAINED BY XXDP & WILL NOT OVERLAY THE LOADER

CHAIN MODE OPERATION

1. THE INPUT DIALOGUE WITH AN OPERATOR IS BYPASSED
2. THE SUBSYSTEM IS MAPPED IN MEMORY
3. THE SYSTEM TEST IS AUTOMATICALLY EXECUTED

NORMAL OPERATION

1. START AT LOC 204
2. SELECT TEST
3. PROGRAM RUNS AND GOES BACK TO MONITOR

3.3 ACT/APT

THIS PROGRAM IS ACT COMPATIBLE TO THE EXTENT THAT APT HOOKS WILL BE IN THE PROGRAM & WILL WORK THRU THE 'UPTON INTERFACE'

114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170

3.4 EXECUTION TIME

MOST INDIVIDUAL TESTS TAKE LESS THEN 1 SEC. THE ONESHOT MODULE TEST
TAKES 15 SEC. THE SYSTEM TEST EXECUTION TIME DEPENDS
ON THE NUMBER AND TYPE OF I/O MODULES.
THE TEST OF ANALOG MODULES TAKES ALSO UP TO 1 MIN PER MODULE.

3.5 DEFAULT ADDRESSES & VECTORS

THE FOLLOWING IS A LIST OF THE DEFAULT ADDRESSES & VECTORS
OF ALL HARDWARE TO BE USED AND THEIR CORRESPONDING MEMORY LOCATIONS.
IN THE CASE THAT THE IOCM IS SET TO AN ADDRESS OTHER THAN
171000 THESE LOCATIONS MUST BE CHANGED.
BASE: .WORD 171000 ;FIRST ADDRESS OF I/OADDRESS BLOCK
CSR: .WORD 171377 ;ADDRESS OF IOCM CSR REGISTER
IAR: .WORD 171376 ; IAR REGISTER
VECTO: .WORD 234 ;INTERRUPT VECTOR OF IOCM
VECTOA: .WORD 236 ;INTERRUPT VECTOR+2 OF IOCM

4.0 OPERATING PROCEDURE

4.1 PROGRAM LOADING

THE PROGRAM CAN BE LOADED FROM PAPER TAPE OR FLOPPY
DISK USING STANDARD PROCEDURES.

4.2 PROGRAM STARTING

LOCATION 200 - STARTING ADDRESS TO RUN THE DIAGNOSTICS MONITOR.
THE PROGRAM WILL PRINT OPTIONS AVAILABLE TO
THE OPERATOR.
LOCATION 204 - STARTING ADDRESS OF MONITOR WITHOUT PRINTING
OPERATOR'S OPTIONS.

4.3 INPUT DIALOGUE

IF AN OPERATOR STARTS THE PROGRAM AT LOCATION 204 HE CAN
SELECT ONE OF FOLLOWING OPTIONS:

4.3.1 S-SYSTEM TEST

THIS OPTION OF THE PROGRAM WILL MAP ALL THE ANALOG
AND DIGITAL MODULES CONNECTED TO THE IOCM AND BUILD
A TABLE OF THEM IN MEMORY. THEN IT WILL PICK UP
EACH ONE AND RUN THE TESTS THAT DO NOT REQUIRE OPERATOR INTERVENTION. IF SWITCH 14

171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227

THE SOFTWARE SWITCH REGISTER IS SET ,IT WILL LOOP ON
CURRENT TEST UNTIL OPERATOR TYPES CONTROL C. AFTER THE SELECTED # OF
PASSES, THE PROGRAM WILL PRINT PASS COUNT UNLESS SWITCH 13 IS SET.

4.3.2 M - MAP OF DBUS

THIS OPTION ALLOWS AN OPERATOR TO CHECK WHICH I/O MODULE
ARE CONNECTED TO THE IOCM. IT CHECKS ALL ADDRESSES BETWEEN
171000 AND 171375. FOR ADDRESSES THAT ANSWER , IT CHECKS THE
GENERIC CODE AND TYPES THE ADDRESS AND INTERFACE TYPE.

4.3.3 D - TEST DIGITAL MODULE

THIS TEST WILL EXERCISE THE DIGITAL MODULE AT THE ADDRESS
SPECIFIED BY THE OPERATOR. FOLLOWING IS A DESCRIPTION OF
THE DIGITAL MODULE TESTS FOR EACH MODULE TYPE..
EACH TEST IS RUN AS MANY TIMES AS SELECTED BY THE L OPTION
AND THE PASS COUNT IS PRINTED.
OPERATOR MUST TYPE THE ADDRESS OF MUT. FIRST.

M5010:

1. SET D BIT
2. CHECK THAT ALL BITS ARE ZERO
3. SET T BIT
4. CHECK THAT ALL BITS ARE ONES
5. SET C BIT

M5011:

1. SET DBIT AND TBIT
2. CHECK IF INPUTS ARE ALL ONES
3. CLEAR TBIT
4. CHECK IF INPUTS ARE ALL ZEROS
5. CLEAR ALL INTERRUPTS
6. SET TBIT
7. ENABLE INTERRUPT
8. CHECK IF INTERRUPT OCCURRED
9. CHECK IF INPUTS ARE ALL ONES
10. CHECK IF COS REGISTERS ARE ALL ONES
11. SET RIF BIT
12. CLEAR ALL COS REGISTERS
13. CHECK IF THEY ARE CLEAR
14. CLEAR T BIT
15. CHECK IF INTERRUPT OCCURED
16. CHECK IF ALL INPUTS ARE ZERO
17. RUN STEP 10 TO 13
18. CLEAR ALL INTERRUPTS

M5012-M5012YA:

1. SET DBIT
2. CHECK IF ALL BITS ARE ZERO

- 228
 - 229
 - 230
 - 231
 - 232
 - 233
 - 234
 - 235
 - 236
 - 237
 - 238
 - 239
 - 240
 - 241
 - 242
 - 243
 - 244
 - 245
 - 246
 - 247
 - 248
 - 249
 - 250
 - 251
 - 252
 - 253
 - 254
 - 255
 - 256
 - 257
 - 258
 - 259
 - 260
 - 261
 - 262
 - 263
 - 264
 - 265
 - 266
 - 267
 - 268
 - 269
 - 270
 - 271
 - 272
 - 273
 - 274
 - 275
 - 276
 - 277
 - 278
 - 279
 - 280
 - 281
 - 282
 - 283
 - 284
3. SET TBIT
 4. CHECK IF ALL BITS ARE ONE
 5. CLEAR T BIT
 6. CLEAR ALL INTERRUPTS
 7. SET T BIT
 8. ENABLE INTERRUPT
 9. CHECK IF INTERRUPT OCCURRED
 10. SET RIF BIT
 11. CLEAR INTERRUPT
 12. CHECK IF INTERRUPTS ARE CLEAR
 13. CLEAR T BIT
 14. RUN STEP 9 TO 12
 15. CLEAR ALL INTERRUPTS

M5013:

1. SET DBIT
2. CHECK IF ALL BITS ARE ZERO
3. SET TBIT
4. TEST IF ALL BITS ARE ONE
5. CLEAR T BIT
6. CLEAR ALL INTERRUPTS
7. SET T BIT
8. ENABLE INTERRUPT
9. CHECK IF INTERRUPT OCCURED
10. SET RIF BIT
11. CLEAR INTERRUPT
12. CHECK IF INTERRUPT IS CLEAR
13. CLEAR T BIT
14. RUN STEP 9 TO 12
15. CLEAR ALL INTERRUPTS

M6010-M6010YA:

1. SET D BIT
2. CLEAR ALL 4 BYTES OF I/O REGISTER
3. SET DATA PATTERN 125 IN FIRST BYTE
4. CHECK IF IT IS SET
5. CHECK IF OTHER BYTES ARE ZERO
6. DO THE SAME WITH DATA 252 , 377 , 000
7. DO THE SAME WITH OTHER BYTES

M6011:

THE LINE CLOCK MUST BE ENABLED

1. SET D BIT
2. SET DATA PATTERN 252 AT MUT
3. CHECK IF IT IS SET
4. WAIT UP TO 10 SEC
5. DATA SHOULD BE CLEAR
6. REPEAT STEP 2 TO 5 WITH DATA EQUAL TO 125

M6012:

M6013:

285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341

1. SET D BIT
2. SET OUTPUT REGISTER TO FOLLOWING DATA PATTERN
0,377,252,125
3. CHECK IF IT IS SET

4.3.4 I - TEST IOCM

THIS TEST WILL EXERCISE ALL THE FEATURES OF THE IOCM. THE FOLLOWING IS A DESCRIPTION OF ALL THE TESTS.
THIS TEST IS RUN AS MANY TIMES AS SELECTED AND THE PASS COUNT IS TYPED

- TST1: CHECKS IF EACH BIT OF THE IOCM IS CLEARED BY THE CBIT
- TST2: CHECKS IF THE INTERRUPT ENABLE BIT (EBIT) CAN BE SET AND CLEARED
- TST3: CHECKS IF THE MAINTENANCE BIT (MBIT) CAN BE SET AND CLEARED
- TST4: CHECKS IF THE DISABLE BIT (DBIT) CAN BE SET AND CLEARED
CHECKS IF DBIT GENERATES CLEAR
- TST5: CHECKS IF THE TEST BIT (TBIT) CAN BE SET AND CLEARED
- TST6: CHECKS IF THE GENERIC CODE BIT (GBIT) CAN BE SET AND CLEARED
- TST7: CHECKS IF THE RIF BIT (RBIT) CAN BE SET AND CLEARED
- TST10: CHECKS DBUS DATA PATHS IN A MAINTENANCE MODE. IF THE MBIT IS SET AND THE CPU ADDRESSES ANY LOCATION BETWEEN 171000 AND 171375 IT SHOULD READ BACK THE LOWER BYTE OF THE MODULE ADDRESS.
- TST11: CHECKS MAINTENANCE INTERRUPT. IF THE MBIT & EBIT ARE SET, THE IOCM WILL GENERATE AN INTERRUPT WITH VECTOR 234. THE IAR (171376) HAS THE LOWER BYTE OF CSR ADDRESS (377).

4.3.5 W -WRAP AROUND TEST

THIS OPTION ALLOWS THE FIELD ENGINEER TO CONNECT MODULE M6010 TO EITHER M5010 OR M5011. THEN IT WRAPS AROUND A SET OF DATA PATTERNS (125,252,377,0). IF SWITCH 14 IN THE SOFTWARE SWITCH REGISTER IS SET, IT WILL LOOP ON THIS TEST UNTIL THE OPERATOR TYPES CONTROL C. EVERY SELECTED # OF PASSES IT WILL PRINT PASS COUNT UNLESS SWITCH 13 IS SET. ERRORS WILL BE PRINTED INDICATING GOOD DATA, BAD DATA AND THE PASS # AT WHICH IT OCCURRED.

4.3.6 F -FIELD TEST

THE PURPOSE OF THIS TEST IS:
A. TO OUTPUT ANY SELECTED DATA PATTERN TO AN OUTPUT MODULE SPECIFIED BY OPERATOR.
B. TO MONITOR DATA FROM AN INPUT MODULE SPECIFIED BY OPERATOR.

342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359

WARNING: THE FIELD ENGINEER MUST REALIZE THAT THIS IS THE ONLY TEST
THAT PERMITS HIM TO OUTPUT DATA TO THE CUSTOMER'S WIRING.
THERE WILL BE VOLTAGE APPLIED TO CUSTOMER EQUIPMENT CONNECTED
TO THE OUTPUT MODULE UNDER TEST.
THEREFORE PRECAUTIONS MUST BE TAKEN THAT AN ERRONEOUS OUTPUT WILL
NOT CAUSE DAMAGE TO THE FIELD EQUIPMENT AND THAT ALL TESTING IS
CONDUCTED WITH THE CUSTOMERS KNOWLEDGE

*****8
PROCEDURE: USER SELECTS THE ADDRESS OF THE MODULE UNDER TEST.
IF IT IS AN OUTPUT MODULE THEN:

USER SELECTS AND OUTPUTS A DATA PATTERN, ONE BYTE AT A TIME,
TO THE MODULE. TYPE CONTROL/C TO ABORT TEST.
AFTER ALL BYTES OF MODULE HAVE BEEN SELECTED, USER MUST TYPE CONTROL/C
TO GO BACK TO MONITOR.

361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417

EXAMPLE:

TYPE ADDRESS OF MUT 171XXX
TYPE ONE BYTE DATA PATTERN YYY
(WHERE YYY IS A 8 BIT DATA PATTERN TO OUTPUT.
ALSO, THE REQUEST FOR DATA FOR OUTPUT WILL BE REPEATED UP TO
4 TIMES DEPENDENT UPON MODULE TYPE)

IF IT IS AN INPUT MODULE THEN:

THE TEST MONITORS AND PRINTS DATA FROM MODULE INPUTS.
IT PRINTS EVERY BYTE ADDRESS OF THE MODULE WITH ITS
CONTENT DURING THE FIRST PASS AND IT CONTINUES TO
MONITOR THE DATA WITHOUT PRINTING IT UNLESS A CHANGE
IN THE DATA OCCURED. TYPE CONTROL/C TO RETURN TO MONITOR.

EXAMPLE:

TYPE ADDRESS OF MUT 171XXX
171XXX : YYY
(WHERE YYY IS THE 8 BITS OF DATA READ FROM THE MODULE.
THE SECOND LINE OF THE MESSAGE WILL BE REPEATED
UP TO 4 TIMES, WITH INCREMENTING ADDRESS,
DEPENDING UPON MODULE TYPE.

4.3.7 L - SET PASS COUNT

TYPE THE OCTAL NUMBER OF ITERATIONS FOR ALL TESTS.
DEFAULT NUMBER IS 1.
MAXIMUM NUMBER IS 177777

4.3.8 A630 DIGITAL TO ANALOG CONVERTER (DAC)

4.3.8.1 AFTER TYPING A630 AN OPERATOR ENTERS THE MONITOR FOR
TESTING A630.HE HAS AN OPTION OF SELECTING ON OF THE
FOLLOWING TESTS:

- A - CALIBRATION TEST
- T - INTERNAL COMPARATOR TEST
- D - D BIT TEST (LOGIC TEST)

4.3.8.2 D-BIT TEST

THE D-BIT TEST CHECKS THE GENERIC CODE AND LOGIC PORTIONS OF THE
MODULE ONLY. IT DOES NOT EXERCISE ANY PORTION OF THE ANALOG
CIRCUITRY. THEREFORE, THE T-BIT TEST MUST BE RUN IN CONJUNCTION
WITH THE D-BIT TEST TO INSURE PROPER MODULE OPERATION.

4.3.8.3 T - INTERNAL COMPERATOR TEST

418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474

```

*****
*          CAUTION: THIS TEST CAUSES VOLTAGES TO APPEAR ON THE
*          MODULE'S OUTPUTS WHICH MAY BE DANGEROUS TO THE CUSTOMER'S
*          PROCESS. ALSO, OVERLOADS OR SHORT CIRCUITS IN THE FIELD WIRING
*          WOULD CAUSE A FAILURE OF THIS TEST. THEREFORE, THE I/O
*          CABLE MUST BE REMOVED FROM THE MODULE UNDER TEST BEFORE
*          PROCEEDING.
*****

```

THE INTERNAL STATUS BIT TEST ACTIVATES THE ANALOG PORTION OF THE MODULE AND UTILIZES ANALOG COMPARATORS TO MONITOR AND REPORT THE RELATIVE LEVELS OF THE VOLTAGE OUTPUTS. THIS TEST DOES NOT MONITOR THE CURRENT OUTPUTS AND PROVIDES NO INDICATION OF THEIR FUNCTIONALITY OR CALIBRATION. FAILURE TO PASS THIS TEST INDICATES THAT THE MODULE IS EITHER DEFECTIVE OR SEVERELY OUT OF CALIBRATION. PASSAGE OF THIS TEST DOES NOT INDICATE THAT THE MODULE'S VOLTAGE OUTPUTS ARE NECESSARILY WITHIN CALIBRATION.

WHEN AN ERROR IS DETECTED, THE EXPECTED AND ACTUAL STATUS BITS ARE PRINTED BY THE DIAGNOSTIC. THEY ARE PRESENTED IN RIGHT JUSTIFIED FORM I.E. 0017 INDICATES THAT ALL BITS ARE SET. A FAILURE IN ANY ONE CHANNEL WILL USUALLY CAUSE TWO STATUS BITS TO BE IN ERROR AT DIFFERENT TIMES IN THE DIAGNOSTIC ACCORDING TO THE TABLE BELOW.

FAULTED CHANNEL	STATUS BITS AFFECTED
CH 0	S1 AND/OR S3
CH 1	S1 AND/OR S2
CH 2	S2 AND/OR S4
CH 3	S3 AND/OR S4

4.3.8.4 A - TEST OR CALIBRATION PROCEDURE

NORMALLY THE DAC MODULE WILL BE CALIBRATED AND SEALED AT THE TIME OF MANUFACTURE. FIELD RECALIBRATION SHOULD ONLY BE ATTEMPTED WHEN THE CUSTOMER WISHES TO CHANGE THE CURRENT OUTPUT OPTION FROM THE 4 TO 20MA RANGE TO THE 0 TO 20MA RANGE OR A MALFUNCTION IS SUSPECTED. (THE MODULE IS SHIPPED WITH THE 4 TO 20MA RANGE SELECTED). BEFORE ATTEMPTING RECALIBRATION, IT IS IMPORTANT TO BECOME FAMILIAR WITH THE LOCATION OF THE VARIOUS ADJUSTMENTS AND POINTS WHERE THE TEST EQUIPMENT WILL BE ATTACHED. FOR THIS INFORMATION REFER TO THE TABLE BELOW AND DRAWING D-UA-A630-0-0 OR FIGURES 6 THROUGH 10 AND TABLES 4 AND 5 OF THE A630 HARDWARE MANUAL.

CHANNEL			
0	1	2	3
---	---	---	---

475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531

VOLTAGE OFFSET ADJUST	R35	R57	R79	R101
CURRENT GAIN ADJUST	R36	R58	R80	R102
CURRENT OFFSET ADJUST	R44	R66	R88	R110
CURRENT RANGE SWITCHES*	E78-3,4	E78-1,2	E80-3,4	E80-1,2
BERG PINS: VOLTAGE OUT	5	13	37	47
GND	6	14	38	48
CURRENT OUT	7	15	39	49
GND	8	16	40	50
SCREW TERMINALS: VOLTAGE OUT	5	13	21	31
GND	6	14	22	32
CURRENT OUT	7	15	23	33
GND	8	16	24	34

* WHEN CALIBRATING THE 4-20MA RANGE, THESE SWITCHES MUST BE 'OFF' FOR THE PARTICULAR CHANNEL BEING CALIBRATED. FOR THE 0-20MA RANGE, THESE SWITCHES MUST BE 'ON' FOR THE PARTICULAR CHANNEL BEING CALIBRATED. SWITCH E79-5 SHOULD BE 'ON' AT ALL TIMES.

BEFORE BEGINNING CALIBRATION OF THE DAC MODULE, THE FOLLOWING EQUIPMENT IS NEEDED:

- DVM WESTON SHLUMBERGER MODEL 443 OR EQUIVALENT
- EXTENDER MODULE W904B
- RESISTOR 500OHM, 0.01%, 0.3 WATT
REQUIRED FOR CURRENT CALIBRATION
DEC PART NO. 13-09985-00

ACCESS TO THE ADJUSTMENTS AND SWITCHES IS PERMITTED BY PUTTING THE DAC ON AN EXTENDER MODULE. CAUTION: THE SYSTEM MUST BE POWERED DOWN BEFORE INSERTING OR REMOVING ANY MODULE. ALSO, THE CUSTOMER'S FIELD WIRING MUST BE REMOVED FROM THE ASSOCIATED SCREW TERMINAL ASSEMBLY.

THE 'A' TEST OR CALIBRATION ROUTINE OF THE DIAGNOSTIC PROGRAM WILL SYSTEMATICALLY AND INTERACTIVELY LEAD THE OPERATOR THROUGH THE REQUIRED STEPS. THE SPECIFIC COURSE OF ACTION AT EACH STEP IS DETAILED BELOW. FOR EACH ADJUSTMENT IT SHOULD FIRST BE DETERMINED IF THAT PARTICULAR ADJUSTMENT NEEDS TO BE MADE BEFORE BREAKING THE SEAL ON THE ADJUSTMENT. THE A TEST AND CHANNEL # SHOULD NOW BE SELECTED. COMPLETE CALIBRATION OF THE SELECTED

532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588

CHANNEL WILL BE COMPLETED BEFORE CONSIDERING ANOTHER CHANNEL.

VOLTAGE REFERENCE ADJUST

THIS ADJUSTMENT IS COMMON TO ALL FOUR CHANNELS. IF THE REFERENCE HAS BEEN SET AND IS IN SPECIFICATION TYPE 'N' TO BYPASS THIS SECTION; OTHERWISE TYPE 'Y' TO PROCEED.

STEP 1: CONNECT THE - LEAD OF THE VOLTMETER TO EITHER PIN 6 OF THE BERG CONNECTOR OR PIN 6 OF THE SCREW TERMINAL ASSEMBLY AND THE + LEAD TO E62, PIN 14. THE READING SHOULD BE +10.240V +-2MV. IF THE READING IS WITHIN SPECIFICATION TYPE CARRIAGE RETURN (CR), OTHERWISE PROCEED AS FOLLOWS:

TRY TO BRING THE REFERENCE WITHIN SPECIFICATION BY ADJUSTING ONLY R135 WITHOUT ALTERING SWITCHES E78 1-4. NORMALLY ADJUSTING R135 SHOULD BE SUFFICIENT. IF THE REFERENCE CAN BE ADJUSTED TYPE CARRIAGE RETURN (CR) OTHERWISE PROCEED AS FOLLOWS.

SET R135 FULLY CLOCKWISE. USE SWITCHES E79-1 THRU 4 IN ANY COMBINATION TO OBTAIN A READING AS CLOSE TO BUT LESS THAN +10.240 V AS POSSIBLE. THE EFFECTS OF THE SWITCHES ARE BINARILY WEIGHTED WITH E79-4 BEING THE LEAST SIGNIFICANT. FINISH THE ADJUSTMENT BY ADJUSTING R135 TO ACHIEVE A READING OF +10.240 V. WHEN DONE TYPE CARRIAGE RETURN (CR).

STEP 2: REMOVE THE + LEAD OF THE VOLTMETER AND CONNECT TO E62 PIN 13. THE READING SHOULD BE +5.120 V +- 2MV. IF THE READING IS IN SPECIFICATION TYPE CARRIAGE RETURN (CR), OTHERWISE ADJUST R134 FOR THE PROPER READING. WHEN DONE TYPE CARRIAGE RETURN (CR).

CHANNEL VOLTAGE OUTPUT ADJUST

REMOVE THE + LEAD OF THE VOLTMETER AND CONNECT TO THE SPECIFIED VOLTAGE OUTPUT TERMINAL ON THE SCREW TERMINAL ASSEMBLY (SEE TABLE ABOVE). REMOVE THE - LEAD OF THE VOLTMETER AND CONNECT IT TO THE SCREW TERMINAL JUST BELOW THE + LEAD. BEFORE MAKING ANY ADJUSTMENTS LOCATE THE VARIOUS ADJUSTMENTS RELATED TO THE SELECTED CHANNEL. FOR THIS INFORMATION REFER TO THE TABLE ABOVE AND DRAWING UA-A630-0 OR FIGURE 7 OF THE A630 HARDWARE MANUAL.

STEP 1: THE VOLTMETER READING SHOULD BE 0.000V +-5MV. IF THE READING IS IN SPECIFICATION TYPE CARRIAGE RETURN (CR), OTHERWISE ADJUST THE PROPER VOLTAGE OFFSET POTENTIOMETER. WHEN DONE TYPE CARRIAGE RETURN (CR).

STEP 2: THE READING SHOULD CHANGE TO +10.230V +- 40MV. IF READING IS OUT OF SPECIFICATION REPLACE THE MODULE. THERE IS NO ADJUSTMENT FOR THIS STEP. TYPE CARRIAGE RETURN (CR).

CHANNEL CURRENT OUTPUT ADJUST

THE DIAGNOSTIC WILL ASK IF THE CURRENT OUTPUT IS TO BE

589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645

CALIBRATED. IF NOT, TYPE 'N' TO BYPASS THE REST OF THE PROCEDURE. IF IT IS TO BE CALIBRATED, TYPE 'Y' TO PROCEED. THE DIAGNOSTIC WILL THEN ASK IF THE 4-20MA RANGE IS TO BE CALIBRATED. IF IT IS, TYPE 'Y', CHECK TO SEE THAT SWITCH E79-5 IS ON AND BOTH OF THE SPECIFIED CURRENT RANGE SWITCHES FOR THAT CHANNEL ARE OFF. (SEE TABLE ABOVE) A 'N' ANSWER WILL SELECT THE 0-20MA RANGE. IN THIS CASE, TURN ON BOTH OF THE SPECIFIED CURRENT RANGE SWITCHES FOR THAT CHANNEL AND SWITCH E79-5.

NOTE: THE FOLLOWING PROCEDURE IS USED FOR BOTH CURRENT RANGES.

THE ADJUSTMENT OF THE CURRENT OUTPUT REQUIRES THE USE OF EITHER A 500OHM, 0.01% PRECISION RESISTOR (DEC PART NO. 13-09985-00) OR ACCURATE CURRENT RANGES ON THE VOLTMETER. IF THE RESISTOR IS AVAILABLE, CONNECT IT BETWEEN THE SPECIFIED CURRENT OUTPUT TERMINAL AND THE GROUND TERMINAL JUST BELOW ON THE SCREW TERMINAL ASSEMBLY. (SEE TABLE ABOVE) CONNECT THE VOLTMETER LEADS DIRECTLY ACROSS THE RESISTOR LEADS WITH THE + LEAD OF THE VOLTMETER CONNECTED TO THE SPECIFIED CURRENT OUTPUT TERMINAL. IF CURRENT RANGES ON THE VOLTMETER ARE TO BE USED IN LIEU OF THE RESISTOR, CONNECT THE METER DIRECTLY TO THE TWO TERMINALS SPECIFIED.

STEP 1: THE VOLTMETER READING SHOULD BE +10.00V +/- 5MV OR 20.000MA +/- 10MA. IF THE READING IS IN SPECIFICATION TYPE CARRIAGE RETURN (CR), OTHERWISE ADJUST THE PROPER CURRENT GAIN POTENTIOMETER. WHEN DONE TYPE CARRIAGE RETURN (CR).

STEP 2: THE VOLTMETER READING SHOULD CHANGE TO +2.000V +/- 5MV/4.000MA +/- 10UA IF IN THE 4-20MA RANGE OR 9.8MV +/- 5MV/19.5UA + 10UA IF IN THE 0-20 MA RANGE. IF THE READING IS IN SPECIFICATION TYPE CARRIAGE RETURN (CR), OTHERWISE ADJUST THE PROPER CURRENT OFFSET POTENTIOMETER. WHEN DONE TYPE CARRIAGE RETURN (CR).

THE DIAGNOSTIC WILL ASK IF THE CALIBRATION HAS BEEN REVERIFIED AND THE ADJUSTMENT OF EITHER POTENTIOMETER WILL AFFECT THE OTHER. THIS IS NECESSARY BECAUSE STEPS 1 AND 2 ABOVE ARE INTERACTIVE TYPE 'Y' ONLY IF STEPS 1 AND 2 CAN BE COMPLETED SEQUENTIALLY AND IN THAT ORDER WITHOUT ADJUSTING EITHER POTENTIOMETER. IF EITHER ADJUSTMENT WAS MADE ON THIS PASS, TYPE 'N'. THE DIAGNOSTIC WILL GO BACK TO STEP 1 OF THE CURRENT OUTPUT ADJUSTMENT. WHEN A 'Y' IS TYPED, CALIBRATION OF THAT CHANNEL IS COMPLETE AND THE DIAGNOSTIC RETURN TO ITS STARTING POINT.

4.3.9 A014 -ANALOG TO DIGITAL CONVERTER

4.3.9.1 THE PURPOSE OF THIS PROGRAM IS TO PERFORM THE TESTS OF THE A014 WITH OPERATOR INTERVENTION.

THE OPERATOR CAN CHOOSE ONE OF THE FOLLOWING TESTS:
D - LOGIC TEST

646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702

C - CALIBRATE AND VERIFY OFFSET AND GAIN OF A014,A156 OR A157
L - TEST THE LINEARITY OF THE CONVERTER
M - CHECK IF CONVERTER SKIPS OR MISSES ANY OUTPUT CODE 'MONOTONICITY TEST'
X - TEST IF A/D WORKS WITH A156 OR A157 MUX

4.3.9.2 OPERATING PROCEDURES:

WHEN ANALOG MODULE TEST IS SELECTED IT PRINTS:
WHICH TEST (D,C,L,M,X OR H FOR HELP)?

OPERATOR MUST RUN THE FOLLOWING TESTS IN THIS ORDER AND SHOULD TYPE:

4.3.9.3 D - A/D LOGIC TEST

THE LOGIC TEST VERIFIES THE FUNCTIONALITY OF THE A014 A/D
CONVERTER CONSISTING OF A MOTHER AND DAUGHTER BOARD AS FOLLOWS:

1. VERIFY ADDRESS RESPONSE
2. VERIFY BIT FUNCTIONALITY (ERROR BIT,DONE BIT, GO BIT)
3. VERIFY SINGLE ENDED MODE FUNCTIONALITY
CHANNEL SELECTION
GAIN SELECTION
4. VERIFY DIFFERENTIAL MODE FUNCTIONALITY
5. CHECK IF CONVERSION GETS DONE
6. VERIFY FUNCTIONS USING T-BIT
SET T-BIT AT IOCM
SELECT CHANNEL FOR +,-,OR 0 REFERENCE MAINTENANCE VOLTAGE
VERIFY INDIVIDUAL OUTPUTS
SET MAINTENANCE RAMP VOLTAGE
VERIFY THAT RAMP GOES FROM +10.240V TO -10.240V
5. SET D-BIT AT IOCM
6. VERIFY ZERO (4000) OUTPUT
8. CLEAR T-BIT AND D-BIT
9. RETURN TO MONITOR

4.3.9.4 C- CALIBRATE AND VERIFY A/D CONVERTER

TO ENSURE COMPLIANCE WITH SPECIFICATIONS, THE VOLTAGE SOURCE USED IN
CALIBRATING THE A014 MUST HAVE BEEN ACCURATELY CALIBRATED WITHIN
THE PREVIOUS SIX MONTHS. THE USE OF AN ACCURATE CUSTOMER PROVIDED
VOLTAGE REFERENCE IS RECOMMENDED WHEN AVAILABLE, AS DIFFERENCES BETWEEN
THE CUSTOMER'S REFERENCE AND THAT USED FOR CALIBRATION COULD INDUCE
AN ERROR DURING CUSTOMER USE.

IN ADDITION THE PERSONNEL DOING THE CALIBRATION SHOULD BE FAMILIAR
WITH THE PROCEDURES FOR ALIGNMENT OF PRECISION ANALOG EQUIPMENT.
IF YOU ARE NOT SURE OF THE FOREGOING, DO NOT ATTEMPT TO CALIBRATE
THE A014; ITS FACTORY CALIBRATION IS PROBABLY BETTER THEN YOU
WILL BE ABLE TO ACCOMPLISH.

IN THIS TEST OPERATOR SHOULD CONNECT HIS VOLTAGE SOURCE
TO THE INPUT OF THE A014,A156 OR A157 MUX ON THE SELECTED CHANNEL.
THE PROGRAM WILL MAKE 8 CONVERSIONS, CALCULATE THE AVERAGE
VALUE OF THIS CONVERSIONS AND PRINT THE RESULT IN OCTAL
FORMAT AND IN MILLIVOLTS. THE RESULTS WILL BE PRINTED EVERY
2 - 4 SEC. TO EXIT THIS TEST TYPE CONTROL A OR CONTROL C.
IN THE CASE OF THE A157 OPERATOR MUST ALSO SELECT THE GAIN.

R41 ADJUSTS THE ZERO OF THE A014. THIS IS BEST DONE AT A CODE

703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759

TRANSITION POINT, SO THAT MORE RESOLUTION THEN THE CONVERTER
QUANTIZATION LEVEL CAN BE OBTAINED.

R31 IS THE GAIN ADJUSTMENT POT; IT ADJUSTS BOTH PLUS AND MINUS
FULL SCALE SYMMETRICALLY ABOUT ZERO. AGAIN, BEST ADJUSTMENT RESOLUTION
CAN BE OBTAINED AT A CODE TRANSITION POINT.

R41 SHOULD ALWAYS BE ADJUSTED FIRST, USING AN INPUT VERY CLOSE TO ZERO.
R31 CAN BE ADJUSTED WITH AN INPUT CLOSE TO EITHER FULL SCALE OR TO
OPTIMIZE ACCURACY FOR A PARTICULAR APPLICATION, AT ANY
POINT IN THE INPUT RANGE SUFFICIENTLY FAR FROM ZERO.
FOR GREATEST AVERAGE ACCURACY OF AN EXPANDED SUBSYSTEM,
THE A014 SHOULD BE CALIBRATED WITH AN INPUT APPLIED THROUGH
THE ON-BOARD MULTIPLEXOR; IF IT IS DESIRED TO OPTIMIZE ACCURACY
THROUGH AN EXPANDER MULTIPLEXOR AT A SPECIFIC GAIN, THE
CALIBRATION INPUTS SHOULD BE APPLIED THROUGH THAT MULTIPLEXOR.
THE USER SHOULD RECOGNIZE THAT THIS MAY PLACE THE A014 ALONE OUT OF SPEC.

ZERO ADJUSTMENT -

1. APPLY -2.5 MV TO THE SCREW TERMINALS OF THE SELECTED
CHANNEL AND START THE CALIBRATION TEST.
2. VERIFY THAT THE OCTAL PRINTOUT VARIES BETWEEN 003777 AND 004000.
A 50 - 50 DISTRIBUTION WOULD BE IDEAL, BUT THIS IS NOT
USUALLY OBTAINABLE. IF NECESSARY, ADJUST R41 (LOWER POT)
UNTIL THE PRINTOUT VARIES PER THE ABOVE. THIS ADJUSTS
THE TRANSITION POINT FROM 0.0 MV TO -5.0 MV. (ONE LSB.)
3. APPLY +2.5 MV TO THE SAME CHANNEL AND VERIFY THAT THE PRINTOUT
IS 004000 OR 004001. VARIANCE IS NOT REQUIRED, BUT
IS ACCEPTABLE BETWEEN THESE TWO VALUES.

GAIN ADJUSTMENT -

1. APPLY A VOLTAGE OF +10.2325V (FULL SCALE MINUS HALF LSB)
TO THE SELECTED CHANNEL AND START THE CALIBRATION TEST.
2. VERIFY THAT THE PRINTOUT VARIES BETWEEN 007776 AND 007777.
IF NECESSARY ADJUST THE GAIN POT R31 UNTIL THE PROPER
PRINTOUT IS OBTAINED.
3. APPLY -10.2375V TO THE SELECTED CHANNEL (FULL SCALE MINUS HALF LSB)
AND RESTART THE CALIBRATION TEST IF NECESSARY.
4. VERIFY THAT THE OCTAL PRINTOUT IS EITHER 000001 OR 000000
OR VARYING BETWEEN THE TWO VALUES.

4.3.9.5 L- LINEARITY TEST

THIS IS A LINEARITY AND NOISE MEASUREMENT TEST USING
10V TEST RAMP. BEFORE RUNNING THIS TEST LOGIC TEST MUST BE RUN.

- A. PROGRAM CALCULATES THE AVERAGE OF CONVERSIONS PER STATE
BY DIVIDING THE TOTAL # OF CONVERSIONS THRU THE RAMP BY
4096 POINTS.
- B. COMPARES # OF CONVERSIONS FOR EACH STATE WITH LOW LIMIT
AND HIGH LIMIT AVERAGES

LOW LIMIT AVERAGE = AVERAGE/2-1

HIGH LIMIT AVERAGE = AVERAGE + AVERAGE/2+1

- C. STORES UP TO 20 ERRORS IN MEMORY, PRINTS ERROR MESSAGE IF
OF CONVERSIONS FOR ANY STATE IS BELOW THE LOW LIMIT OR ABOVE
THE HIGH LIMIT AND RETURN TO MONITOR

FOR EXAMPLE:

760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810

LINEARITY TEST ERROR			
TSTPNT	MIN	MAX	WAS(# OF CONVERSIONS)
7770	4	12	3
4000	4	12	13

IN THE EXEMPLE THE STATE 7770 WAS TOO NARROW AND THE STATE AT 4000 (OV) WAS TOO WIDE.
D. PRINTS A MESSAGE 'TOO MANY ERRORS' FOR 20 ERRORS OR MORE AND RETURN TO MONITOR
E. PRINTS A MESSAGE 'END OF RAMP' IF NO ERROR FOUND AND RETURN TO MONITOR
F. OPERATOR TYPES CNTRC TO TERMINATE ERRORS PRINTOUT AND RETURN TO MONITOR
G. AN EXCESSIVE # OF ERRORS INDICATES THAT THE CONVERTER IS NOT RESPONDING IN A LINEAR MANNER.

4.3.9.6 M - MONOTONICITY TEST

THE OBJECTIVE OF THIS TEST IS TO READ EVERY POSSIBLE STATE THRU THE RAMP AT LEAST ONCE STARTING FROM THE BOTTOM (-10.240), GOING UP THE RAMP (+10.240) AND THEN DOWN TO (-10.240). BEFORE RUNNING THIS TEST LOGIC TEST MUST BE RUN.

1. COMPARES LAST CONVERSION WITH THE PREVIOUS ONE AT ALL POINTS.
2. CONTINUE CONVERSION WITH NEXT POINT IN RAMP IF ANY ERROR IS FOUND.
3. PRINT ERRORS AT THE END OF THE RAMP, IF LESS THAN 20 ERRORS FOUND. THE ERRORS OCCUR IF THE PROGRAM DETECTS ANY STATE THAT IS MISSING OR ANY NOISE THAT IS MORE THEN 2 BITS.

EXAMPLE: OUT OF RANGE BY TWO BITS

GDDAT	BDDAT	RAMP(UP=0,DOWN=1)	
0	2	0	GOING UP THE RAMP
7776	7774	1	GOING DOWN THE RAMP

4. PRINTS A MESSAGE 'TOO MANY ERRORS' FOR 20 ERRORS OR MORE AND RETURN TO MONITOR.
5. PRINTS A MESSAGE 'END OF RAMP' FOR NO ERRORS FOUND
6. OPERATOR TYPES CNTR A TO TERMINATE ERRORS PRINTOUT AND RETURN TO MONITOR.

4.3.9.6 X - MULTIPLEXER TEST (A156, A157)

THE OPERATOR IS ASKED FIRST WHICH MUX HE WANTS TO TEST. THEN THE PROGRAM DETERMINES FROM THE GENERIC CODE IF IT IS A156 SINGLE ENDED, A156 DIFF MODE OR A157 AND IT CHECKS THE LOGIC OF THE SELECTED MUX.

THE TEST INCLUDES:

1. CHANNEL SELECT REGISTER
2. GAIN SELECT REGISTER
3. ERROR BIT FUNCTIONALITY
4. RESULTS OF THE CONVERSION WITH T BIT SET (4000)

812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836

4.3.10 T - SET SOFTWARE SWITCH REGISTER
TYPE THE OCTAL NUMBER TO BE LOADED TO SWREG

100000 - HALT ON ERROR
40000 - LOOP ON TEST
20000 - INHIBIT ERROR AND END OF PASS PRINT

4.3.11 CONTROL C

TYPING CONTROL C CAUSES THE PRESENT TEST TO BE ABORTED
AND PROGRAM RETURNS TO THE MONITOR.
IN SOME CASES IT MAY TAKE UP TO 20 SECONDS FOR THIS TO
HAPPEN.

4.3.12 CONTROL A

IN CASE THAT THE MODULE HAS AN INDIVIDUAL MONITOR (A014,A630)
BY TYPING CONTROL A DURING THE TEST EXECUTION THE PROGRAM RETURNS
TO THE BEGINING OF THIS MONITOR.

000000

000174 000174
000176 000000

000200 000137 004024
837 000100 000100
838 000100 025404
839 000102 000102
840 000102 000340
841 000042 000042
842 000042 000000
843 000046 000046
844 000046 006302
845 000204 000204
846 000204 000137 004336
847

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

.=100
NOCLK
.=102
340
.=42
HALT
.=46
LOGIC
.=204
JMP SETCLK
.SBTTL BASIC DEFINITIONS
;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
ERROR =EMT
SCOPE = IOT
;*MISCELLANEOUS DEFINITIONS
HT= 11 ;;CODE FOR HORIZONTAL TAB
LF= 12 ;;CODE FOR LINE FEED
CR= 15 ;;CODE FOR CARRIAGE RETURN
CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776 ;;PROCESSOR STATUS WORD
PSW = PS
STKLMT= 177774 ;;STACK LIMIT REGISTER


```
177772 PIRQ= 177772 ::PROGRAM INTERRUPT REQUEST REGISTER
177570 DSWR= 177570 ::HARDWARE SWITCH REGISTER
177570 DDISP= 177570 ::HARDWARE DISPLAY REGISTER
:*GENERAL PURPOSE REGISTER DEFINITIONS
000000 R0= %0 ::GENERAL REGISTER
000001 R1= %1 ::GENERAL REGISTER
000002 R2= %2 ::GENERAL REGISTER
000003 R3= %3 ::GENERAL REGISTER
000004 R4= %4 ::GENERAL REGISTER
000005 R5= %5 ::GENERAL REGISTER
000006 R6= %6 ::GENERAL REGISTER
000007 R7= %7 ::GENERAL REGISTER
000006 SP= %6 ::STACK POINTER
000007 PC= %7 ::PROGRAM COUNTER
:*PRIORITY LEVEL DEFINITIONS
000000 PR0= 0 ::PRIORITY LEVEL 0
000040 PR1= 40 ::PRIORITY LEVEL 1
000100 PR2= 100 ::PRIORITY LEVEL 2
000140 PR3= 140 ::PRIORITY LEVEL 3
000200 PR4= 200 ::PRIORITY LEVEL 4
000240 PR5= 240 ::PRIORITY LEVEL 5
000300 PR6= 300 ::PRIORITY LEVEL 6
000340 PR7= 340 ::PRIORITY LEVEL 7
:*"SWITCH REGISTER" SWITCH DEFINITIONS
100000 SW15= 100000
040000 SW14= 40000
020000 SW13= 20000
010000 SW12= 10000
004000 SW11= 4000
002000 SW10= 2000
001000 SW09= 1000
000400 SW08= 400
000200 SW07= 200
000100 SW06= 100
000040 SW05= 40
000020 SW04= 20
000010 SW03= 10
000004 SW02= 4
000002 SW01= 2
000001 SW00= 1
001000 SW9 = SW09
000400 SW8 = SW08
000200 SW7 = SW07
000100 SW6 = SW06
000040 SW5 = SW05
000020 SW4 = SW04
000010 SW3 = SW03
000004 SW2 = SW02
000002 SW1 = SW01
000001 SW0 = SW00
:*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000 BIT15= 100000
040000 BIT14= 40000
020000 BIT13= 20000
010000 BIT12= 10000
004000 BIT11= 4000
002000 BIT10= 2000
```



```
001000 BIT09= 1000
000400 BIT08= 400
000200 BIT07= 200
000100 BIT06= 100
000040 BIT05= 40
000020 BIT04= 20
000010 BIT03= 10
000004 BIT02= 4
000002 BIT01= 2
000001 BIT00= 1
001000 BIT9 = BIT09
000400 BIT8 = BIT08
000200 BIT7 = BIT07
000100 BIT6 = BIT06
000040 BIT5 = BIT05
000020 BIT4 = BIT04
000010 BIT3 = BIT03
000004 BIT2 = BIT02
000002 BIT1 = BIT01
000001 BIT0 = BIT00
;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004 ERRVEC= 4 ;;TIME OUT AND OTHER ERRORS
000010 RESVEC= 10 ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014 TBITVEC=14 ;;'T' BIT
000014 TRTVEC= 14 ;;TRACE TRAP
000014 BPTVEC= 14 ;;BREAKPOINT TRAP (BPT)
000020 IOTVEC= 20 ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024 PWRVEC= 24 ;;POWER FAIL
000030 EMTVEC= 30 ;;EMULATOR TRAP (EMT) **ERROR**
000034 TRAPVEC=34 ;;'TRAP' TRAP
000060 TKVEC= 60 ;;TTY KEYBOARD VECTOR
000064 TPVEC= 64 ;;TTY PRINTER VECTOR
000240 PIRQVEC=240 ;;PROGRAM INTERRUPT REQUEST VECTOR
.SBTTL BIT DEFINITIONS
848 FBIT=BIT7 ;;FLAG BIT
849 EBIT=BIT6 ;;INTERRUPT ENABLE
850 MBIT=BIT5 ;;MAINTENANCE INTERRUPT
851 DBIT=BIT4 ;;I/O DISABLE BIT
852 TBIT=BIT3 ;;TEST BIT, INVERTS I/O BITS
853 GBIT=BIT2 ;;GENERIC CODE ENABLE
854 CBIT=BIT1 ;;CLEAR I/O
855 RBIT=BIT0 ;;RIF BIT, CLEARS I/O INTERRUPT
912
944
951 . =1100
952 * .SBTTL APT PARAMETER BLOCK
;*****
;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
001100 . $X=. ;;SAVE CURRENT LOCATION
000024 . =24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000044 200 ;;FOR APT START UP
000044 . =44 ;;POINT TO APT INDIRECT ADDRESS PNTR.
$APTHDR ;;POINT TO APT HEADER BLOCK
. =.$X ;;RESET LOCATION COUNTER
;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
```



```
001100  
001100 000000  
001102 001202  
001104 000120  
001106 000600  
001110 000600  
001112 000052
```

:INTERFACE SPEC.
\$APTHD:
\$HIBTS: .WORD 0 ::TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
\$MBADR: .WORD \$MAIL ::ADDRESS OF APT MAILBOX (BITS 0-15)
\$TSTM: .WORD 120 ::RUN TIM OF LONGEST TEST
\$PASTM: .WORD 600 ::RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
\$UNITM: .WORD 600 ::ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
\$ETEND-\$MAIL/2 ::LENGTH MAILBOX-ETABLE(WORDS)

1120

001114 001114
001114 000000
001116 000
001117 000
001120 000000
001122 000000
001124 000000
001126 000000
001130 000
001131 001
001132 000000
001134 000000
001136 000000
001140 000000
001142 000000
001144 000000
001146 000000
001150 000
001151 000
001152 000000
001154 177570
001156 177570
001160 177560
001162 177562
001164 177564
001166 177566
001170 000
001171 002
001172 012
001173 000
001174 000000
001176 077
001177 015
001200 012 000

```
.SBTTL COMMON TAGS
:*****
:*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
:*USED IN THE PROGRAM.
.=1114
$CMTAG:                ;;START OF COMMON TAGS
      .WORD           0
$STNM:  .BYTE         0      ;;CONTAINS THE TEST NUMBER
$ERFLG: .BYTE         0      ;;CONTAINS ERROR FLAG
$ICNT:  .WORD         0      ;;CONTAINS SUBTEST ITERATION COUNT
$LPADR: .WORD         0      ;;CONTAINS SCOPE LOOP ADDRESS
$LPERR: .WORD         0      ;;CONTAINS SCOPE RETURN FOR ERRORS
$ERTTL: .WORD         0      ;;CONTAINS TOTAL ERRORS DETECTED
$ITEMB: .BYTE         0      ;;CONTAINS ITEM CONTROL BYTE
$ERMAX: .BYTE         1      ;;CONTAINS MAX. ERRORS PER TEST
$ERRPC: .WORD         0      ;;CONTAINS PC OF LAST ERROR INSTRUCTION
$GDADR: .WORD         0      ;;CONTAINS ADDRESS OF 'GOOD' DATA
$BDADR: .WORD         0      ;;CONTAINS ADDRESS OF 'BAD' DATA
$GDDAT: .WORD         0      ;;CONTAINS 'GOOD' DATA
$BDDAT: .WORD         0      ;;CONTAINS 'BAD' DATA
      .WORD           0      ;;RESERVED--NOT TO BE USED
      .WORD           0
$AUTOB: .BYTE         0      ;;AUTOMATIC MODE INDICATOR
$INTAG: .BYTE         0      ;;INTERRUPT MODE INDICATOR
      .WORD           0
SWR:    .WORD         DSWR    ;;ADDRESS OF SWITCH REGISTER
DISPLAY: .WORD         DDISP  ;;ADDRESS OF DISPLAY REGISTER
$TKS:   177560          ;;TTY KBD STATUS
$TKB:   177562          ;;TTY KBD BUFFER
$TPS:   177564          ;;TTY PRINTER STATUS REG. ADDRESS
$TPB:   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
$NULL:  .BYTE         0      ;;CONTAINS NULL CHARACTER FOR FILLS
$FILLS: .BYTE         2      ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
$FILLC: .BYTE        12      ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
$TPFLG: .BYTE         0      ;;'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
$ESCAPE:0              ;;ESCAPE ON ERROR ADDRESS
$QUES:  .ASCII        /?/    ;;QUESTION MARK
$CRLF:  .ASCII        <15>   ;;CARRIAGE RETURN
$LF:    .ASCII        <12>   ;;LINE FEED
:*****
```

```
.SBTTL APT MAILBOX-ETABLE
:*****
```

001202
001202 000000
001204 000000
001206 000000
001210 000000
001212 000000
001214 000000
001216 000000
001220 000000
001222
001222 000
001223 000
001224 000000
001226 000000

```
.EVEN
$MAIL:                ;;APT MAILBOX
$MSGTY: .WORD         AMSGTY  ;;MESSAGE TYPE CODE
$FATAL: .WORD         AFATAL  ;;FATAL ERROR NUMBER
$TESTN: .WORD         ATESTN  ;;TEST NUMBER
$PASS:  .WORD         APASS   ;;PASS COUNT
$DEVCT: .WORD         ADEVCT  ;;DEVICE COUNT
$UNIT:  .WORD         AUNIT   ;;I/O UNIT NUMBER
$MSGAD: .WORD         AMSGAD  ;;MESSAGE ADDRESS
$MSGLG: .WORD         AMSGLG  ;;MESSAGE LENGTH
$ETABLE:                ;;APT ENVIRONMENT TABLE
$ENV:   .BYTE         AENV    ;;ENVIRONMENT BYTE
$ENVM:  .BYTE         AENVM   ;;ENVIRONMENT MODE BITS
$SWREG: .WORD         ASWREG  ;;APT SWITCH REGISTER
$USWR:  .WORD         AUSWR   ;;USER SWITCHES
```



```
001230 000000 $CPUOP: .WORD ACPUOP ;;CPU TYPE,OPTIONS
:*
:* BITS 15-11=CPU TYPE
:* 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
:* 11/70=06,PDQ=07,Q=10
:* BIT 10=REAL TIME CLOCK
:* BIT 9=FLOATING POINT PROCESSOR
:* BIT 8=MEMORY MANAGEMENT
001232 000 $MAMS1: .BYTE AMAMS1 ;;HIGH ADDRESS,M.S. BYTE
001233 000 $MTYP1: .BYTE AMTYP1 ;;MEM. TYPE,BLK#1
:*
:* MEM.TYPE BYTE -- (HIGH BYTE)
:* 900 NSEC CORE=001
:* 300 NSEC BIPOLAR=002
:* 500 NSEC MOS=003
001234 000000 $MADR1: .WORD AMADR1 ;;HIGH ADDRESS,BLK#1
:*
:* MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE
001236 000 $MAMS2: .BYTE AMAMS2 ;;HIGH ADDRESS,M.S. BYTE
001237 000 $MTYP2: .BYTE AMTYP2 ;;MEM. TYPE,BLK#2
001240 000000 $MADR2: .WORD AMADR2 ;;MEM.LAST ADDRESS,BLK#2
001242 000 $MAMS3: .BYTE AMAMS3 ;;HIGH ADDRESS,M.S.BYTE
001243 000 $MTYP3: .BYTE AMTYP3 ;;MEM. TYPE,BLK#3
001244 000000 $MADR3: .WORD AMADR3 ;;MEM.LAST ADDRESS,BLK#3
001246 000 $MAMS4: .BYTE AMAMS4 ;;HIGH ADDRESS,M.S.BYTE
001247 000 $MTYP4: .BYTE AMTYP4 ;;MEM. TYPE,BLK#4
001250 000000 $MADR4: .WORD AMADR4 ;;MEM.LAST ADDRESS,BLK#4
001252 000000 $VECT1: .WORD AVECT1 ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001254 000000 $VECT2: .WORD AVECT2 ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001256 000000 $BASE: .WORD ABASE ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001260 000000 $DEVN: .WORD ADEVN ;;DEVICE MAP
001262 000000 $CDW1: .WORD ACDW1 ;;CONTROLLER DESCRIPTION WORD#1
001264 000000 $CDW2: .WORD ACDW2 ;;CONTROLLER DESCRIPTION WORD#2
001266 000000 $DDW0: .WORD ADDW0 ;;DEVICE DESCRIPTOR WORD#0
001270 000000 $DDW1: .WORD ADDW1 ;;DEVICE DESCRIPTOR WORD#1
001272 000000 $DDW2: .WORD ADDW2 ;;DEVICE DESCRIPTOR WORD#2
001274 000000 $DDW3: .WORD ADDW3 ;;DEVICE DESCRIPTOR WORD#3
001276 000000 $DDW4: .WORD ADDW4 ;;DEVICE DESCRIPTOR WORD#4
001300 000000 $DDW5: .WORD ADDW5 ;;DEVICE DESCRIPTOR WORD#5
001302 000000 $DDW6: .WORD ADDW6 ;;DEVICE DESCRIPTOR WORD#6
001304 000000 $DDW7: .WORD ADDW7 ;;DEVICE DESCRIPTOR WORD#7
001306 000000 $DDW8: .WORD ADDW8 ;;DEVICE DESCRIPTOR WORD#8
001310 000000 $DDW9: .WORD ADDW9 ;;DEVICE DESCRIPTOR WORD#9
001312 000000 $DDW10: .WORD ADDW10 ;;DEVICE DESCRIPTOR WORD#10
001314 000000 $DDW11: .WORD ADDW11 ;;DEVICE DESCRIPTOR WORD#11
001316 000000 $DDW12: .WORD ADDW12 ;;DEVICE DESCRIPTOR WORD#12
001320 000000 $DDW13: .WORD ADDW13 ;;DEVICE DESCRIPTOR WORD#13
001322 000000 $DDW14: .WORD ADDW14 ;;DEVICE DESCRIPTOR WORD#14
001324 000000 $DDW15: .WORD ADDW15 ;;DEVICE DESCRIPTOR WORD#15
001326 $ETEND:
001326 ATABL: .BLKW 80. ;TABLE A=ADDRESSES OF I/O MODULES
001566 BTABL: .BLKW 80. ;TABLE B=GENERIC CODES OF THIS MODULES
002026 000141 GENER: .WORD 141 ;GENERIC CODE FOR NONISOLATED DC 32 BITS IN
002030 000121 .WORD 121 ;GENERIC CODE FOR NONISOLATED DC 16 BITS IN
002032 000122 .WORD 122 ;GENERIC CODE FOR ISOLATED DC 16 BITS IN
002034 000101 .WORD 101 ;GENERIC CODE FOR AC 8 BITS IN
002036 000041 .WORD 41 ;GENERIC CODE FOR NONISOLATED DC 32 BITS OUT
002040 000021 .WORD 21 ;GENERIC CODE FOR ONESHOT 16 BITS OUT
002042 000001 .WORD 1 ;GENERIC CODE FOR ISOLATED DC 8 BITS OUT
002044 000002 .WORD 2 ;GENERIC CODE FOR AC 8 BITS OUT
```


002046	000123		.WORD	123		:16 BIT TTL COMPATABLE ISOLATED INPUT
002050	000043		.WORD	43		:32 BIT TTL COMPATABLE NONISOLATED OUTPUT
002052	000261		.WORD	261		:GENERIC CODE FOR FOUR CHANEL DAC
002054	000321		.WORD	321		:A/D SINGLE ENDED
002056	000301		.WORD	301		:A/D DIFFERENTIAL MODE
002060	000000		.WORD	0		
002062	005010	MUT:	.WORD	5010		
002064	005011		.WORD	5011		
002066	005012		.WORD	5012		
002070	005013		.WORD	5013		
002072	006010		.WORD	6010		
002074	006011		.WORD	6011		
002076	006012		.WORD	6012		
002100	006013		.WORD	6013		
002102	005012		.WORD	5012		:M5012-YA
002104	006010		.WORD	6010		:M6010-YA
002106	000630		.WORD	630		
002110	000014		.WORD	14		
002112	000014		.WORD	14		
002114	010570	MODUL:	.WORD	M5010		:THIS TABLE HAS ADDRESSES OF I/O SUBROUTINE
002116	012216		.WORD	M5011		
002120	013046		.WORD	M5012		
002122	010672		.WORD	M5013		
002124	011334		.WORD	M6010		
002126	011720		.WORD	M6011		
002130	011526		.WORD	M6012		
002132	011526		.WORD	M6013		
002134	013046		.WORD	M5012		:M5012-YA SAME A M5012
002136	011334		.WORD	M6010		:M6010-YA SAME AS M6010
002140	013524		.WORD	DCAMON		
002142	017064		.WORD	ADTST		
002144	017064		.WORD	ADTST		
002146	000000		.WORD	0		
002150	000000	YLOOP:	.WORD	0		
002152	000000	MXNUM:	.WORD	0		
002154	000000	DBUFF:	.WORD	0		
002156	000000		.WORD	0		
002160	125	PATT:	.BYTE	125		
002161	252		.BYTE	252		
002162	377		.BYTE	377		
002163	000		.BYTE	0		
002164	000042	XXDP:	.WORD	42		
002166	000000	\$MUT:	.WORD	0		
002170	000000	BYTNUM:	.WORD	0		:NUMBER OF BITES OF I/O
002172	000000	TADDR:	.WORD	0		:ADDRESS OF MUT
002174	000000	TADDR1:	.WORD	0		
002176	000000	TBADDR:	.WORD	0		
002200	000000	COSADR:	.WORD	0		:SECOND ADDRESS OF COS MODULE
002202	000000	DMUT:	.WORD	0		:DIGITAL MODULE UNDER TEST
002204	000000	LONGIN:	.WORD	0		
002206	177546	CLKADR:	.WORD	177546		
002210	171377	CSR:	.WORD	171377		
002212	171376	IAR:	.WORD	171376		
002214	000234	VECTO:	.WORD	234		
002216	000236	VECTOA:	.WORD	236		
002220	000000	VECT1:	.WORD	0		
002222	000000	VECT1A:	.WORD	0		

002224	171000	BASE:	.WORD	171000	;FIRST I/O
002226	000000	CLK:	.WORD	0	
002230	000000	NORAMP:	.WORD	0	
002232	000000	TEMP:	.WORD	0	
002234	000000	RERROR:	.WORD	0	
002236	000000	XXX:	.WORD	0	
002240	000000	AFLAG:	.WORD	0	
002242	000000	TEMP1:	.WORD	0	
002244	000000	INTDAT:	.WORD	0	
002246	000001	PASCNT:	.WORD	1	
002250	000100	CLKVC:	.WORD	100	
002252	000102	CLKVCA:	.WORD	102	
002254	016756	MOD:	.WORD	F5010	
002256	016756		.WORD	F5011	
002260	016770		.WORD	F5012	
002262	017002		.WORD	F5013	
002264	017014		.WORD	F6010	
002266	017050		.WORD	F6011	
002270	017034		.WORD	F6012	
002272	017034		.WORD	F6013	
002274	016770		.WORD	F5012	;M5012-YA
002276	017014		.WORD	F6010	;M6010-YA
002300	000000		.WORD	0	
002302	000060	KBVEC:	.WORD	60	
002304	000000	CHNUM:	.WORD	0	
002306	000000	CH0:	.WORD	0	
002310	000000	CH1:	.WORD	0	
002312	000000	CH2:	.WORD	0	
002314	000000	CH3:	.WORD	0	
002316	000000	CH4:	.WORD	0	
002320	000000	DCHAN:	.WORD	0	
002322	000000	XCHAN:	.WORD	0	
002324	000000	VCHAN:	.WORD	0	
002326	000000	ICHAN:	.WORD	0	
002330	171000	LDATA0:	.WORD	171000	
002332	171001	HDATA0:	.WORD	171001	
002334	171002	LDATA1:	.WORD	171002	
002336	171003	HDATA1:	.WORD	171003	
002340	171004	LDATA2:	.WORD	171004	
002342	171005	HDATA2:	.WORD	171005	
002344	171006	LDATA3:	.WORD	171006	
002346	171007	HDATA3:	.WORD	171007	
002350	000000	KOUNT:	.WORD	0	
002352	000000	ACOUNT:	.WORD	0	
002354	000000	BCOUNT:	.WORD	0	
002356	000000	CCOUNT:	.WORD	0	
002360	000000	ANSW:	.WORD	0	
002362	000000	DCOUNT:	.WORD	0	
002364	000000	CONV:	.WORD	0	
002366	000000	DATALO:	.WORD	0	
002370	000000	CONVCT:	.WORD	0	
002372	000000	ECOUNT:	.WORD	0	
002374		RAMP1:	.BLKW	40.	
002514		RAMP2:	.BLKW	40.	
002634		RAMP3:	.BLKW	40.	
002754	000000	BLAST:	.WORD	0	

002756 000000
002760 000000
002762 000000
002764 177777
002766 000000
002770 000000
002772 000000
002774 000000
002776 000000
003000 000000
003002 000000
003004 000000
003006 000000
003010 000001
003012 000002
003014 000010
003016 000020
003020 000050
003022 000100
003024 000200
003026 001000
003030 000000
003032 000000
003034 000020
003036 000100
003040 000120
003042 000200
003044 000220
003046 000300
003050 000320
003052 000000
003054 000000
003056 000000
003060 000000
003062 000000
003064 000
003067 002
003072 003

001 001
002 003
004

LAST: .WORD 0
LASTCN: .WORD 0
ROTPAT: .WORD 0
ROTFLG: .WORD -1
TEMP2: 0
TEMP3: 0
TEMP4: 0
TEMP5: 0
GBITE: 0
STAT1: 0
STAT2: 0
LBYTE: 0
HBYTE: 0
GAINTB: .WORD 1 ;TABLE OF A157 GAINS
.WORD 2
.WORD 10
.WORD 20
.WORD 50
.WORD 100
.WORD 200
.WORD 1000
GAIN: .WORD 0 ;CORRESPONDING OCTAL VALUES TO BE
.WORD 20 ;LOADED INTO THE GAIN REGISTER OF A157
.WORD 100
.WORD 120
.WORD 200
.WORD 220
.WORD 300
.WORD 320
ST1SAV: .WORD 0 ;STAT1 SAVE
ST2SAV: .WORD 0 ;STAT2 SAVE
RETURN: 0
AVRSAB: 0
AVRTBL: .BYTE 0,1,1,2,2,3,3,4

.SBTTL ERROR POINTER TABLE
:*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
:*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
:*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
:*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
:*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
:* EM ;;POINTS TO THE ERROR MESSAGE
:* DH ;;POINTS TO THE DATA HEADER
:* DT ;;POINTS TO THE DATA
:* DF ;;POINTS TO THE DATA FORMAT
\$ERRTB:

1121	003074	032074	EM1
1122	003076	032204	DH1
1123	003100	044454	DT1
1124	003102	000000	0
1125	003104	032123	EM2
1126	003106	032204	DH1
1127	003110	044454	DT1
1128	003112	000000	0
1129	003114	032235	EM3
1130	003116	000000	0
1131	003120	000000	0
1132	003122	000000	0
1133	003124	032266	EM4
1134	003126	032204	DH1
1135	003130	044454	DT1
1136	003132	000000	0
1137	003134	032314	EM5
1138	003136	032334	DH5
1139	003140	044470	DT5
1140	003142	000000	0
1141	003144	032367	EM6
1142	003146	032204	DH1
1143	003150	044454	DT1
1144	003152	000000	0
1145	003154	032436	EM7
1146	003156	032453	DH7
1147	003160	044434	DT7
1148	003162	000000	0
1149	003164	032523	EM10
1150	003166	000000	0
1151	003170	000000	0
1152	003172	000000	0
1153	003174	032556	EM11
1154	003176	032453	DH7
1155	003200	044434	DT7
1156	003202	000000	0
1157	003204	032614	EM12
1158	003206	032334	DH5
1159	003210	044470	DT5
1160	003212	000000	0
1161	003214	032644	EM13
1162	003216	000000	0
1163	003220	000000	0
1164	003222	000000	0
1165	003224	032712	EM14
1166	003226	000000	0

1167	003230	000000	0
1168	003232	000000	0
1169	003234	032755	EM15
1170	003236	000000	0
1171	003240	000000	0
1172	003242	000000	0
1173	003244	033020	EM16
1174	003246	032204	DH1
1175	003250	044454	DT1
1176	003252	000000	0
1177	003254	032436	EM17
1178	003256	032453	DH7
1179	003260	044364	DT17
1180	003262	000000	0
1181	003264	033063	EM20
1182	003266	032453	DH7
1183	003270	044404	DT20
1184	003272	000000	0
1185	003274	033115	EM21
1186	003276	033157	DH21
1187	003300	044504	DT21
1188	003302	000000	0
1189	003304	033203	EM22
1190	003306	000000	0
1191	003310	000000	0
1192	003312	000000	0
1193	003314	033257	EM23
1194	003316	033314	DH23
1195	003320	044424	DT23
1196	003322	000000	0
1197	003324	033407	EM24
1198	003326	000000	0
1199	003330	000000	0
1200	003332	000000	0
1201	003334	034273	EM25
1202	003336	000000	0
1203	003340	000000	0
1204	003342	000000	0
1205	003344	040625	EM26
1206	003346	032334	DH5
1207	003350	044470	DT5
1208	003352	000000	0
1209			
1210	003354	040561	EM27
1211	003356	032334	DH5
1212	003360	044470	DT5
1213	003362	000000	0
1214			
1215	003364	041175	EM30
1216	003366	041237	DH30
1217	003370	044516	DT30
1218	003372	000000	0
1219			
1220	003374	044202	EM31
1221	003376	044332	DH31
1222	003400	044674	DT31
1223	003402	000000	0

1224			
1225	003404	044255	EM32
1226	003406	044332	DH31
1227	003410	044674	DT31
1228	003412	000000	0
1229			
1230	003414	040701	EM33
1231	003416	040735	DH33
1232	003420	044532	DT33
1233	003422	000000	0
1234			
1235	003424	041004	EM34
1236	003426	041071	DH34
1237	003430	044554	DT34
1238	003432	000000	0
1239			
1240	003434	041304	EM35
1241	003436	032453	DH7
1242	003440	044434	DT7
1243	003442	000000	0
1244			
1245	003444	041355	EM36
1246	003446	032334	DH5
1247	003450	044470	DT5
1248	003452	000000	0
1249			
1250	003454	041423	EM37
1251	003456	032334	DH5
1252	003460	044470	DT5
1253	003462	000000	0
1254			
1255	003464	041464	EM40
1256	003466	032334	DH5
1257	003470	044470	DT5
1258	003472	000000	0
1259			
1260	003474	041527	EM41
1261	003476	032334	DH5
1262	003500	044470	DT5
1263	003502	000000	0
1264			
1265	003504	041572	EM42
1266	003506	032453	DH7
1267	003510	044434	DT7
1268	003512	000000	0
1269			
1270	003514	041633	EM43
1271	003516	032334	DH5
1272	003520	044470	DT5
1273	003522	000000	0
1274			
1275	003524	041710	EM44
1276	003526	032334	DH5
1277	003530	044470	DT5
1278	003532	000000	0
1279			
1280	003534	041747	EM45

1281	003536	032334	DH5
1282	003540	044470	DT5
1283	003542	000000	0
1284			
1285	003544	042014	EM46
1286	003546	032453	DH7
1287	003550	044434	DT7
1288	003552	000000	0
1289			
1290	003554	042066	EM47
1291	003556	032334	DH5
1292	003560	044470	DT5
1293	003562	000000	0
1294			
1295	003564	042114	EM50
1296	003566	032453	DH7
1297	003570	044434	DT7
1298	003572	000000	0
1299			
1300	003574	000000	0
1301	003576	000000	0
1302	003600	044576	DT51
1303	003602	044706	DF51
1304			
1305	003604	000000	0
1306	003606	000000	0
1307	003610	044612	DT52
1308	003612	044712	DF52
1309			
1310	003614	034565	EM53
1311	003616	000000	0
1312	003620	000000	0
1313	003622	000000	0
1314			
1315	003624	034523	EM54
1316	003626	000000	0
1317	003630	000000	0
1318	003632	000000	0
1319			
1320	003634	035030	EM55
1321	003636	032334	DH5
1322	003640	044470	DT5
1323	003642	000000	0
1324			
1325	003644	035103	EM56
1326	003646	032334	DH5
1327	003650	044470	DT5
1328	003652	000000	0
1329			
1330			
1331	003654	035206	EM57
1332	003656	035235	DH57
1333	003660	044630	DT57
1334	003662	000000	0
1335			
1336	003664	035261	EM60
1337	003666	035421	DH61

1338	003670	044640	DT61
1339	003672	000000	0
1340			
1341	003674	035315	EM61
1342	003676	041147	DH40
1343	003700	044424	DT23
1344	003702	000000	0
1345			
1346	003704	041633	EM43
1347	003706	035474	DH62
1348	003710	044660	DT62
1349	003712	000000	0
1350			
1351	003714	041572	EM42
1352	003716	035421	DH61
1353	003720	044640	DT61
1354	003722	000000	0
1355			
1356	003724	041175	EM30
1357	003726	035421	DH61
1358	003730	044640	DT61
1359	003732	000000	0
1360			
1361	003734	035364	EM63
1362	003736	035474	DH62
1363	003740	044660	DT62
1364	003742	000000	0
1365			
1366	003744	041747	EM45
1367	003746	035474	DH62
1368	003750	044660	DT62
1369	003752	000000	0
1370			
1371	003754	035533	EM62
1372	003756	035474	DH62
1373	003760	044660	DT62
1374	003762	000000	0
1375			
1376	003764	041464	EM40
1377	003766	035474	DH62
1378	003770	044660	DT62
1379	003772	000000	0
1380			
1381	003774	041423	EM37
1382	003776	035474	DH62
1383	004000	044660	DT62
1384	004002	000000	0
1385			
1386	004004	042114	EM50
1387	004006	035421	DH61
1388	004010	044640	DT61
1389	004012	000000	0
1390			
1391	004014	035574	EM73
1392	004016	035474	DH62
1393	004020	044660	DT62
1394	004022	000000	0

1396

1399 004024

```
START:
.SBTTL INITIALIZE THE COMMON TAGS
::CLEAR THE COMMON TAGS ($CMTAG) AREA
004024 012706 001114      MOV    # $CMTAG,R6      ::FIRST LOCATION TO BE CLEARED
004030 005026             CLR    (R6)+           ::CLEAR MEMORY LOCATION
004032 022706 001154      CMP    #SWR,R6        ::DONE?
004036 001374             BNE    -6              ::LOOP BACK IF NO
004040 012706 001100      MOV    #STACK,SP      ::SETUP THE STACK POINTER
::INITIALIZE A FEW VECTORS
004044 012737 027156 000020  MOV    # $SCOPE,@#IOTVEC ::IOT VECTOR FOR SCOPE ROUTINE
004052 012737 000340 000022  MOV    #340,@#IOTVEC+2 ::LEVEL 7
004060 012737 030016 000030  MOV    # $ERROR,@#EMTVEC ::EMT VECTOR FOR ERROR ROUTINE
004066 012737 000340 000032  MOV    #340,@#EMTVEC+2 ::LEVEL 7
004074 012737 031634 000034  MOV    # $TRAP,@#TRAPVEC ::TRAP VECTOR FOR TRAP CALLS
004102 012737 000340 000036  MOV    #340,@#TRAPVEC+2::LEVEL 7
004110 013737 006332 006324  MOV    $ENDCT,$EOPCT   ::SETUP END-OF-PROGRAM COUNTER
004116 005037 001174             CLR    $ESCAPE        ::CLEAR THE ESCAPE ON ERROR ADDRESS
004122 112737 000001 001131  MOVB   #1,$ERMAX      ::ALLOW ONE ERROR PER TEST
004130 012737 004130 001124  MOV    #,$LPERR       ::SETUP THE ERROR LOOP ADDRESS
::SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
::EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
004136 013746 000004             MOV    @#ERRVEC,-(SP)  ::SAVE ERROR VECTOR
004142 012737 004176 000004  MOV    #64$,@#ERRVEC  ::SET UP ERROR VECTOR
004150 012737 177570 001154  MOV    #DSWR,SWR      ::SETUP FOR A HARDWARE SWICH REGISTER
004156 012737 177570 001156  MOV    #DDISP,DISPLAY ::AND A HARDWARE DISPLAY REGISTER
004164 022777 177777 174762  CMP    #-1,@SWR      ::TRY TO REFERENCE HARDWARE SWR
004172 001012             BNE    66$           ::BRANCH IF NO TIMEOUT TRAP OCCURRED
::AND THE HARDWARE SWR IS NOT = -1
004174 000403             BR     65$           ::BRANCH IF NO TIMEOUT
004176 012716 004204 64$:    MOV    #65$,(SP)     ::SET UP FOR TRAP RETURN
004202 000002             RTI
004204 012737 000176 001154 65$:    MOV    #SWREG,SWR    ::POINT TO SOFTWARE SWR
004212 012737 000174 001156  MOV    #DISPREG,DISPLAY
004220 012637 000004 66$:    MOV    (SP)+,@#ERRVEC ::RESTORE ERROR VECTOR
004224 005037 001210             CLR    $PASS         ::CLEAR PASS COUNT
004230 132737 000200 001223  BITB   #APTSIZE,$ENVM ::TEST USER SIZE UNDER APT
004236 001403             BEQ    67$           ::YES,USE NON-APT SWITCH
004240 012737 001224 001154  MOV    # $SWREG,SWR   ::NO,USE APT SWITCH REGISTER
004246 67$:
```



```
1401 .SBTTL DIAGNOSTICS MONITOR
1402 .SBTTL TYPE PROGRAM NAME
      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
      INC #-1 ;;FIRST TIME?
      BNE 68$ ;;BRANCH IF NO
      CMP #SENDAD,@#42 ;;ACT-11?
      BEQ 68$ ;;BRANCH IF YES
      TYPE ,69$ ;;TYPE ASCIZ STRING
      BR 68$ ;;GET OVER THE ASCIZ
      ;;69$: .ASCIZ <CRLF>/DIGITAL IO SYSTEM TEST VERSION 1/<CRLF>
      68$:
1403 004336 012737 004354 000004 SETCLK: MOV #1$,ERRVEC
1404 004344 052777 000100 175634 BIS #BIT6,@CLKADR ;ENABLE LINE CLOCK INTERRUPT FOR UNIBUS COMPUTERS
1405 004352 000401 BR 2$
1406 004354 022626 1$: CMP (SP)+,(SP)+ ;TRAP IF LSI11
1407 004356 012737 005044 000004 2$: MOV #TMOVEC,ERRVEC ;RESTORE ERROR VECTOR
1408 004364 012706 001100 MOV #1100,R6
1409 004370 005777 175570 TST @XXDP ;ARE WE IN XXDP CHAIN MODE?
1410 004374 001404 BEQ 6$ ;NO
1411 004376 005037 001210 CLR $PASS
1412 004402 000137 @05442 JMF AUTO
1413
1414 004406 132737 000001 001222 6$: BITB #BIT0,$ENV ;ARE WE UNDER APT
1415 004414 001413 BEQ 4$ ;NO
1416 004416 132737 000040 001223 BITB #BIT5,$ENV+1 ;INHIBIT PRINT?
1417 004424 001403 BEQ 3$
1418 004426 052737 020000 000176 BIS #BIT13,SWREG
1419 004434 005037 001210 3$: CLR $PASS
1420 004440 000137 005442 JMP AUTO
1421 004444 4$:
      004444 012777 025556 175630 MOV #KBINT,@KBVEC
      004452 152777 000100 174500 BISB #BIT6,@$TKS ;ENABLE KB INTERRUPT
1422 004460 012746 000000 MOV #PRO,-(SP) ;SET PSW TO PRIORITY 0
      004464 012746 004472 MOV #64$,-(SP)
      004470 000002 RTI
      004472 000240 64$: NOP
1423 004474 000424 BR MONIT
1424 004476 104401 033455 TYPOPT: TYPE ,M13 ;TEST OPTIONS
1425 004502 104401 033477 TYPE ,M14 ;S - SYSTEM TEST
1426 004506 104401 033520 TYPE ,M15 ;D - DIGITAL MODULE TEST
1427 004512 104401 033567 TYPE ,M17 ;M - MAP OF DBUS DEVICES
1428 004516 104401 033642 TYPE ,M19 ;I - IOCM TEST
1429 004522 104401 033670 TYPE ,M21 ;T - SET SWREG
1430 004526 104401 034076 TYPE ,M24 ;F - FIELD TEST
1431 004532 104401 034116 TYPE ,M25 ;W - WRAP AROUND TEST
1432 004536 104401 034135 TYPE ,M26 ;L - LOAD LOOP COUNT
1433 004542 104401 034224 TYPE ,M28 ;A - ANALOG MODULE TEST
1434
```


1493 005044 104401 040310
1494 005050 011646
005052 104402
1495 005054 000000

TMOVEC: TYPE ,MASS45
MOV (R6),-(SP)
TYPOC
HALT

:UNEXPECTED TIMEOUT ERROR
::SAVE (R6) FOR TYPEOUT
::GO TYPE--OCTAL ASCII(ALL DIGITS)


```
1498 .SBTTL DIGITAL MODULE MONITOR
1499 DIGIT:
005056 012777 025556 175216 MOV #KBINT,@KBVEC
005064 152777 000100 174066 BISB #BIT6,@$TKS ;ENABLE KB INTERRUPT
1500 005072 004737 025502 JSR PC,CNTRC
1501 005076 104401 036415 TYPE ,MASS13 ;TYPE ADDRESS OF MUT
1502 005102 104410 RDOCT
1503 005104 012637 002172 MOV (SP)+,TADDR ;;POP STACK INTO TADDR
1504 005110 023737 002172 002224 CMP TADDR,BASE
1505 005116 103757 BLO DIGIT
1506 005120 013746 000004 MOV ERRVEC,-(SP) ;SAVE ERROR VECTOR
1507 005124 012737 005400 000004 MOV #5$,ERRVEC ;SET LOC 4
1508 005132 004737 016652 JSR PC,KLEER ;CLEAR THE IOCM
1509 005136 105777 175046 TSTB @CSR ;MAKE SURE IOCM IS OK
1510 005142 001404 BEQ 17$
1511 005144 104010 ERROR!10 ;TEST ABORTED,IOCM ERROR
1512 005146 012637 000004 MOV (SP)+,ERRVEC ;;POP STACK INTO ERRVEC
1513 005152 000207 RTS PC
1514 005154 112777 000004 175026 17$: MOVB #GBIT,@CSR ;SET GENERIC BIT
1515 005162 117700 175004 MOVB @TADDR,R0 ;R0=GENERIC CODE OF MUT
1516 005166 042700 177400 BIC #177400,R0
1517 005172 012701 002114 MOV #MODUL,R1 ;ADDRESS OF TEST
1518 005176 012703 002062 MOV #MUT,R3
1519 005202 012702 002026 MOV #GENER,R2 ;TABLE OF GENERIC CODES
1520 005206 020022 3$: CMP R0,(R2)+
1521 005210 001050 BNE 1$
1522 005212 011337 002166 MOV (R3),SMUT
1523 005216 004737 016652 JSR PC,KLEER
1524 005222 011137 002202 MOV (R1),DMUT
1525 005226 005037 001210 9$: CLR $PASS
1526 005232 004777 174744 4$: JSR PC,ADMUT ;TEST MODULE
1527 005236 004737 025502 JSR PC,CNTRC ;CONTROL C?
1528 005242 005737 002246 TST PASCNT ;IF ZERO LOOP ON TEST
1529 005246 001771 BEQ 4$
1530 005250 005237 001210 INC $PASS
1531 005254 023737 002246 001210 CMP PASCNT,$PASS
1532 005262 001363 BNE 4$
1533 005264 032737 020000 000176 BIT #BIT13,SWREG ;INHIBIT PRINT ?
1534 005272 001005 BNE 7$
1535 005274 013746 002246 MOV PASCNT,-(SP) ;;SAVE PASCNT FOR TYPEOUT
005300 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1536 005302 104401 033363 TYPE ,M11 ;# OF PASSES
1537 005306 032737 040000 000176 7$: BIT #BIT14,SWREG ;LOOP ?
1538 005314 001344 BNE 9$
1539 005316 104401 036474 TYPE ,MASS15 ;END OF MODULE TEST
1540 005322 004737 016652 JSR PC,KLEER
1541 005326 000137 004546 JMP MONIT
1542 005332 062701 000002 1$: ADD #2,R1
1543 005336 062703 000002 ADD #2,R3
1544 005342 021227 000261 CMP (R2),#261 ;END OF TABLE FOR 'DIGITAL' TESTS ?
1545 005346 001317 BNE 3$
1546 005350 104401 036307 TYPE ,MASS11 ;UNKNOWN GENERIC CODE
1547 005354 010046 MOV R0,-(SP) ;;SAVE R0 FOR TYPEOUT
005356 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1548 005360 104401 035646 TYPE ,ASSO ;CR,LF
1549 005364 012637 000004 MOV (SP)+,ERRVEC ;RESTORE ERRVEC
1550 005370 004737 016652 JSR PC,KLEER
```


1551 005374 000137 004546
1552 005400 022626
1553 005402 104024
1554 005404 012637 000004
1555 005410 000137 004546

5\$: JMP MONIT
CMP (SP)+,(SP)+ ;RESTORE STACK POINTER
ERROR!24 ;MODULE NOT RESPONDING
MOV (SP)+,ERRVEC
JMP MONIT

```

1557
1558
1559 005414 104401 033707
1560 005420 104401 034064
1561 005424 104410
1562 005426 012637 000176
1563 005432 104401 035646
1564 005436 000137 004546
1565
1566
1567
1568
1569

```

.SBTTL SET SOFTWARE SWITCH REGISTER

```

SWREGS: TYPE      ,M22           ;SWICHES OPTIONS
          TYPE      ,M23           ;SW REG =
          RDOCT
          MOV      (SP)+,SWREG     ;:POP STACK INTO SWREG
          TYPE      ,MASS0        ;:CR,LF
          JMP      MONIT

```


1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582 005442
1583 005442 012777 025556 174632
005450 152777 000100 173502
1584 005456 004737 007542
1585 005462 004737 016652
1586 005466 105777 174516
1587 005472 001402
1588 005474 104010
1589 005476 000207
1590 005500 013746 000004
1591 005504 012737 005662 000004
1592 005512 013700 002224
1593 005516 112777 000004 174464
1594 005524 012702 001326
1595 005530 012705 001566
1596 005534 111001
1597 005536 042701 177400
1598 005542 004737 025250
1599 005546 000164 005552
1600 005552 000417
1601 005554 000416
1602 005556 000417
1603 005560 000417
1604 005562 000413
1605 005564 000414
1606 005566 000414
1607 005570 000413
1608 005572 000411
1609 005574 000406
1610 005576 000403
1611 005600 000404
1612 005602 000403
1613 005604 000405
1614 005606 062700 000004
1615 005612 005200
1616 005614 005200
1617 005616 005200
1618 005620 005200
1619 005622 020037 002212
1620 005626 103742
1621 005630 005012
1622 005632 022702 001326
1623 005636 001017
1624 005640 012637 000004
1625 005644 142777 000004 174336
1626 005652 104401 040531
1627 005656 000137 006146

.SBTTL AUTO TEST MONITOR

::*****
:THIS PART OF A PROGRAM IS A MONITOR FOR AUTOMATIC TEST.
:FIRST IT TEST IOCM. THEN IT GENERATES A TABLE AND B TABLE.
:THEN IT TESTS EACH INDIVIDUAL I/O MODULE CONNECTED TO DBUS.
:REGISTERS R0 & R1 SERVE AS THE POINTERS FOR MUT

::*****

AUTO:

MOV #KBINT,@KBVEC
BISB #BIT6,@STKS ;ENABLE KB INTERRUPT
JSR PC,TIOCM
JSR PC,KLEER ;CLEAR THE IOCM
TSTB @CSR ;MAKE SURE IOCM IS OK
BEQ 7\$
ERROR!10 ;TEST ABORTED
RTS PC
7\$: MOV @ERRVEC,-(SP) ;SAVE ERROR VECTOR
MOV #1\$,ERRVEC ;SET ERROR VECTOR
MOV BASE,R0 ;R0=171000
MOVB #GBIT,@CSR ;SET GENERIC CODE ENABLE
MOV #ATABL,R2
MOV #BTABL,R5
5\$: MOVB (R0),R1 ;R1=GENERIC CODE,READ GENERIC CODE
BIC #177400,R1
JSR PC,TABLE ;FIND WHAT I/O IT IS
JMP 3\$(R4)
3\$: BR 10\$
BR 10\$
BR 12\$
BR 13\$
BR 10\$
BR 12\$
BR 13\$
BR 13\$
BR 12\$;M5012-YA
BR 10\$;M6010-YA
BR 9\$
BR 10\$
BR 10\$
BR 13\$
9\$: ADD #4,R0
10\$: INC R0
11\$: INC R0
12\$: INC R0
13\$: INC R0
CMP R0,IAR
BLO 5\$
CLR (R2)
CMP #ATABL,R2
BNE 6\$
MOV (SP)+,ERRVEC ;:POP STACK INTO ERRVEC
BICB #CBIT,@CSR
TYPE MASS50 ;NO I/O
JMP 26\$

```
1628 005662 022626          1$:  CMP      (SP)+,(SP)+
1629 005664 005200          INC      R0
1630 005666 020037 002212      CMP      R0,IAR
1631 005672 103720          BLO      5$
1632 005674 005015          CLR      (R5)
1633 005676 012637 000004      6$:  MOV      (SP)+,@#ERRVEC ;START TESTING INDIVIDUAL I/O
1634 005702 142777 000004 174300 BICB     #GBIT,@CSR ;CLEAR G BIT
1635 005710 004737 007542      25$:  JSR      PC,TIOCM
1636 005714 005000          CLR      R0
1637 005716 005001          CLR      R1
1638 005720 026061 001566 002026      22$:  CMP      BTABL(R0),GENER(R1)
1639 005726 001416          23$:  BEQ      21$
1640 005730 062701 000002      ADD      #2,R1
1641 005734 005761 002026      TST      GENER(R1)
1642 005740 001367          BNE      23$
1643 005742 016037 001566 002766      MOV      BTABL(R0),TEMP2
1644 005750 016037 001326 002770      MOV      ATABL(R0),TEMP3
1645 005756 104057          ERROR!57
1646 005760 000137 004546          JMP      MONIT
1647 005764 016137 002114 002202      21$:  MOV      MODUL(R1),DMUT
1648 005772 016037 001326 002172      MOV      ATABL(R0),TADDR
1649 006000 016137 002062 002166      MOV      MUT(R1),$MUT
1650 006006 010146          MOV      R1,-(SP) ;:PUSH R1 ON STACK
1651 006010 010046          MOV      R0,-(SP) ;:PUSH R0 ON STACK
1652 006012 004777 174164          JSR      PC,@DMUT ;:TEST THIS MODULE
1653 006016 012600          MOV      (SP)+,R0 ;:POP STACK INTO R0
1654 006020 012601          MOV      (SP)+,R1 ;:POP STACK INTO R1
1655 006022 022761 000014 002062      CMP      #14,MUT(R1) ;:WAS IT A014
1656 006030 001037          BNE      30$
1657 006032 010046          MOV      R0,-(SP) ;:PUSH R0 ON STACK
1658 006034 010146          MOV      R1,-(SP) ;:PUSH R1 ON STACK
1659 006036 013746 000004      MOV      ERRVEC,-(SP) ;:PUSH ERRVEC ON STACK
1660 006042 012737 006264 000004      MOV      #31$,ERRVEC
1661 006050 012737 000040 002152      MOV      #40,MXNUM ;:SET FIRST MUX NUMBER
1662 006056 152777 000001 174124      32$:  BISB     #RBIT,@CSR
1663 006064 113777 002152 174706      MOV      MXNUM,@STAT1
1664 006072 105777 174702          TST      @STAT1 ;:CHECK IF RESPONDS
1665 006076 004737 023710          JSR      PC,MUX1 ;:TEST MUX
1666 006102 062737 000040 002152      33$:  ADD      #40,MXNUM ;:DO NEXT MUX
1667 006110 022737 000400 002152      CMP      #400,MXNUM ;:LAST ONE?
1668 006116 001357          BNE      32$
1669 006120 012637 000004      MOV      (SP)+,ERRVEC ;:POP STACK INTO ERRVEC
1670 006124 012601          MOV      (SP)+,R1 ;:POP STACK INTO R1
1671 006126 012600          MOV      (SP)+,R0 ;:POP STACK INTO R0
1672 006130 062700 000002      30$:  ADD      #2,R0
1673 006134 005760 001566          TST      BTABL(R0)
1674 006140 001266          BNE      22$
1675 006142 004737 025502          JSR      PC,CNTRC ;:CONTROL C ?
1676 006146 004737 016652      26$:  JSR      PC,KLEER
1677 006152 005777 174006          TST      @XXDP ;:RUNNING UNDER XXDP CHAIN MODE?
1678 006156 001051          BNE      LOGIC
1679 006160 132737 000001 001222      BITB     #BIT0,$ENV ;:RUNNING UNDER APT?
1680 006166 001045          BNE      LOGIC
1681 006170 005737 002246          TST      PASCNT
1682 006174 001645          BEQ      27$
1683 006176 005237 001210          INC      $PASS
1684 006202 023737 002246 001210      CMP      PASCNT,$PASS ;:IS IT LAST $PASS
```



```
1685 006210 001237          BNE      25$
1686 006212 005037 001210   CLR      $PASS
1687 006216 032737 020000 000176 BIT      #BIT13,SWREG      ;INHIBIT PRINT
1688 006224 001005          BNE      20$
1689 006226 013746 002246   MOV      PASCNT,-(SP)      ;;SAVE PASCNT FOR TYPEOUT
                                TYP0C      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1690 006234 104402          TYPE     ,M11              ;# OF PASSES COMPLETED
1691 006240 032737 040000 000176 20$:   BIT      #BIT14,SWREG      ;DO WE LOOP ON TESTS
1692 006246 001220          BNE      25$
1693 006250 104401 036546   24$:   TYPE     ,MASS17      ;END OF SYSTEM TEST
1694 006254 005037 002240   CLR      AFLAG
1695 006260 000137 004546   JMP      MONIT
1696
1697
1698 006264 022626          31$:   CMP      (SP)+,(SP)+
1699 006266 152777 000001 173714 BISB     #RBIT,@CSR
1700 006274 105777 174500   TSTB    @STAT1
1701 006300 000700          BR      33$
1702
1703
1704 006302
```

```
LOGIC:
.SBTTL  END OF PASS ROUTINE
*****
;*INCREMENT THE PASS NUMBER ($PASS)
;*IF THERES A MONITOR GO TO IT
;*IF THERE ISN'T JUMP TO AUTO
$EOP:
```

```
006302
006302 000004          SCOPE
006304 005037 001116   CLR      $TSTNM          ;;ZERO THE TEST NUMBER
006310 005237 001210   INC      $PASS           ;;INCREMENT THE PASS NUMBER
006314 042737 100000 001210 BIC      #100000,$PASS    ;;DON'T ALLOW A NEG. NUMBER
006322 005327          DEC      (PC)+          ;;LOOP?
006324 000001          $EOPCT: .WORD    1
006326 003013          BGT      $DOAGN          ;;YES
006330 012737          MOV      (PC)+,@(PC)+      ;;RESTORE COUNTER
006332 000001          $ENDCT: .WORD    1
006334 006324          $EOPCT
006336 013700 000042   $GET42: MOV      @#42,R0      ;;GET MONITOR ADDRESS
006342 001405          BEQ      $DOAGN          ;;BRANCH IF NO MONITOR
006344 000005          RESET          ;;CLEAR THE WORLD
006346 004710          $ENDAD: JSR      PC,(R0)      ;;GO TO MONITOR
006350 000240          NOP          ;;SAVE ROOM
006352 000240          NOP          ;;FOR
006354 000240          NOP          ;;ACT11
006356          $DOAGN:
006356 000137          JMP      @(PC)+          ;;RETURN
006360 005442          $RTNAD: .WORD    AUTO
```

```

1706          .SBTTL LOOP TEST
1707
1708          ;THIS TEST ENABLE FIELD ENGINEER TO CONNECT OUTPUT MODULE TO INPUT MODULE
1709          ;OUTPUT TEST PATTERNS AND READ THEM BACK
1710          :
1711          :
1712 006362 104401 036772          LOOP: TYPE ,MASS23          ;CONNECT OUTPUT TO INPUT MODULE
1713 006366 012777 025556 173706 MOV #KBINT,@KBVEC
1714 006374 152777 000100 172556 BISB #BIT6,@$TKS          ;ENABLE KB INTERRUPT
1715 006402 104401 036705          TYPE ,MASS21          ;TYPE ADDRESS OF OUTPUT MUT
1716 006410 012637 002172          RDOCT
1717 006414 104401 036740          MOV (SP)+,TADDR          ;;POP STACK INTO TADDR
1718 006420 104410          TYPE ,MASS22          ;TYPE ADDRESS OF INPUT MUT
1719 006422 012637 002242          RDOCT
1720 006426 112777 000004 173554 MOV (SP)+,TEMP1          ;;POP STACK INTO TEMP1
1721 006434 117700 173532          MOVB #GBIT,@CSR          ;SET GENERIC BIT
1722 006440 020027 000041          MOVB @TADDR,R0          ;GENERIC CODE OF OUTPUT MUT
1723 006444 001051          CMP R0,#41          ;IS IT M6010?
1724 006446 117701 173570          BNE 1$          ;NOT M6010, ERROR
1725 006452 020127 000141          MOVB @TEMP1,R1          ;GENERIC CODE OF INPUT MUT
1726 006456 001403          CMP R1,#141          ;IS IT M5010?
1727 006460 020127 000121          BEQ 2$
1728 006464 001044          CMP R1,#121          ;IS IT M5011?
1729 006466 005037 001210          BNE 6$
1730 006472 004737 026074          2$: CLR $PASS
1731 006476 004737 025414          3$: JSR PC,LOPTST          ;TEST SUBROUTINE
1732 006502 004737 025502          JSR PC,CLRINT          ;CLEAR ALL INTERRUPTS
1733 006506 005237 001210          JSR PC,CNTRC          ;CONTROL-C?
1734 006512 023737 002246 001210          INC $PASS
1735 006520 001364          CMP PASCNT,$PASS          ;
1736 006522 032737 020000 000176          BNE 3$
1737 006530 001005          BIT #BIT13,SWREG          ;INHIBIT PRINTOUT
1738 006532 013746 002246          BNE 5$
1739 006536 104402          MOV PASCNT,-(SP)          ;;SAVE PASCNT FOR TYPEOUT
1740 006540 104401 033363          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1741 006544 032737 040000 000176 5$: TYPE ,M11          ;PASSES COMPLETED
1742 006552 001345          BIT #BIT14,SWREG          ;LOOP
1743 006554 104401 037033          BNE 2$
1744 006560 004737 016652          TYPE ,MASS25          ;END OF LOOP TEST
1745 006564 000137 004546          JSR PC,KLEER
1746 006570 104401 037056          JMP MONIT
1747 006574 000402          TYPE ,MASS26          ;WRONG MODULE -OUTPUT MUST BE M6010
1748 006576 104401 037125          BR 4$
1749 006602 004737 016652          6$: TYPE ,MASS28          ;WRONG MODULE- INPUT MUST BE EITHER M5010 OR M5011
1749 006606 000137 004546          4$: JSR PC,KLEER
1749 006606 000137 004546          JMP MONIT
  
```



```

1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804 007012 013746 000004
1805 007016 012737 007420 000004
1806 007024 012777 025556 173250
      007032 152777 000100 172120
1807 007040 013700 002224
1808 007044 112777 000004 173136
1809 007052 111001
1810 007054 042701 177400
1811 007060 004737 025224
1812 007064 104401 035651
1813 007070 010046
      007072 104402
1814 007074 000164 007100
1815 007100 104401 035663
1816 007104 000535
1817 007106 104401 035732
1818 007112 000532
1819 007114 104401 036001
1820 007120 000531
1821 007122 104401 036045
1822 007126 000527
1823 007130 104401 036077
1824 007134 000521
1825 007136 104401 036147
1826 007142 000520
1827 007144 104401 036210
1828 007150 000516
1829 007152 104401 036254
1830 007156 000513
1831 007160 104401 040432
1832 007164 000507
1833 007166 104401 040471
1834 007172 000502
1835 007174 104401 037210
1836 007200 000475
1837 007202 104401 037253
1838 007206 000412
1839 007210 104401 037321
1840 007214 000407
1841 007216 104401 036307
1842 007222 010146
      007224 104402
1843 007226 104401 035646
1844 007232 000465
  
```

.SBTTL MAP OF DBUS

```

:;*****
      ;THIS TEST WILL LIST ALL I/O INTERFACES CONNECTED TO IOCM
      ;IT WILL ALSO LIST QBUS ADDRESSES
:;*****

MAPE:  MOV @#ERRVEC,-(SP) ;SAVE ERROR VECTOR
      MOV #1$,ERRVEC ;SET ERROR VECTOR
      MOV #KBINT,@KBVEC
      BISB #BIT6,@$TKS ;ENABLE KB INTERRUPT
      MOV BASE,R0 ;R0=171000
      MOV #GBIT,@CSR ;SET GENERIC CODE ENABLE
5$:  MOV (R0),R1 ;R1=GENERIC CODE,READ GENERIC CODE
      BIC #177400,R1
      JSR PC,GENCOD ;FIND WHAT I/O IT IS
      TYPE ,MASS2 ;ADDRESS
      MOV R0,-(SP) ;SAVE R0 FOR TYPEOUT
      TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
      JMP 3$(R4)
3$:  TYPE ,MASS3 ;M5010
      BR 10$
      TYPE ,MASS4 ;M5011
      BR 10$
      TYPE ,MASS5 ;M5012
      BR 12$
      TYPE ,MASS6 ;M5013
      BR 13$
      TYPE ,MASS7 ;M6010
      BR 10$
      TYPE ,MASS8 ;M6011
      BR 12$
      TYPE ,MASS9 ;M6012
      BR 13$
      TYPE ,MASS10 ;M6013
      BR 13$
      TYPE ,MASS48 ;M5012-YA
      BR 12$
      TYPE ,MASS49 ;M6010-YA
      BR 10$
      TYPE ,MASS29 ;A630
      BR 9$
      TYPE ,MASS30 ;A014 SE
      BR 30$
      TYPE ,MASS31 ;A014 DM
      BR 30$
      TYPE ,MASS11 ;UNKNOWN GENERIC CODE
      MOV R1,-(SP) ;SAVE R1 FOR TYPEOUT
      TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
      TYPE ,MASS0 ;CR,LF
      BR 13$
  
```



```

1845 007234 012737 007452 000004 30$: MOV #2$,ERRVEC ;SET NEW TIME OUT VECTOR
1846 007242 005004 CLR R4 ;SET MUX # POINTER
1847 007244 005003 CLR R3
1848 007246 062700 000002 ADD #2,R0 ;SET A/D TO SECOND ADDRESS
1849 007252 062703 000040 25$: ADD #40,R3 ;INC MUX #
1850 007256 005204 INC R4
1851 007260 032704 000010 BIT #BIT3,R4 ;LAST MUX?
1852 007264 001037 BNE 24$
1853 007266 110310 MOV# R3,(R0) ;SET MUX #
1854 007270 111001 MOV# (R0),R1
1855 007272 042701 177400 BIC #177400,R1 ;GET GENERIC CODE
1856 007276 104401 037375 TYPE ,MASS32 ;MUX #
1857 007302 010446 MOV R4,-(SP) ;SAVE R4 FOR TYPEOUT
007304 104402 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
1858 007306 022701 000342 CMP #342,R1
1859 007312 001003 BNE 21$
1860 007314 104401 037404 TYPE ,MASS33 ;A156 - SINGLE ENDED
1861 007320 000754 BR 25$
1862
1863 007322 022701 000322 21$: CMP #322,R1
1864 007326 001003 BNE 22$
1865 007330 104401 037434 TYPE ,MASS34 ;A156 - DIFFERENTIAL
1866 007334 000746 BR 25$
1867
1868 007336 022701 000323 22$: CMP #323,R1
1869 007342 001003 BNE 23$
1870 007344 104401 037471 TYPE ,MASS35 ;A157
1871 007350 000740 BR 25$
1872
1873 007352 23$: MOV R1,-(SP) ;SAVE R1 FOR TYPEOUT
007352 010146 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
007354 104402 TYPE ,MASS11 ;UNKNOWN GENERIC CODE
1874 007356 104401 036307 BR 25$
1875 007362 000733
1876
1877 007364 012737 007420 000004 24$: MOV #1$,ERRVEC ;RESTORE TIME-OUT VECTOR
1878 007372 000404 BR 12$
1879
1880 007374 062700 000004 9$: ADD #4,R0
1881 007400 005200 10$: INC R0 ;INC ADDRESS BY APPROPRIATE NUMBER OF BYTES
1882 007402 005200 11$: INC R0
1883 007404 005200 12$: INC R0
1884 007406 005200 13$: INC R0
1885 007410 020037 002212 CMP R0,IAR ;LAST ADDRESS ?
1886 007414 103616 BLO 5$ ;NO, DO IT AGAIN
1887 007416 000407 BR 6$
1888 007420 022626 1$: CMP (SP)+,(SP)+ ;HERE IF ADDRESS IS NOT RESPONDING
1889 007422 004737 025502 JSR PC,CNTRC
1890 007426 005200 INC R0 ;INC ADDRESS
1891 007430 020037 002212 CMP R0,IAR ;LAST ONE ?
1892 007434 001206 BNE 5$
1893 007436 012637 000004 6$: MOV (SP)+,@#ERRVEC ;RESTORE ERROR VECTOR
1894 007442 004737 016652 JSR PC,KLEER
1895 007446 000137 004546 JMP MONIT
1896
1897
1898 007452 022626 2$: CMP (SP)+,(SP)+ ;HERE IF NO MUXES
  
```

1899	007454	152777	000001	172526
1900	007462	004737	025502	
1901	007466	105710		
1902	007470	000137	007252	

BISB	#RBIT,@CSR
JSR	PC,CNTRC
TSTB	(R0)
JMP	25\$

;CLEAR HANG A/D

1904
1905
1906
1907
1908
1909
1910
1911

.SBTTL IOCM TEST

::*****

:THIS PART OF DIAGNOSTIC TEST THE IOCM
:IT HAS 9 TESTS. IT CHECKS IF ALL THE BITS OF THE IOCM
:CAN BE SET AND CLEAR , CHECKS MAINTENANCE
:INTERRUPT AND CHECKS ALL ADDRESSES IN MAINTENANCE MODE

1913
1914 007474
1915 007500 013746 000004 172574
007474 012777 025556 171444
007506 152777 000100
1916 007514 004737 027060
1917 007520 007542
1918 007522 104401 036523
1919 007526 012637 000004
1920 007532 004737 016652
1921 007536 000137 004546
1922
1923
1924
1925 007542
007542 000004
1926 007544 112737 000001 001206
1927 007552 012737 007622 000004
1928 007560 112777 000074 172422
1929 007566 004737 016652
1930 007572 005037 001142
1931 007576 005037 001140
1932 007602 117737 172402 001142
1933 007610 105737 001142
1934 007614 001405
1935 007616 104001
1936 007620 000207
1937 007622 104003
1938 007624 022626
1939 007626 000207
1940 007630

```
*****  
IOCM:  MOV      ERRVEC,-(SP)      ;;PUSH ERRVEC ON STACK  
      MOV      #KBINT,@KBVEC  
      BISB     #BIT6,@$TKS      ;ENABLE KB INTERRUPT  
      JSR      PC,SWLOOP  
      TIOCM    ;DO IOCM TEST  
      TYPE     ,MASS16          ;IOCM TEST PASSED  
      MOV      (SP)+,ERRVEC     ;;POP STACK INTO ERRVEC  
      JSR      PC,KLEER  
      JMP      MONIT  
  
      ;THIS TEST CHECKS IF EACH BIT OF CSR  
      ;IS CLEAR BY CBIT  
TIOCM:  
*****  
TST1:  SCOPE  
      MOVB     #1,$TESTN  
      MOV      #10$,ERRVEC  
      MOVB     #74,@CSR         ;SET ALL BITS  
      JSR      PC,KLEER         ;SET CBIT  
      CLR      $BDDAT  
      CLR      $GDDAT  
      MOVB     @CSR,$BDDAT     ;STORE CONTENTS OF CSR IN BDDAT  
      TSTB     $BDDAT         ;IS CSR CLEAR  
      BEQ      2$             ;YES  
      ERROR!1                ;CBIT NOT CLEARING IOCM  
      RTS      PC             ;FATAL ERROR RETURN TO MONITOR  
10$:   ERROR!3                ;IOCM NOT RESPONDING  
      CMP      (SP)+,(SP)+     ;RESTORE STACK  
      RTS      PC  
2$:
```



```
1942                                     ;THIS TEST CHECKS E BIT
1943                                     :*****
1944 007630 000004                       TST2: SCOPE
1945 007632 112737 000002 001206         MOVB #2,$TESTN
1946 007640 012737 000100 001140         MOV #EBIT,$GDDAT ;SET TESTED BIT
1947 007646 113737 033356 032172         MOVB M7,EM2X ;SET EBIT IN ERROR MESSAGE
1948 007654 004737 025300                 JSR PC,BITSET
1949 007660 104002                         ERROR!2 ;BIT IS NOT SETTING
1950 007662 005037 001140                 CLR $GDDAT
1951 007666 004737 025300                 JSR PC,BITSET
1952                                     ERROR!2
1953                                     ;THIS TEST CHECKS MBIT
1954                                     :*****
1955 007674 000004                       TST3: SCOPE
1956 007676 112737 000003 001206         MOVB #3,$TESTN
1957 007704 012737 000240 001140         MOV #MBIT!FBIT,$GDDAT;SET MAITENANCE BIT
1958 007712 113737 033353 032172         MOVB M6,EM2X ;SET ERROR MESSAGE
1959 007720 004737 025300                 JSR PC,BITSET
1960 007724 104002                         ERROR!2 ;BIT NOT SETTING
1961 007726 005037 001140                 CLR $GDDAT
1962 007732 004737 025300                 JSR PC,BITSET
1963                                     ERROR!2 ;BIT NOT CLEARING
1964                                     ;THIS TEST CHECKS DBIT
1965                                     :*****
1966                                     TST4: SCOPE
1967 007740 000004                       MOVB #4,$TESTN
1968 007742 112737 000004 001206         MOV #DBIT,$GDDAT
1969 007750 012737 000020 001140         MOVB M5,EM2X ;FIX ERROR MESSAGE
1970 007756 113737 033350 032172         JSR PC,BITSET ;SET D BIT
1971 007770 104002                         ERROR!2 ;BIT NOT SETTING
1972 007772 005037 001140                 CLR $GDDAT
1973 007776 004737 025300                 JSR PC,BITSET
1974 010002 104002                         ERROR!2 ;BIT NOT CLEARING
1975 010004 005037 001140                 CLR $GDDAT ;INITIALIZE EXPECTED DATA.
1976 010010 112777 000074 172172         MOVB #74,@CSR ;SET FEW BITS AT CSR
1977 010016 142777 000020 172164         BICB #DBIT,@CSR ;CLEAR DBIT
1978 010024 005037 002150                 CLR YLOOP ;WAIT
1979 010030 005237 002150 000007 64$: INC YLOOP
1980 010034 023727 002150                 CMP YLOOP,#7
1981 010042 001372                         BNE 64$
1982 010044 117737 172140 001142         MOVB @CSR,$BDDAT
1983 010052 132737 000177 001142         BITB #177,$BDDAT
1984 010060 001401                         BEQ 1$ ;CHECK IF DBIT CLEARING CLEARS CSR
1985 010062 104016                         ERROR!16 ;CLEARING DBIT DOES NOT CLEAR CSR
1986 010064 004737 016652                 JSR PC,KLEER
```

```
1985
1986
1987
1988 010070 000004
1989 010100 012737 000005 001206
1990 010106 113737 033345 032172
1991 010114 004737 025300
1992 010120 104002
1993 010122 005037 001140
1994 010126 004737 025300
1995 010132 104002
1996 010134 004737 025414
1997
1998
1999
2000 010140 000004
2001 010150 012737 000006 001206
2002 010156 113737 033342 032172
2003 010164 004737 025300
2004 010170 104002
2005 010172 005037 001140
2006 010176 004737 025300
2007 010202 104002
2008
2009
2010
2011
2012 010204 000004
2013 010206 112737 000007 001206
2014 010214 012737 000001 001140
2015 010222 113737 033340 032172
2016 010230 004737 025300
2017 010236 005037 001140
2018 010242 004737 025300
2019 010246 104002

;THIS TEST CHECKS TBIT
:*****
TST5: SCOPE
      MOVB #5,$TESTN
      MOV #TBIT,$GDDAT
      MOVB M4,EM2X ;SET ERROR MESSAGE
      JSR PC,BITSET ;SET AND CHECK T BIT
      ERROR!2 ;BIT IS NOT SETTING
      CLR $GDDAT
      JSR PC,BITSET ;CLEAR T BIT
      ERROR!2 ;BIT NOT CLEAR
      JSR PC,CLRINT ;CLEAR ALL INTERRUPT CAUSED BY T BIT

;THIS TEST CHECKS GBIT
:*****
TST6: SCOPE
      MOVB #6,$TESTN
      MOV #GBIT,$GDDAT
      MOVB M3,EM2X ;FIX ERROR MESSAGE
      JSR PC,BITSET ;SET AND CHECK G BIT
      ERROR!2 ;BIT NOT SETTING
      CLR $GDDAT
      JSR PC,BITSET ;CLEAR G BIT
      ERROR!2 ;BIT'S NOT CLEARING

;THIS TEST CHECKS RBIT
:*****
TST7: SCOPE
      MOVB #7,$TESTN
      MOV #RBIT,$GDDAT
      MOVB M2,EM2X ;SET ERROR MESSAGE
      JSR PC,BITSET ;SET BIT
      ERROR!2 ;BIT IS NOT SETTING
      CLR $GDDAT
      JSR PC,BITSET ;CLEAR BIT
      ERROR!2 ;BIT IS NOT CLEAR
```


2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031

2032	010250	000004		
2033	010252	112737	000010	001206
2034	010260	112777	000040	171722
2035	010266	013700	002224	
2036	010272	005001		
2037	010274	005037	001140	
2038	010300	005037	001142	
2039	010304	111001		
2040	010306	042701	177400	
2041	010312	120001		
2042	010314	001405		
2043	010316	110037	001140	
2044	010322	110137	001142	
2045	010326	104004		
2046	010330	005200		
2047	010332	122700	000376	
2048	010336	001362		
	010340	004737	016652	

: THIS TEST CHECKS ALL BITS OF DBUS IN A MAINTENANCE MODE.
: IF MBIT IS SET AND CPU ADDRESSES ANY LOCATION BETWEEN
: 171000 AND 171375 IT SHOULD READ BACK A LOWER
: BYTE OF AN ADDRESS.

```
TST10: SCOPE
        MOVB #10,$TESTN
        MOVB #MBIT,@CSR      ;SET MAINTENANCE MODE
        MOV  BASE,R0
        CLR  R1
        CLR  $GDDAT
        CLR  $BDDAT
1$:     MOVB (R0),R1          ;READ FIRST ADDRESS
        BIC  #177400,R1
        CMPB R0,R1          ;CHECK IF DBUS=ADDRESS
        BEQ  2$
        MOVB R0,$GDDAT
        MOVB R1,$BDDAT
        ERROR!4           ;DBUS BIT STACK
2$:     INC  R0
        CMPB #376,R0       ;IS IT THE LAST ADDRESS
        BNE  1$
        JSR  PC,KLEER      ;CLEAR IOCM
```

2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092

010344 000004
010346 112737 000011 001206
010354 012777 010450 171632
010362 012777 000340 171626
010370 013746 000004
010374 012737 010562 000004
010402 012746 000000
010406 012746 010414
010412 000002
010414 000240
010416 112777 000140 171564
010424 005037 002150
010430 005237 002150
010434 023727 002150 177777
010442 001372
010444 104014
010446 000431
010450 022626
010452 117737 171534 002232
010460 122737 000377 002232
010466 001407
010470 012737 000377 001140
010476 013737 002232 001142
010504 104006
010506 152777 000001 171474
010514 105777 171470
010520 132777 000200 171462
010526 001401
010530 104015
010532 004737 016652
010536 012637 000004
010542 012746 000000
010546 012746 010554
010552 000002
010554 000240
010556 000004
010560 000207
010562 022626
010564 104022
010566 000761

::*****
:THIS TEST WILL CHECK MAINTENANCE INTERRUPT
:IF MBIT & EBIT ARE SET IOCM WILL GENERATE AN INTERRUPT
:AT LOCATION 234 AND IAR WILL HAVE LOWER BYTE
:OF CSR ADDRESS
::*****

::*****

TST11: SCOPE
MOV #11,\$TESTN
MOV #5\$,@VECTO ;SET VECTOR ADDRESS
MOV #PR7,@VECTOA ;SET VECTOR+2 ADDRESS
MOV ERRVEC,-(SP) ;;PUSH ERRVEC ON STACK
MOV #8\$,ERRVEC
MOV #PR0,-(SP) ;SET PSW TO PRIORITY 0
MOV #64\$,-(SP)
RTI
NOP
64\$: MOV #EBIT!MBIT,@CSR;START INTERRUPT
1\$: CLR YLOOP ;WAIT
INC YLOOP
65\$: CMP YLOOP,#-1
BNE 65\$
ERROR!14 ;NO INTERRUPT
BR 6\$
5\$: CMP (SP)+,(SP)+ ;INTERRUPT WORKED
MOV @IAR,TEMP ;READ ADDRESS OF INTERRUPTING MODULE
CMPB #377,TEMP ;IAR SHOULD HAVE ALL ONES
BEQ 7\$
MOV #377,\$GDDAT
MOV TEMP,\$BDDAT
ERROR!6 ;WRONG PATTERN IN IAR
7\$: BISB #RBIT,@CSR
TSTB @CSR ;CLEAR INTERRUPT
BITB #FBIT,@CSR ;CHECK IF INTERRUPT CLEAR
BEQ 6\$
ERROR!15 ;RIF BIT NOT CLEARING INTERRUPT
6\$: JSR PC,KLEER ;CLEAR CSR
MOV (SP)+,ERRVEC ;;POP STACK INTO ERRVEC
MOV #PR0,-(SP) ;SET PSW TO PRIORITY 0
MOV #66\$,-(SP)
RTI
66\$: NOP
SCOPE
RTS PC
8\$: CMP (SP)+,(SP)+
ERROR!22 ;ADDRESS TEST WITH MBIT SET NOT WORKING
BR 6\$

2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115

010570
010570 000004
010572 112737 000012 001206
010600 152777 000020 171402
010606 112737 000000 001140
010614 012737 000004 002170
010622 004737 025016
010626 152777 000010 171354
010634 112737 000377 001140
010642 012737 000004 002170
010650 004737 025016
010654 142777 000010 171326
010662 004737 025414
010666 000004
010670 000207

```
.SBTTL DIGITAL MODUL TEST ,M5010
;:*****
;THIS TEST WILL CHECK M5010 MODULE
;32 BIT NONISOLATED DC SENSE
;:*****

M5010:
;:*****
TST12: SCOPE
        MOVB    #12,$TESTN
        BISB    #DBIT,@CSR      ;SET DISABLE BIT
        MOVB    #0,$GDDAT      ;SET WHAT DATA SHOULD BE
        MOV     #4,BYTNUM      ;SET NUMBER OF BYTES
        JSR     PC,TSTBYT      ;CHECK IF DATA IS CORRECT

        BISB    #TBIT,@CSR
        MOVB    #377,$GDDAT    ;SET WHAT DATA SHOULD BE
        MOV     #4,BYTNUM      ;SET NUMBER OF BYTES
        JSR     PC,TSTBYT      ;CHECK IF DATA IS CORRECT

        BICB    #TBIT,@CSR
        JSR     PC,CLRINT
        SCOPE
        RTS     PC
```

```
2117
2118 .SBTTL DIGITAL MODULE TEST ,M5013
2119
2120 ;:*****
2121
2122 ;THIS TEST CHECKS M5013 MODULE
2123 ;8 BIT AC SENSE
2124
2125 ;:*****
2126
2127
2128 010672 M5013:
2129 ;:*****
2130 010672 000004 TST13: SCOPE
2131 010674 112737 000013 001206 MOVB #13,$TESTN
2132 010702 152777 000020 171300 BISB #DBIT,@CSR ;SET DISABLE BIT
2133 010710 112737 000000 001140 MOVB #0,$GDDAT ;SET WHAT DATA SHOULD BE
2134 010716 012737 000001 002170 MOV #1,BYTNUM ;SET NUMBER OF BYTES
2135 010724 004737 025016 JSR PC,TSTBYT ;CHECK IF DATA IS CORRECT
2136
2137 010730 152777 000010 171252 BISB #TBIT,@CSR ;SET COMPLIM BIT
2138 010736 004737 025414 JSR PC,CLRINT
2139 010742 112737 000377 001140 MOVB #377,$GDDAT ;SET WHAT DATA SHOULD BE
2140 010750 012737 000001 002170 MOV #1,BYTNUM ;SET NUMBER OF BYTES
2141 010756 004737 025016 JSR PC,TSTBYT ;CHECK IF DATA IS CORRECT
2142 010762 142777 000010 171220 BICB #TBIT,@CSR
2143 010770 004737 025414 JSR PC,CLRINT
2144
2145 010774 012746 000000 MOV #PRO,-(SP) ;SET PSW TO PRIORITY 0
2146 011000 012746 011006 MOV #64$,-(SP)
2147 011004 000002 RTI
2148 011006 000240 .64$: NOP
2149 011010 012777 011114 171176 MOV #5$,@VECTO ;SET INTERRUPTVECTOR ADDRESS
2150 011016 012777 000340 171172 MOV #PR7,@VECTOA
2151 011024 152777 000010 171156 BISB #TBIT,@CSR ;START INTERRUPT
2152 011032 152777 000100 171150 BISB #EBIT,@CSR ;ENABLE INTERRUPT
2153 011040 005037 002150 CLR YLOOP ;WAIT
2154 011044 005237 002150 .65$: INC YLOOP
2155 011050 023727 002150 177777 CMP YLOOP,#-1
2156 011056 001372 BNE 65$
2157 011060 104005 ERROR!5 ;NO INTERRUPT
2158 011062 142777 000100 171120 .7$: BICB #EBIT,@CSR ;CLEAR CSR
2159 011070 142777 000010 171112 BICB #TBIT,@CSR
2160 011076 004737 025414 JSR PC,CLRINT ;CLEAR ALL INTERRUPT
2161 011102 012737 005044 000004 MOV #TMOVEC,ERRVEC
2162 011110 000004 SCOPE
2163 011112 000207 RTS PC ;RETURN TO MONITOR
2164
2165 011114 022626 .5$: CMP (SP)+,(SP)+ ;RESET STACK
2166 011116 005037 002242 CLR TEMP1
2167 011122 012737 011324 000004 MOV #4$,ERRVEC
2168 011130 117737 171056 002232 .6$: MOVB @IAR,TEMP ;CHECK WHICH I/O INTERRUPTED
2169 011136 152777 000001 171044 BISB #RBIT,@CSR ;HERE IF INTERRUPT
2170 011144 123737 002232 002172 CMPB TEMP,IADDR ;IS IT MUT
2171 011152 001414 BEQ 2$
2172 011154 053737 002224 002232 BIS BASE,TEMP ;ASSEMBLE ADDRESS OF INTERRUPTING MODULE
```


2163	011162	105777	171044		TSTB	@TEMP		
2164	011166	005237	002242		INC	TEMP1		
2165	011172	022737	000400	002242	CMP	#400,TEMP1		;LOOP NO MORE THEN 400 TIMES
2166	011200	001353			BNE	6\$		
2167	011202	104005			ERROR!5			;NO INTERRUPT
2168	011204	105777	170762		TSTB	@TADDR		;CLEAR INTERRUPT
2169	011210	132777	000200	170772	BITB	#FBIT,@CSR		;CHECK IF INTERRUPT CLEAR
2170	011216	001413			BEQ	8\$		
2171	011220	117737	170766	002232	MOVB	@IAR,TEMP		
2172	011226	023737	002232	002172	CMP	TEMP,TADDR		
2173	011234	001002			BNE	9\$		
2174	011236	104015			ERROR!15			;RIF BIT NOT CLEARING INTERRUPT
2175	011240	000710			BR	7\$		
2176	011242	004737	025414		JSR	PC,CLRINT		
2177	011246	132777	000010	170734	BITB	#TBIT,@CSR		
2178	011254	001702			BEQ	7\$		
2179	011256	012746	000000		MOV	#PRO,-(SP)		;SET PSW TO PRIORITY 0
	011262	012746	011270		MOV	#66,-(SP)		
	011266	000002			RTI			
	011270	000240			NOP			
2180	011272	142777	000010	170710	BICB	#TBIT,@CSR		;CHECK IF CLEARING TBIT CAUSE INTERRUPT
2181	011300	005037	002150		CLR	YLOOP		;WAIT
	011304	005237	002150		INC	YLOOP		
	011310	023727	002150	177777	CMP	YLOOP,#-1		
	011316	001372			BNE	67\$		
2182	011320	104005			ERROR!5			;NO INTERRUPT
2183	011322	000657			BR	7\$		
2184	011324	022626			CMP	(SP)+,(SP)+		
2185	011326	104005			ERROR!5			;NO INTERRUPT
2186	011330	000137	011204		JMP	2\$		
2187								

2189
2190
2191
2192
2193
2194
2195
2196
2197
2198 011334
2199
011334 000004
2200 011336 112737 000014 001206
2201 011344 152777 000020 170636
2202 011352 013700 002172
2203 011356 105020
2204 011360 105020
2205 011362 105020
2206 011364 105020
2207 011366 005000
011370 005001
011372 005037 002154
011376 005037 002156
011402 116160 002160 002154
011410 013737 002172 002232
011416 060037 002232
011422 116177 002160 170602
011430 005002
011432 013737 002172 002242
011440 060237 002242
011444 117737 170572 001142
011452 123762 001142 002154
011460 001404
011462 116237 002154 001140
011470 104017
011472 005202
011474 022702 000004
011500 001354
011502 005201
011504 022701 000004
011510 001334
011512 005200
011514 022700 000004
011520 001323
2208 011522 000004
2209 011524 000207
2210
2211

```
.SBTTL DIGITAL MODULE TEST ,M6010,M6010YA  
:;*****  
:THIS TEST CHECKS M6010 MODULE  
:32 BIT NONISOLATED DC OUT  
:;*****  
M6010:  
:;*****  
TST14: SCOPE  
MOV#14,$TESTN  
BISB #DBIT,@CSR ;SET D BIT  
MOV TADDR,R0  
CLRB (R0)+ ;CLEAR ALL FOUR BYTES OF I/O REGISTER  
CLRB (R0)+  
CLRB (R0)+  
CLRB (R0)+  
CLR R0  
4$: CLR R1  
CLR DBUFF ;CLEAR IMIGE OF I/O REGISRERS  
CLR DBUFF+2  
3$: MOV#PATT(R1),DBUFF(R0);SET DATA PATTERN IN IMIGE  
MOV TADDR,TEMP  
ADD R0,TEMP  
MOV#PATT(R1),@TEMP ;SET DATA IN I/O REG  
CLR R2  
2$: MOV TADDR,TEMP1  
ADD R2,TEMP1  
MOV#@TEMP1,$BDDAT ;READ I/O REGISTER  
CMPB $BDDAT,DBUFF(R2);IS DATA OK?  
BEQ 1$  
MOV#DBUFF(R2),$GDDAT  
ERROR!17 ;DATA ERROR  
1$: INC R2  
CMP #4,R2  
BNE 2$ ;TEST IF OTHER BYTES ARE OK  
INC R1  
CMP #4,R1 ;LAST PATTERN?  
BNE 3$ ;TEST NEXT DATA PATTERN  
INC R0  
CMP #4,R0 ;LAST BYTE?  
BNE 4$ ;NO,TEST NEXT BYTE  
SCOPE  
RTS PC
```



```
2213 .SBTTL DIGITAL MODULE TEST , M6012,M6013
2214
2215 ;:*****
2216
2217 ;THIS TEST CHECKS MODULES M6012 AND M6013
2218 ;8 BIT AC AND DC OUT
2219
2220 ;:*****
2221
2222
2223 011526 M6012:
2224 011526 M6013:
2225 ;:*****
011526 000004 TST15: SCOPE
2226 011530 112737 000015 001206 MOVB #15,$TESTN
2227 011536 152777 000020 170444 BISB #DBIT,@CSR
2228 011544 012737 000125 001140 MOV #125,$GDDAT ;SET DATA PATTERN
011552 012737 000001 002170 MOV #1,BYTNUM ;SET NUMBER OF BYTES
011560 004737 025162 JSR PC,SETBYT ;SET DATA IN I/O MODULE
011564 012737 000001 002170 MOV #1,BYTNUM
011572 004737 025016 JSR PC,TSTBYT ;TEST IF DATA IS SET CORRECTLY
2229
2230
2231 011576 012737 000252 001140 MOV #252,$GDDAT ;SET DATA PATTERN
011604 012737 000001 002170 MOV #1,BYTNUM ;SET NUMBER OF BYTES
011612 004737 025162 JSR PC,SETBYT ;SET DATA IN I/O MODULE
011616 012737 000001 002170 MOV #1,BYTNUM
011624 004737 025016 JSR PC,TSTBYT ;TEST IF DATA IS SET CORRECTLY
2232
2233 011630 012737 000377 001140 MOV #377,$GDDAT ;SET DATA PATTERN
011636 012737 000001 002170 MOV #1,BYTNUM ;SET NUMBER OF BYTES
011644 004737 025162 JSR PC,SETBYT ;SET DATA IN I/O MODULE
011650 012737 000001 002170 MOV #1,BYTNUM
011656 004737 025016 JSR PC,TSTBYT ;TEST IF DATA IS SET CORRECTLY
2234
2235
2236 011662 012737 000000 001140 MOV #0,$GDDAT ;SET DATA PATTERN
011670 012737 000001 002170 MOV #1,BYTNUM ;SET NUMBER OF BYTES
011676 004737 025162 JSR PC,SETBYT ;SET DATA IN I/O MODULE
011702 012737 000001 002170 MOV #1,BYTNUM
011710 004737 025016 JSR PC,TSTBYT ;TEST IF DATA IS SET CORRECTLY
2237 011714 000004 SCOPE
2238 011716 000207 RTS PC
```

```

2240          .SBTTL DIGITAL MODULE TEST ,M6011
2241          ::*****
2242
2243          ;THIS TEST CHECKS MODULE M6011
2244          ;ONE SHOT DC OUT
2245          ::*****
2246
2247
2248 011720    M6011:
2249          ::*****
2250 011720    000004    TST16: SCOPE
2251 011722    112737    000016    001206    MOVB    #16,$TESTN
2252 011730    013737    002172    002174    MOV     TADDR,TADDR1
2253 011736    005237    002174    INC     TADDR1
2254 011742    152777    000020    170240    BISB   #DBIT,@CSR
2255 011750    005037    002226    CLR    CLK ;CLEAR CLOCK COUNTER
2256 011754    012777    025406    170266    MOV    #COUNT,@CLKVC ;SET LOC 100- CLOCK VECTOR
2257 011762    012777    000340    170262    MOV    #PR7,@CLKVCA ;SET LOC 102
2258 011770    012746    000000    MOV    #PRO,-(SP) ;SET PSW TO PRIORITY 0
2259 011774    012746    012002    MOV    #64$,-(SP)
2260 012000    000002    RTI
2261 012002    000240    64$: NOP
2262 012004    012737    000252    001140    MOV    #252,$GDDAT ;SET DATA PATTERN
2263 012012    012737    000002    002170    MOV    #2,BYTNUM ;SET NUMBER OF BYTES
2264 012020    004737    025162    JSR    PC,SETBYT ;SET DATA IN I/O MODULE
2265 012024    012737    000002    002170    MOV    #2,BYTNUM
2266 012032    004737    025016    JSR    PC,TSTBYT ;TEST IF DATA IS SET CORRECTLY
2267 012036    005001    CLR    R1
2268 012040    005201    11$: INC    R1 ;CHECK IF LINE CLOCK IS INTERRUPTING
2269 012042    001376    BNE    11$
2270 012044    005737    002226    TST    CLK
2271 012050    001010    BNE    1$
2272 012052    104025    ERROR!25 ;LINE CLOCK ISN'T INTERRUPTING
2273 012054    000453    BR     M6011B
2274 012056    105777    170110    2$: TSTB  @TADDR
2275 012062    001003    BNE    1$
2276 012064    105777    170104    TSTB  @TADDR1
2277 012070    001410    BEQ    M6011A
2278 012072    022737    001132    002226    1$: CMP  #602.,CLK ;WAIT 10SEC
2279 012100    001366    BNE    2$
2280 012102    005037    001140    CLR   $GDDAT
2281 012106    004737    025100    JSR   PC,TSTONE ;PATTERN SHOULD BE CLEAR
2282 012112    M6011A:
2283 012112    005037    002226    CLR   CLK ;CLEAR CLOCK COUNTER
2284 012116    012737    000125    001140    MOV  #125,$GDDAT ;SET DATA PATTERN
2285 012124    012737    000002    002170    MOV  #2,BYTNUM ;SET NUMBER OF BYTES
2286 012132    004737    025162    JSR  PC,SETBYT ;SET DATA IN I/O MODULE
2287 012136    012737    000002    002170    MOV  #2,BYTNUM
2288 012144    004737    025016    JSR  PC,TSTBYT ;TEST IF DATA IS SET CORRECTLY
2289 012150    105777    170016    2$: TSTB  @TADDR
2290 012154    001003    BNE    1$
2291 012156    105777    170012    TSTB  @TADDR1
2292 012162    001410    BEQ    M6011B
2293 012164    022737    001132    002226    1$: CMP  #602.,CLK ;WAIT 10 SEC
2294 012172    001366    BNE    2$
2295 012174    005037    001140    CLR   $GDDAT
2296 012200    004737    025100    JSR   PC,TSTONE ;PATTERN SHOULD BE CLEAR
    
```


2285	012204	000004			M6011B: SCOPE			
2286	012206	012777	025404	170034	MOV	#NOCLK,@CLKVC	;RESTORE	CLOCK VECTOR
2287	012214	000207			RTS	PC		

2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299 012216

.SBTTL DIGITAL MODULE TEST ,M5011

::*****

:THIS TEST CHECKS M5011 MODULE
:16 BIT CHANGE OF STATE SENSE

::*****

M5011:

::*****

```
TST17: SCOPE
MOV      #17,$TESTN
MOV      TADDR,COSADR
ADD      #2,COSADR
BISB     #DBIT!TBIT,@CSR;SET DISABLE BIT AND TBIT
MOV      #377,$GDDAT ;SET WHAT DATA SHOULD BE
MOV      #2,BYTNUM   ;SET NUMBER OF BYTES
JSR      PC,TSTBYT  ;CHECK IF DATA IS CORRECT

2305
2306 012270 142777 000010 167712 BICB     #TBIT,@CSR ;CLEAR COMPLIM BIT
2307 012276 112737 000000 001140 MOV      #0,$GDDAT ;SET WHAT DATA SHOULD BE
      012304 012737 000002 002170 MOV      #2,BYTNUM ;SET NUMBER OF BYTES
      012312 004737 025016 JSR      PC,TSTBYT ;CHECK IF DATA IS CORRECT
2308 012316 004737 025414 JSR      PC,CLRINT
2309 012322 012746 000000 MOV      #PRO,-(SP) ;SET PSW TO PRIORITY 0
      012326 012746 012334 MOV      #64$,-(SP)
      012332 000002 RTI
      012334 000240 NOP
2310 012336 012777 012442 167650 64$: MOV      #1$,@VECTO ;SET INTERRUPT VECTOR
2311 012344 012777 000340 167644 MOV      #PR7,@VECTOA
2312 012352 152777 000010 167630 BISB     #TBIT,@CSR ;START INTERRUPT
2313 012360 152777 000100 167622 BISB     #EBIT,@CSR ;ENABLE INTERRUPT
2314 012366 005037 002150 CLR      YLOOP ;WAIT
      012372 005237 002150 65$: INC      YLOOP
      012376 023727 002150 177777 CMP      YLOOP,#-1
      012404 001372 BNE     65$
2315 012406 104005 ERROR!5
2316 012410 142777 000100 167572 2$: BICB     #EBIT,@CSR ;NO INTERRUPT
2317 012416 142777 000010 167564 BICB     #TBIT,@CSR ;CLEAR CSR
2318 012424 012737 005044 000004 MOV      #TMOVEC,ERRVEC
2319 012432 004737 025414 JSR      PC,CLRINT ;CLEAR ALL INTERRUPTS
2320 012436 000004 SCOPE
2321 012440 000207 RTS      PC ;RETURN TO MONITOR
2322
2323 012442 022626 1$: CMP      (SP)+,(SP)+ ;ADJUST STOCK POINTER
2324 012444 005037 CLR      TEMP1
2325 012450 012737 013036 000004 MOV      #10$,ERRVEC
2326 012456 117737 167530 002232 7$: MOV      @IAR,TEMP ;FIND WHICH I/O INTERRUPTED
2327 012464 123737 002232 002172 CMPB     TEMP,TADDR ;IS IT MUT
2328 012472 001421 BEQ     8$ ;YES
2329 012474 053737 002224 002232 BIS      BASE,TEMP ;ASSEMBLE ADDRESS OF INTERRUPTING MODULE
2330 012502 152777 000001 167500 BISB     #RBIT,@CSR ;CLEAR INTERRUPT
2331 012510 105777 167516 TSTB     @TEMP
2332 012514 005237 002242 INC      TEMP1
2333 012520 022737 000400 002242 CMP      #400,TEMP1
```



```
2334 012526 001353 BNE 7$
2335 012530 104005 ERROR!5 ;NO INTERRUPT
2336 012532 000137 012410 JMP 2$
2337 012536 142777 000100 167444 8$: BICB #EBIT,@CSR
2338 012544 112737 000377 001140 MOVB #377,$GDDAT
2339 012552 117737 167422 001142 MOVB @COSADR,$BDDAT ;TEST COS REGISTER
2340 012560 123737 001140 001142 CMPB $GDDAT,$BDDAT ;IS IT ALL ONES
2341 012566 001401 BEQ 3$
2342 012570 104020 ERROR!20 ;COS REGISTER ERROR
2343 012572 152777 000001 167410 3$: BISB #RBIT,@CSR
2344 012600 105777 167374 TSTB @COSADR ;CLEAR INTERRUPT
2345 012604 005037 001140 CLR $GDDAT
2346 012610 117737 167364 001142 MOVB @COSADR,$BDDAT
2347 012616 001401 BEQ 4$
2348 012620 104013 ERROR!13 ;RIF BIT NOT CLEARING FF
2349 012622 005237 002200 4$: INC COSADR
2350 012626 005237 002172 INC TADDR ;TEST NEXT BYTE
2351 012632 117737 167354 002232 MOVB @IAR,TEMP
2352 012640 123737 002232 002172 CMPB TEMP,TADDR ;CHECK IF IT INTERRUPTED
2353 012646 001403 BEQ 11$
2354 012650 104005 ERROR!5 ;NO INTERRUPT
2355 012652 000137 012410 JMP 2$
2356 012656 112737 000377 001140 11$: MOVB #377,$GDDAT
2357 012664 117737 167310 001142 MOVB @COSADR,$BDDAT ;CHECK IF COS REGISTER IS ALL ONES
2358 012672 123737 001140 001142 CMPB $GDDAT,$BDDAT
2359 012700 001401 BEQ 5$
2360 012702 104020 ERROR!20 ;COS REGISTER ERROR
2361 012704 152777 000001 167276 5$: BISB #RBIT,@CSR
2362 012712 105777 167262 TSTB @COSADR ;CLEAR INTERRUPT
2363 012716 005037 001140 CLR $GDDAT
2364 012722 117737 167252 001142 MOVB @COSADR,$BDDAT
2365 012730 001401 BEQ 6$
2366 012732 104013 ERROR!13 ;RIF BIT NOT CLEARING FF
2367 012734 005337 002200 6$: DEC COSADR
2368 012740 005337 002172 DEC TADDR ;RESET ADDRESS
2369 012744 004737 025414 JSR PC,CLRINT ;CLEAR REMAINING INTERRUPTS
2370 012750 132777 000010 167232 BITB #TBIT,@CSR ;CHECK IF CLEARING T BIT CAUSE INTERRUPT
2371 012756 001614 BEQ 2$
2372 012760 012746 000000 MOV #PRO,-(SP) ;SET PSW TO PRIORITY 0
    012764 012746 012772 MOV #66$,-(SP)
    012770 000002 RTI
    012772 000240 66$: NOP
2373 012774 142777 000010 167206 BICB #TBIT,@CSR ;CLEAR INPUTS
2374 013002 152777 000100 167200 BISB #EBIT,@CSR
2375 013010 005037 002150 CLR YLOOP ;WAIT
    013014 005237 002150 67$: INC YLOOP
    013020 023727 002150 177777 CMP YLOOP,#-1
    013026 001372 BNE 67$
2376 013030 104005 ERROR!5 ;NO INTERRUPT
2377 013032 000137 012410 JMP 2$
2378
2379 013036 022626 10$: CMP (SP)+,(SP)+ ;RESET STACK POINTER
2380 013040 104005 ERROR!5 ;NO INTERRUPT
2381 013042 000137 012410 JMP 2$
```

```
2384 .SBTTL DIGITAL MODULE TEST M5012,M5012YA
2385
2386 ;:*****
2387
2388 ;THIS TEST CHECKS M5012 MODULE
2389 ;ISOLATED 16 BIT DC INPUT
2390
2391 ;:*****
2392
2393 M5012:
2394 ;:*****
2395 013046 000004 000020 001206 TST20: SCOPE
2396 013050 012737 000020 167124 MOV #20,$TESTN
2397 013056 152777 000020 167124 BISB #DBIT,@CSR
2398 013064 112737 000000 001140 MOVB #0,$GDDAT ;SET WHAT DATA SHOULD BE
2399 013072 012737 000002 002170 MOV #2,BYTNUM ;SET NUMBER OF BYTES
2400 013100 004737 025016 JSR PC,TSTBYT ;CHECK IF DATA IS CORRECT
2401
2402 013104 152777 000010 167076 BISB #TBIT,@CSR
2403 013112 112737 000377 001140 MOVB #377,$GDDAT ;SET WHAT DATA SHOULD BE
2404 013120 012737 000002 002170 MOV #2,BYTNUM ;SET NUMBER OF BYTES
2405 013126 004737 025016 JSR PC,TSTBYT ;CHECK IF DATA IS CORRECT
2406
2407 013132 142777 000010 167050 BICB #TBIT,@CSR
2408 013140 004737 025414 JSR PC,CLRINT
2409 013144 012746 000000 MOV #PRO,-(SP) ;SET PSW TO PRIORITY 0
2410 013150 012746 013156 MOV #64,-(SP)
2411 013154 000002 RTI
2412 013156 000240 64$: NOP
2413 013160 012777 013264 167026 MOV #6$,@VECTO ;SET INTERRUPT VECTOR
2414 013166 012777 000340 167022 MOV #PR7,@VECTOA
2415 013174 152777 000100 167006 BISB #EBIT,@CSR ;ENABLE INTERRUPT
2416 013202 152777 000010 167000 BISB #TBIT,@CSR ;START INTERRUPT
2417 013210 005037 002150 CLR YLOOP ;WAIT
2418 013214 005237 002150 65$: INC YLOOP
2419 013220 023727 002150 177777 CMP YLOOP,#-1
2420 013226 001372 BNE 65$
2421
2422 013230 104005 ERROR!5 ;NO INTERRUPT
2423 013232 142777 000100 166750 7$: BICB #EBIT,@CSR ;CLEAR CSR
2424 013240 142777 000010 166742 BICB #TBIT,@CSR
2425 013246 012737 005044 000004 MOV #TMOVEC,ERRVEC
2426 013254 004737 025414 JSR PC,CLRINT ;CLEAR ALL INTERRUPT
2427 013260 000004 SCOPE
2428 013262 000207 RTS PC ;RETURN TU MONITOR
2429
2430 013264 022626 6$: CMP (SP)+,(SP)+ ;RESET STACK POINTER
2431 013266 005037 002242 CLR TEMP1
2432 013272 012737 013520 000004 MOV #4$,ERRVEC
2433 013300 117737 166706 002232 3$: MOVB @IAR,TEMP ;FIND WHICH I/O INTERRUPTED
2434 013306 152777 000001 166674 BISB #RBIT,@CSR
2435 013314 123737 002232 002172 CMPB TEMP,TADDR ;IS IT MUT
2436 013322 001415 BEQ 2$ ;YES
2437 013324 113737 002225 002233 MOVB BASE+1,TEMP+1 ;ASSEMBLE ADDRESS OF INTERRUPTING I/O
2438 013332 105777 166674 TSTB @TEMP ;CLEAR INTERRUPT
2439 013336 005237 002242 10$: INC TEMP1
2440 013342 022737 000400 002242 CMP #400,TEMP1
2441 013350 001353 BNE 3$
2442 013352 104005 ERROR!5 ;NO INTERRUPT
```



```
2430 013354 000726          BR      7$
2431 013356 105777 166610    2$:    TSTB   @TADDR      ;CLEAR INTERRUPT
2432 013362 005237 002172          INC    TADDR
2433 013366 152777 000001 166614    BISB   #RBIT,@CSR
2434 013374 105777 166572          TSTB   @TADDR      ;CLEAR INTERRUPT IN NEXT BYTE
2435 013400 005337 002172          DEC    TADDR
2436 013404 132777 000200 166576    BITB   #FBIT,@CSR  ;CHECK IF INTERRUPT CLEAR
2437 013412 001413          BEQ    8$
2438 013414 117737 166572 002232    MOVB   @IAR,TEMP
2439 013422 123737 002232 002172    CMPB   TEMP,TADDR
2440 013430 001002          BNE    9$
2441 013432 104015          ERROR!15
2442 013434 000676          BR      7$
2443 013436 004737 025414    9$:    JSR    PC,CLRINT  ;CLEAR ALL REMINDING INTERRUPTS
2444 013442 132777 000010 166540    8$:    BITB   #TBIT,@CSR
2445 013450 001670          BEQ    7$
2446 013452 012746 000000          MOV    #PRO,-(SP)  ;SET PSW TO PRIORITY 0
      013456 012746 013464          MOV    #66$,-(SP)
      013462 000002          RTI
      013464 000240          NOP
2447 013466 142777 000010 166514    66$:   BICB   #TBIT,@CSR  ;CHECK IF CLEARING T BIT CAUSE INTERRUPT
2448 013474 005037 002150          CLR    YLOOP      ;WAIT
      013500 005237 002150          INC    YLOOP
      013504 023727 002150 177777    67$:   CMP    YLOOP,#-1
      013512 001372          BNE    67$
2449 013514 104005          ERROR!5
2450 013516 000645          BR      7$
      2451
2452 013520 022626    4$:    CMP    (SP)+,(SP)+ ;RESET STACK
2453 013522 000705          BR      10$
2454
```

```
2456 .SBTTL MONITOR TO SELECT DAC TESTS
2457 013524 DCAMON: MOV #KBINT,@KBVEC
013524 012777 025556 166550 BISB #BIT6,@$TKS ;ENABLE KB INTERRUPT
013532 152777 000100 165420 JSR PC,GTADRS ;SET UP ADDRESSES FOR DAC
2458 013540 004737 014036 JSR PC,TSTADR ;CHECK PRESENCE OF ADDRESSES
2459 013544 004737 014132 RTS PC
2460 013550 000207 JSR PC,DDBIT ;GO DO D-BIT TEST
2461 013552 004737 014100 RTS PC ;RETURN TO AUTO MONITOR
2462 013556 000207
2463
2464 013560 012737 013560 003060 DACSTR: MOV #DACSTR,RETURN ;SET RETURN FOR CONTROL A
2465 013566 104401 036415 TYPE ,MASS13 ;ASK ADDRESS
2466 013572 104410 RDOCT
2467 013574 012637 002172 MOV (SP)+,TADDR ;;POP STACK INTO TADDR
2468 013600 023737 002172 002224 CMP TADDR,BASE
2469 013606 103764 BLO DACSTR
2470 013610 004737 014036 DACMON: JSR PC,GTADRS ;SET UP ADDRESSES
2471 013614 004737 014132 JSR PC,TSTADR ;CHECK RESPONSE OF ADDRESSES
2472 013620 000207 DCOUT: RTS PC ;EXIT TO MONITOR ON ERROR
2473 013622 004737 016504 JSR PC,DAGTST ;CHECK THAT GENERIC CODE OF
2474 013626 104401 042367 TYPE ,DANGC ;THE DAC IS 261 IF IT IS GO TEST
2475 013632 000207 RTS PC ;ELSE RETURN TO MAIN MONITER
2476 013634 104401 042523 DCMN: TYPE ,DANSEL ;SELECT TEST A, T, D,
2477 013640 004737 016652 JSR PC,KLEER
2478 013644 104406 RDCHR
2479 013646 012637 002360 MOV (SP)+,ANSW ;;POP STACK INTO ANSW
2480 013652 104401 002360 TYPE ,ANSW
2481 013656 122737 000110 002360 CMPB #'H,ANSW ;IF QUESTION THE ELIGHTEN
2482 013664 001003 BNE 1$ ;USER BY TYPING TEXT OF TEST
2483 013666 104401 042631 TYPE ,TEXT ;SELECTION DESCRIPTORS
2484 013672 000760 BR DCMN ;NOW SELEC TEST
2485 013674 122737 000101 002360 1$: CMPB #'A,ANSW ;SELECT DAC CALIBRATION TEST
2486 013702 001015 BNE 2$ ;IF NOT THIS TEST THEN CHECK NEXT
2487 013704 104401 034334 TYPE ,M30 ;DISCONNECT CUSTOMER WIRES
2488 013710 104406 5$: RDCHR
2489 013712 012637 002360 MOV (SP)+,ANSW ;;POP STACK INTO ANSW
2490 013716 122737 000015 002360 CMPB #15,ANSW
2491 013724 001371 BNE 5$
2492 013726 004737 014516 JSR PC,DACTST ;ELSE DO THIS TEST
2493 013732 000137 013634 JMP DCMN
2494 013736 122737 000124 002360 2$: CMPB #'T,ANSW ;SELECT TBIT (STATUS BITS) TEST
2495 013744 001016 BNE 3$ ;IF NOT THIS TEST THEN CHECK NEXT
2496 013746 104401 034334 TYPE ,M30
2497 013752 104406 6$: RDCHR
2498 013754 012637 002360 MOV (SP)+,ANSW ;;POP STACK INTO ANSW
2499 013760 122737 000015 002360 CMPB #15,ANSW
2500 013766 001371 BNE 6$
2501 013770 004737 027060 14$: JSR PC,SWLOOP
2502 013774 015256 INTCOM ;DO INTCOM TEST
2503 013776 000137 013634 JMP DCMN
2504 014002 122737 000104 002360 3$: CMPB #'D,ANSW ;SELECT DBIT TEST
2505 014010 001004 BNE 4$ ;IF NOT THIS TEST THEN CHECK NEXT
2506 014012 004737 014100 9$: JSR PC,DDBIT ;DO DDBIT TEST
2507 014016 000137 013634 JMP DCMN
2508 014022 004737 025502 4$: JSR PC,CNTRC ;CHECK FOR EXIT BY CONTROL 'C'
2509 014026 104401 033661 TYPE ,M20 ;TYPE ? IF NO TST SELECTION FOUND
2510 014032 000137 013634 JMP DCMN ;GO TO START OF DAC MONITOR
```



```

2511
2512
2513
2514 014036 005000          GTADRS: CLR    R0          ;CLEAR ADDRESS POINTER
2515 014040 005001          CLR    R1          ;CLEAR ADDRESS COUNTER
2516 014042 013746 002172  MOV    TADDR,-(SP)  ;;PUSH TADDR ON STACK
2517 014046 013760 002172 002330 1$:  MOV    TADDR,LDATA0(R0) ;WRITE ADDRESSES
2518 014054 105237 002172  INCB   TADDR          ;UPDATE TO NEXT REGISTER
2519 014060 005720          TST    (R0)+       ;UPDATE TO NEXT LOCATION
2520 014062 005201          INC    R1          ;UPDATE REGISTER COUNTER
2521 014064 022701 000010  CMP    #10,R1      ;CHECK FOR EIGHT REGISTERS
2522 014070 001366          BNE   1$          ;IF NOT LAST GET NEXT
2523 014072 012637 002172  MOV    (SP)+,TADDR ;;POP STACK INTO TADDR
2524 014076 000207          RTS    PC          ;AND RETURN
2525
2526          .SBTTL D-BIT TEST
2527
2528 014100 004737 016652  DDRBIT: JSR    PC,KLEER
2529 014104 152777 000020 166076  BISB  #DBIT,@CSR   ;SET THE D-BIT
2530 014112 004737 014224  JSR    PC,TSTRGS   ;GO CHECK REGISTERS
2531 014116 142777 000020 166064  BICB  #DBIT,@CSR   ;CLEAR D-BIT WHEN DONE
2532 014124 004737 016652  JSR    PC,KLEER    ;AND CLEAR THE WORLD
2533 014130 000207          RTS    PC          ;AND RETURN
2534

```

```
2536 .SBTTL REGISTER VERIFICATION
2537 014132 013703 000004 000004 TSTADR: MOV ERRVEC,R3 ;SAVE PRESENT LOC 4
2538 014136 012737 014170 000004 MOV #2$,ERRVEC ;SET UP TIME OUT VECTOR
2539 014144 004737 016652 JSR PC,KLEER ;INIT THE SYSTEM
2540 014150 005000 CLR R0 ;CLR ADDRESS LOC
2541 014152 005001 CLR R1 ;CLR ADDRESS COUNTER
2542 014154 013700 002172 MOV TADDR, R0 ;GET FIRST REG ADDRESS
2543 014160 105720 1$: TSTB (R0)+ ;IS IT THERE AND CLEAR
2544 014162 001407 BEQ 3$ ;IF IT IS GET NEXT ADDRESS
2545 014164 104027 ERROR!27 ;ELSE TYPE NOT ZERO
2546 014166 000405 BR 3$ ;AND THEN GET NEXT ADDRESS
2547 014170 104026 2$: ERROR!26 ;IF I CAME HERE I TIMED OUT
2548 014172 004737 025502 JSR PC,CNTRC ;CHECK FOR CONTROL 'C' EXIT
2549 014176 022626 CMP (SP)+,(SP)+ ;ADJUST THE STACK AFTER TIME OUT
2550 014200 000406 BR 4$ ;RETURN MAIN MONITOR
2551 014202 005201 3$: INC R1 ;UPDATE ADDRESS COUNTER
2552 014204 022701 000010 CMP #10,R1 ;CHECK FOR LAST REG
2553 014210 001363 BNE 1$ ;IF IT IS THEN
2554 014212 062716 000002 ADD #2,(SP) ;CONTINUE WITH TEST
2555 014216 010337 000004 4$: MOV R3,ERRVEC ;RESTORE LOC4
2556 014222 000207 RTS PC ;RETURN
2557
2558 014224
```

```
TSTRGS:
:*****
TST21: SCOPE
2559 014224 000004 000021 001206 MOV #21,$TESTN
2560 014234 005037 002762 CLR ROTPAT ;CLEAR THE PATRN SPOT
2561 014240 005000 CLR R0 ;CLEAR THE INDEX REGISTER
2562 014242 012737 000007 002350 MOV #7,KOUNT ;DO SEVEN FOR NOW
2563 014250 004737 016452 1$: JSR PC,ROTDAT ;NOW GO ROTATE A BIT
2564 014254 113770 002762 002330 2$: MOVB ROTPAT,@LDATA0(R0) ;NOW GO WRITE IT
2565 014262 004737 014476 JSR PC,UPREG ;NOW GET THE NEXT REGISTER
2566 014266 000772 2$ BR ;AND WRITE IN (4 IN ALL)
2567 014270 013737 002762 001140 MOV ROTPAT,$GDDAT ;SAVE PATTERN FOR CHECKING
2568 014276 117037 002330 001142 3$: MOVB @LDATA0(R0),$BDDAT ;READ THE RSGISTER
2569 014304 016037 002330 002232 MOV LDATA0(R0),TEMP ;AND STORE ITS ADDRESS
2570 014312 023737 001140 001142 CMP $GDDAT,$BDDAT ;SEE IF IT IS GOOD
2571 014320 001401 BEQ 4$ ;IF GOOD CONTINUE
2572 014322 104030 ERROR!30 ;ELSE ERROR
2573 014324 004737 025502 4$: JSR PC,CNTRC ;CHECK FOR CONTROL 'C' EXIT
2574 014330 004737 014476 JSR PC,UPREG ;READ NEXT REGISTER
2575 014334 000760 BR 3$ ;UNTIL 4 DONE
2576 014336 005337 002350 DEC KOUNT ;AND UNTIL7 ROTATES
2577 014342 100342 BPL 1$ ;NOW GO BACK AND DO IT
2578 014344 005037 002762 CLR ROTPAT ;CLEAR PATTERN SPOT
2579 014350 012737 000001 002350 MOV #1,KOUNT ;DO ONLY 2 THIS TIME
2580 014356 004737 016452 5$: JSR PC,ROTDAT ;GO ROTATE A BIT NOW
2581 014362 113770 002762 002332 6$: MOVB ROTPAT,@HDATA0(R0) ;GO WRITE IT NOW
2582 014370 004737 014476 JSR PC,UPREG ;GET THE NEXT REGISTER
2583 014374 000772 BR 6$ ;AND WRITE IN IT (4 IN ALL)
2584 014376 013737 002762 001140 MOV ROTPAT,$GDDAT ;SAVE PATTERN FOR CHECKING
2585 014404 112770 000000 002330 7$: MOVB #0,@LDATA0(R0) ;NOW LET IT TRANSFER
2586 014412 117037 002332 001142 MOVB @HDATA0(R0),$BDDAT ;READ THE REGISTER
2587 014420 016037 002332 002232 MOV HDATA0(R0),TEMP ;AND STORE ITS ADDRESS
2588 014426 123737 001140 001142 CMPB $GDDAT,$BDDAT ;SEE IF IT IS GOOD
2589 014434 001401 BEQ 8$ ;IF GOOD CONTINUE
2590 014436 104030 ERROR!30 ;ELSE ERROR
```



```
2591 014440 004737 025502      8$:   JSR   PC,CNTRC      ;CHECK FOR CONTROL 'C' EXIT
2592 014444 004737 014476      JSR   PC,UPREG      ;READ NEXT REGISTER
2593 014450 000755                BR    7$            ;UNTIL 4 ARE DONE
2594 014452 005337 002350      DEC   KOUNT        ;AND FOR 2 ROTATES
2595 014456 100337                BPL   5$            ;GO BACK AND DO IT
2596 014460 000004      SCOPE
2597 014462 005737 002240      TST   AFLAG
2598 014466 001002      BNE   9$
2599 014470 104401 036474      TYPE  ,MASS15      ;END OF TEST
2600 014474 000207      9$:   RTS   PC
2601
2602
2603
2604
2605 014476 022020      UPREG:  CMP   (R0)+,(R0)+      ;UPDATE INDEX
2606 014500 022700 000020      CMP   #20,R0      ;UNTIL 4 REGISTERS
2607 014504 001003      BNE   1$            ;ARE DONE
2608 014506 005000      CLR   R0           ;CLR THE INDEX REG
2609 014510 062716 000002      ADD   #2,(SP)     ;UPDATE THE STACK
2610 014514 000207      1$:   RTS   PC      ;CONTINUE TEST
2611
2612
2613
```

```
.SBTTL DAC CALIBRATION TEST
2615
2616
2617
2618 014516 004737 016652 DACTST: JSR PC,KLEER
2619
2620 014522 004737 016550 DATST: JSR PC,DECCHN ;GET CHANEL # AND DECODE IT
2621 014526 104401 042737 TYPE ;DO REFER VOLTAGE CAL Y-N
2622 014532 104406 RDCHR ;GET Y-N ANSW
2623 014534 012637 002360 MOV (SP)+,ANSW ;;POP STACK INTO ANSW
2624 014540 104401 002360 TYPE ;ANSW ;ECHO IT
2625 014544 122737 000116 002360 CMPB #'N,ANSW ;IF IT IS A NO
2626 014552 001413 BEQ 5$ ;GO TO CHAN AND
2627 014554 122737 000131 002360 CMPB #'Y,ANSW ;IS IT A YES?
2628 014562 001403 BEQ 20$
2629 014564 104401 033661 TYPE ;M20 ;?
2630 014570 000754 BR DATST
2631 014572 004737 025502 20$: JSR PC,CNTRC
2632 014576 004737 014614 JSR PC,DCTST ;PASS REFER VOLT CAL TST
2633 014602 004737 025502 5$: JSR PC,CNTRC
2634 014606 104401 042223 TYPE ;M81
2635 014612 000411 BR DTST
2636
2637 014614 104401 043133 DCTST: TYPE ;CM1 ;ADJ R135 @ PIN 6 OF E82 FOR 10.24
2638 014620 004737 025662 JSR PC,CRTST ;USING SWITCHES AT E79 IF NEEDED
2639 ;WHEN DONE HIT <CR>
2640 014624 104401 043202 TYPE ;CM2 ;ADJ R134 @ PIN 60 OF E81 FOR 5.12
2641 014630 004737 025662 JSR PC,CRTST ;WHEN DONE HIT <CR>
2642 014634 000207 RTS PC
2643 014636 013700 002322 DTST: MOV XCHAN,RO ;GET REG INDEX TO CHAN
2644 014642 104401 043252 TYPE ;CM3 ;ADJ VOLTAGE OFFSET ON CHAN 1 TO 0.000
2645 014646 004737 025662 21$: JSR PC,CRTST ;WAIT FOR <CR>
2646 014652 112770 000003 002332 1$: MOVB #3,@HDATA0 (RO) ;LOAD ALL ONES INTO
2647 014660 112770 000377 002330 MOVB #377,@LDATA0 (RO) ;CHAN SELECTED
2648 014666 104401 043322 TYPE ;CM4 ;CHECK VOLTAGE LEVEL ON CHANNEL SELECTED
2649 014672 004737 025662 22$: JSR PC,CRTST ;FOR 10.23V + OR - 10 MIN
2650 014676 104401 042262 TYPE ;M82
2651 014702 104406 41$: RDCHR
2652 014704 012637 002360 MOV (SP)+,ANSW ;;POP STACK INTO ANSW
2653 014710 104401 002360 TYPE ;ANSW
2654 014714 004737 025502 JSR PC,CNTRC
2655 014720 122737 000116 002360 CMPB #'N,ANSW
2656 014726 001002 BNE 40$
2657 014730 000137 015244 JMP 7$
2658 014734 122737 000131 002360 40$: CMPB #'Y,ANSW
2659 014742 001357 BNE 41$
2660 014744 104401 044121 TYPE ;CM12 ;ASK IF 4 TO 20 MA RANGE WANTED
2661 014750 104406 23$: RDCHR ;READ ANSWER
2662 014752 012637 002360 MOV (SP)+,ANSW ;;POP STACK INTO ANSW
2663 014756 104401 002360 TYPE ;ANSW ;ECHO ANSWER
2664 014762 122737 000131 002360 CMPB #'Y,ANSW ;IF YES GO TO 4 TO 20 MA TEST
2665 014770 001455 BEQ 5$ ;ELSE DO 0 TO 20 MA TEST
2666 014772 122737 000116 002360 CMPB #'N,ANSW
2667 015000 001403 BEQ 4$
2668 015002 004737 025502 JSR PC,CNTRC
2669 015006 000760 BR 23$
2670 015010 112770 000003 002332 4$: MOVB #3,@HDATA0 (RO) ;RELOAD UPPER BYTE OF CHANEL
2671 015016 112770 000377 002330 MOVB #377,@LDATA0 (RO) ;SELECTED AND LET IT GO (CONV)
```



```
2727 015336 042701 177703      BIC      #177703,R1      ;CLEAR UNWANTED BITS
2728 015342 022701 000074      CMP      #74,R1        ;IF ALL BIT ARE SET
2729 015346 001417                BEQ      5$            ;CONTINUE WITH TEST
2730
2731 015350 010137 002232      MOV      R1,TEMP
2732 015354 006237 002232      ASR      TEMP
2733 015360 006237 002232      ASR      TEMP
2734 015364 012737 000017 002770  MOV      #17,TEMP3
2735 015372 104031                ERROR!31
2736 015374 004737 016652      JSR      PC,KLEER
2737 015400 000207                RTS      PC
2738 015402 004737 025502      JSR      PC,CNTRC
2739 015406 062700 000002 5$:      ADD      #2,R0
2740 015412 023700 002346      CMP      HDATA3,R0
2741 015416 003746                BLE      4$
2742 015420 005000                CLR      R0
2743 015422 004737 016652      JSR      PC,KLEER
2744 015426 152777 000010 164554  BISB    #TBIT,@CSR
2745 015434 112777 000000 164666  MOVB    #0,@LDATA0
2746 015442 112777 000004 164664  MOVB    #4,@LDATA1
2747 015450 112777 000010 164662  MOVB    #10,@LDATA2
2748 015456 112777 000014 164660  MOVB    #14,@LDATA3
2749 015464 004737 016730      JSR      PC,DALLY
2750 015470 117702 164636      MOVB    @HDATA0,R2
2751 015474 042702 177703      BIC      #177703,R2
2752 015500 022702 000074      CMP      #74,R2
2753 015504 001415                BEQ      6$
2754 015506 010237 002232      MOV      R2,TEMP
2755 015512 006237 002232      ASR      TEMP
2756 015516 006237 002232      ASR      TEMP
2757 015522 012737 000017 002770  MOV      #17,TEMP3
2758 015530 104031                ERROR!31
2759 015532 004737 016652      JSR      PC,KLEER
2760 015536 000207                RTS      PC
2761 015540 004737 025502 6$:      JSR      PC,CNTRC
2762 015544 004737 016652      JSR      PC,KLEER
2763 015550 152777 000010 164432  BISB    #TBIT,@CSR
2764 015556 112777 000014 164544  MOVB    #14,@LDATA0
2765 015564 112777 000010 164542  MOVB    #10,@LDATA1
2766 015572 112777 000004 164540  MOVB    #4,@LDATA2
2767 015600 112777 000000 164536  MOVB    #0,@LDATA3
2768 015606 004737 016730      JSR      PC,DALLY
2769 015612 117702 164514      MOVB    @HDATA0,R2
2770 015616 042702 177703      BIC      #177703,R2
2771 015622 005702                TST      R2
2772 015624 001414                BEQ      7$
2773 015626 010237 002232      MOV      R2,TEMP
2774 015632 006237 002232      ASR      TEMP
2775 015636 006237 002232      ASR      TEMP
2776 015642 005037 002770      CLR      TEMP3
2777 015646 104032                ERROR!32
2778 015650 004737 016652      JSR      PC,KLEER
2779 015654 000207                RTS      PC
2780 015656 004737 025502 7$:      JSR      PC,CNTRC
2781
2782 015662 012737 000000 002352  MOV      #0,ACOUNT
2783 015670 012737 000020 002354  MOV      #20,BCOUNT
;SET UP 0&10 INPUT TO
;CHANELS 0 TO 3 I.E. 0,20,0,20
```



```

2784 015676 012737 000064 002306      MOV      #64,CH0      ;GET GOOD STATUS OUTPUT
2785 015704 012737 000040 002310      MOV      #40,CH1      ;GET ERROR
2786 015712 012737 000070 002312      MOV      #70,CH2      ;STATUS OUTPUTS
2787 015720 012737 000054 002314      MOV      #54,CH3      ;TOTAL OF FOUR
2788 015726 012737 000004 002316      MOV      #4,CH4       ;TO BE LOOKED AT
2789 015734 152777 000010 164246      BISB     #TBIT,@CSR   ;SET THE TBIT IN THE CSR
2790 015742 004737 016214          8$:      JSR      PC,INTLD     ;LOAD INPUT REGS
2791 015746 117737 164360 002232      MOVVB   @HDATA0,TEMP ;GET STATUS BITS
2792 015754 042737 177703 002232      BIC     #177703,TEMP  ;CLEAR UNWANTED BITS
2793 015762 013737 002232 002770      MOV     TEMP,TEMP3
2794 015770 006237 002770      ASR     TEMP3
2795 015774 006237 002770      ASR     TEMP3
2796 016000 004737 016302      JSR     PC,CKICT     ;GO CHECK STATUS BITS
2797 016004 104033          ERROR!33 ;REPORT ERROR HERE
2798 016006 004737 025502      JSR     PC,CNTRC     ;CHECK OFR CONTROL 'C' EXIT
2799 016012 004737 016416      JSR     PC,GOUP      ;GET NEXT INPUT VALUES
2800 016016 022737 001750 002352      CMP     #1750,ACOUNT ;THIS IS THE LAST INPUT
2801 016024 003746          BLE     8$          ;IF NOT GET NEXT INPUT
2802
2803 016026 012737 001770 002352      MOV     #1770,ACOUNT ;SET UP UPPER LIMIT TO DO
2804 016034 012737 001760 002354      MOV     #1760,BCOUNT ;STATUS BIT COMPLIMENT DOWN
2805 016042 012737 000010 002306      MOV     #10,CH0      ;GET GOOD STATUS OUTPUT
2806 016050 012737 000034 002310      MOV     #34,CH1      ;GET STATUS
2807 016056 012737 000004 002312      MOV     #4,CH2       ;ERROR OUTPUTS
2808 016064 012737 000020 002314      MOV     #20,CH3      ;TOTAL OF FOUR
2809 016072 012737 000070 002316      MOV     #70,CH4      ;TO BE LOOKED AT
2810 016100 004737 016214          9$:      JSR     PC,INTLD     ;LOAD INPUT REGS
2811 016104 117737 164222 002232      MOVVB   @HDATA0,TEMP ;GET STATUS BITS
2812 016112 042737 177703 002232      BIC     #177703,TEMP  ;CLEAR UNWANTED BITS
2813 016120 013737 002232 002770      MOV     TEMP,TEMP3
2814 016126 006237 002770      ASR     TEMP3
2815 016132 006237 002770      ASR     TEMP3
2816 016136 004737 016302      JSR     PC,CKICT     ;GO VERIFY DATA
2817 016142 104033          ERROR!33
2818 016144 004737 025502      JSR     PC,CNTRC     ;CHECK CONTROL 'C' EXIT
2819 016150 004737 016434      JSR     PC,GODN      ;GO COUNT DOWN
2820 016154 005737 002354      TST     BCOUNT       ;CHECK FOR LAST INPUT
2821 016160 001347          BNE     9$          ;IF NOT LAST GET NEXT INPUT
2822 016162 000004          SCOPE
2823 016164 142777 000010 164016      BICB    #TBIT,@CSR   ;CLEAR THE TBIT AND
2824 016172 004737 016652          JSR     PC,KLEER     ;CLEAR THE EMPIRE
2825 016176 032777 020000 162750      BIT     #BIT13,@SWR  ;CHECK INHIBIT PRINTOUT
2826 016204 001002          BNE     10$
2827 016206 104401 036474          TYPE   ,MASS15      ;TYPE FINISHED
2828 016212 000207          10$:     RTS     PC
2829
2830
2831 016214 113777 002353 164110      INTLD:  MOVVB  ACOUNT+1,@HDATA0 ;LOAD UPPER CH0 REG
2832 016222 113777 002352 164100      MOVVB  ACOUNT,@LDATA0 ;LOAD LOWER CH0 REG
2833 016230 113777 002355 164100      MOVVB  BCOUNT+1,@HDATA1 ;LOAD UPPER CH1 REG
2834 016236 113777 002354 164070      MOVVB  BCOUNT,@LDATA1 ;LOAD LOWER CH1 REG
2835 016244 113777 002353 164070      MOVVB  ACOUNT+1,@HDATA2 ;LOAD UPPER CH2 REG
2836 016252 113777 002352 164060      MOVVB  ACOUNT,@LDATA2 ;LOAD LOWER CH2 REG
2837 016260 113777 002355 164060      MOVVB  BCOUNT+1,@HDATA3 ;LOAD UPPER CH3 REG
2838 016266 113777 002354 164050      MOVVB  BCOUNT,@LDATA3 ;LOAD LOWER CH3 REG
2839 016274 004737 016730      JSR     PC,DALLY     ;WAIT FOR CONVERSION
2840 016300 000207          RTS     PC          ;RETURN

```

```

2841
2842 016302 023737 002306 002232 CKICT:  CMP      CHO,TEMP      ;CHECK GOOD DATA
2843 016310 001003          BNE      1$          ;IF BAD CHECK WHIC CHANEL
2844 016312 062716 000002          ADD      #2,(SP)     ;ELSE BY PASS ERROR
2845 016316 000207          RTS      PC          ;RETURN
2846 016320 005037 002304          1$:      CLR      CHNUM     ;GET CHO
2847 016324 023737 002310 002232          CMP      CH1,TEMP   ;SEE IF THIS CHANEL IS BAD
2848 016332 001430          BEQ      2$          ;IF YES GO REPORT IT ELSE
2849 016334 012737 000001 002304          MOV      #1,CHNUM   ;GET CH1
2850 016342 023737 002312 002232          CMP      CH2,TEMP   ;SEE IF THIS CHANEL IS BAD
2851 016350 001421          BEQ      2$          ;IF YES GO REPORT IT ELSE
2852 016352 012737 000002 002304          MOV      #2,CHNUM   ;GET CH2
2853 016360 023737 002314 002232          CMP      CH3,TEMP   ;SEE IF THIS CHANEL IS BAD
2854 016366 001412          BEQ      2$          ;IF YES GO REPORT IT ELSE
2855 016370 012737 000003 002304          MOV      #3,CHNUM   ;GET CH3
2856 016376 023737 002316 002232          CMP      CH4,TEMP   ;SEE IF THIS CHANEL IS BAD
2857 016404 001403          BEQ      2$          ;IF YES GO REPORT IT ELSE
2858 016406 104034          ERROR!34 ;TELL OF MULTIPLE UNDEFINED ERRORS
2859 016410 062716 000002          ADD      #2,(SP)     ;UPDATE STACK AND RETURN
2860 016414 000207          2$:      RTS      PC          ;TO CONTINUE TEST
2861
2862 016416 062737 000010 002352 GOUP:   ADD      #10,ACOUNT ;GO UP TO 1760 BY
2863 016424 062737 000010 002354          ADD      #10,BCOUNT ;INCREMENT OF TEN
2864 016432 000207          RTS      PC
2865
2866 016434 162737 000010 002352 GODN:   SUB      #10,ACOUNT ;GO DOWN TO ZERO BY
2867 016442 162737 000010 002354          SUB      #10,BCOUNT ;INVERSE INCREMENT OF TEN
2868 016450 000207          RTS      PC          ;RETURN
2869
2870 016452 005237 002764          ROTDAT: INC      ROTFLG ;CLEAR THIS FLAG ON FIRST ENTRY
2871 016456 001001          BNE      1$          ;CHECK IF ZERO FIRST TIME
2872 016460 000261          SEC      ;FIRST TIME IN SET CARRY
2873 016462 106137 002762          1$:      ROLB     ROTPAT   ;ROTATE PATTERN ONCE
2874 016466 005737 002350          TST      KOUNT      ;DID IT GET DONE 8 TIMES
2875 016472 001003          BNE      2$          ;IF WE HAVE SET ROTFLG
2876 016474 012737 177777 002764          MOV      #-1,ROTFLG ;BACK TO MINUS ONE
2877 016502 000207          2$:      RTS      PC          ;NOW USE DATA
2878
2879 016504 004737 016652          DAGTST: JSR      PC,KLEER   ;CLEAR IT ALL
2880 016510 152777 000004 163472          BISB     #GBIT, @CSR ;SET THE GENERIC BIT IN THE CSR
2881 016516 117700 163606          MOVVB    @LDATA0,RO ;READ THE GENERIC CODE IN DAC
2882 016522 042700 177400          BIC      #177400,RO ;CLEAR THE UPPER BYTE
2883 016526 022700 000261          CMP      #261,RO    ;CHECK THAT GENERIC CODE IS 261
2884 016532 001002          BNE      1$          ;IF NOT RETURN ERR MESS AND EXIT
2885 016534 012716 013634          MOV      #DCMN, (SP) ;ELSE CONTINUE TO TEST
2886 016540 142777 000004 163442          1$:      BICB     #GBIT, @CSR ;BUT FIRST CLEAR GBIT
2887 016546 000207          RTS      PC
2888
2889
2890 016550 104401 042563          DECCHN: TYPE     ,DACHN ;ASK FOR CHANEL NUMBER (0,1,2,3)
2891 016554 104410          RDOCT    ;GET CHANEL NUMBER
2892 016556 012637 002360          MOV      (SP)+,ANSW ;POP STACK INTO ANSW
2893 016562 022737 000004 002360          CMP      #4,ANSW   ;CHECK FOR RIGHT CHAN
2894 016570 003003          BGT      1$
2895 016572 104401 033661          TYPE     ,:20      ;?
2896 016576 000764          BR       DECCHN
2897 016600 013737 002360 002320          1$:      MOV      ANSW,DCHAN ;GET DAC CHANEL SELECTED

```


2898	016606	000241			CLC				
2899	016610	006337	002360		ASL	ANSW			:NO END AROUND CARRY WANTED
2900	016614	013737	002360	002322	MOV	ANSW,XCHAN			:CONVERT TO VCHAN SELECTION
2901	016622	006337	002322		ASL	XCHAN			:GET ANSW TO INDEX
2902	016626	013737	002360	002324	MOV	ANSW,VCHAN			:MULTIPLY BY 2 AGAIN
2903	016634	062737	000001	002360	ADD	#1,ANSW			:STORE SELECTED VCHAN
2904	016642	013737	002360	002326	MOV	ANSW,ICHAN			:CONVERT TO ICHAN SELECTED
2905	016650	000207			RTS	PC			:STORE SELECTE ICHAN
2906									:RETURN
2907	016652								
	016652	152777	000002	163330	KLEER:	BISB	#CBIT,@CSR		:SET C BIT
	016660	005037	002150			CLR	YLOOP		:WAIT
	016664	005237	002150		64\$:	INC	YLOOP		
	016670	023727	002150	000007		CMP	YLOOP,#7		
	016676	001372				BNE	64\$		
	016700	152777	000002	163302		BISB	#CBIT,@CSR		:DO IT AGAIN
	016706	005037	002150			CLR	YLOOP		:WAIT
	016712	005237	002150		65\$:	INC	YLOOP		
	016716	023727	002150	000007		CMP	YLOOP,#7		
	016724	001372				BNE	65\$		
2908	016726	000207			RTS	PC			:AND RETURN
2909									
2910	016730				DALLY:				
	016730	005037	002150			CLR	YLOOP		:WAIT
	016734	005237	002150		64\$:	INC	YLOOP		
	016740	023727	002150	000200		CMP	YLOOP,#200		
	016746	001372				BNE	64\$		
2911	016750	000207			RTS	PC			
2912									
2913	016752	000137	004546		EXERC:	JMP	MONIT		

```
2915 016756          F5010:
2916 016756          F5011:
2917 016756 012737 000004 002170      MOV    #4, BYTNUM      ;# OF BYTES PER MODULE
2918 016764 004737 025756              JSR    PC, MONDAT     ;SUBROUTINE TO MONITOR DATA CONTINUOUSLY
2919
2920 016770          F5012:
2921 016770 012737 000002 002170      MOV    #2, BYTNUM      ;# OF BYTES PER MODULE
2922 016776 004737 025756              JSR    PC, MONDAT
2923
2924 017002          F5013:
2925 017002 012737 000001 002170      MOV    #1, BYTNUM      ;# OF BYTES PER MODULE
2926 017010 004737 025756              JSR    PC, MONDAT
2927
2928 017014          F6010:
2929 017014 012737 000004 002170      MOV    #4, BYTNUM;MODULE IS 4 BYTE LONG
2930 017022 012700 000004              MOV    #4, R0 ;SET A BYTE COUNTER
2931 017026 004737 025706              JSR    PC, SETPTN
2932 017032 000207                      RTS    PC
2933
2934 017034          F6012:
2935 017034          F6013:
2936 017034 012737 000001 002170      MOV    #1, BYTNUM
2937 017042 004737 025706              JSR    PC, SETPTN     ;ROUTINE TO OUTPUT ANY PATTERN TOOUT MODULE
2938 017046 000207                      RTS    PC
2939
2940 017050          F6011:
2941 017050 012737 000002 002170      MOV    #2, BYTNUM      ;MODULE IS 2 BYTE LONG
2942 017056 004737 025706              JSR    PC, SETPTN
2943 017062 000207                      RTS    PC
```


2945
2946
2947
2948
2949
2950
2951
2952 017064
017064 012777 025556 163210
017072 152777 000100 162060
2953 017100
2954 017102 004737 026652
2955 017106 005705
2956 017110 001017
2957 017112 152777 000001 163070
2958 017120 105077 163654
2959 017124 105077 163652
2960 017130 004737 027026
2961 017134 004737 020452
2962 017140 004737 022100
2963 017144 004737 023152
2964 017150 000207
2965
2966
2967
2968 017152
017152 012777 025556 163122
017160 152777 000100 161772
2969 017166 004737 016652
2970 017172 012737 017222 003060
2971 017200 104401 036415
2972 017204 104410
2973 017206 012637 002172
2974 017212 023737 002172 002224
2975 017220 103754
2976 017222 005005
2977 017224 004737 026652
2978 017230 005705
2979 017232 001347
2980 017234 152777 000001 162746
2981 017242 105077 163532
2982 017246 004737 027026
2983
2984
2985 017252 104401 040370
2986 017256 012737 000014 002166
2987 017264 105077 163510
2988 017270 105077 163506
2989 017274 104406
2990 017276 012637 002360
2991 017302 104401 002360
2992 017306 104401 035646
2993 017312 004737 025502
2994 017316 122737 000103 002360
2995
2996 017324 001003
2997 017326 004737 017560

.SBITL A/D MONITOR

:THIS IS MONITOR FOR FOR AUTO MODE FOR A/D
ADTST:

MOV #KBINT,@KBVEC
BISB #BIT6,@\$TKS ;ENABLE KB INTERRUPT
CLR R5
JSR PC,ADADDR ;SET ADDRESSES AND CHECK REG.
TST R5 ;ADDRESS ERROR?
BNE 1\$
BISB #RBIT,@CSR
CLRB @STAT1
CLRB @STAT2
JSR PC,GCODE ;FIND IF SE/DIFF MODE
JSR PC,ADLOG ;TEST LOGIC
JSR PC,LINEAR ;TEST LINEARITY
JSR PC,RUMP ;TEST MONOTINICITY
1\$: RTS PC

:THIS IS MONITOR FOR OPERATOR INTERVENTION TESTS

ATOD:

MOV #KBINT,@KBVEC
BISB #BIT6,@\$TKS ;ENABLE KB INTERRUPT
JSR PC,KLEER
MOV #ADRET,RETURN ;SET RETURN ADDRESS FOR CONTROL A
TYPE ,MASS13 ;WHICH ADDRESS
RDOCT
MOV (SP)+,TADDR ;:POP STACK INTO TADDR
CMP TADDR,BASE
BLO ATOD
ADRET: CLR R5
JSR PC,ADADDR ;SET ADDRESSES
TST R5 ;ADDRESS ERROR?
BNE ATOD
BISB #RBIT,@CSR
CLRB @STAT1 ;SET CHANNEL ZERO-NO MAX
JSR PC,GCODE ;FIND GENERIC CODE

ADMON:

TYPE ,MASS46 ;WHICH TEST TO RUN?
MOV #14,\$MUT ;SET MODULE TYPE FOR ERRORS
CLRB @STAT1
CLRB @STAT2
RDCHR
MOV (SP)+,ANSW ;:POP STACK INTO ANSW
TYPE ,ANSW ;ECHO
TYPE ,MASS0 ;CR,LF
JSR PC,CNTRC ;CHECK FOR CONTROL C
CMPB #'C,ANSW ;IS IT CALIBRATION-C
BNE 1\$;NO
JSR PC,ADCALB ;EXECUTE CALIBRATION

```

2998 017332 000747          BR      ADMON
2999
3000
3001 017334 122737 000104 002360 1$:  CMPB  #'D,ANSW      ;IS IT LOGIC TEST
3002 017342 001005          BNE   2$           ;NO
3003 017344 004737 027060          JSR   PC,SWLOOP
3004 017350 020452          ADLOG
3005 017352 000137 017252          JMP   ADMON      ;DO ADLOG TEST
3006
3007 017356 122737 000115 002360 2$:  CMPB  #'M,ANSW      ;IS IT MCNOTONICITY TEST-M
3008 017364 001005          BNE   9$
3009 017366 004737 027060          JSR   PC,SWLOOP
3010 017372 023152          RUMP
3011 017374 000137 017252          JMP   ADMON      ;DO RUMP TEST
3012
3013 017400 122737 000114 002360 9$:  CMPB  #'L,ANSW      ;IS IT LINEARITY TEST-L
3014 017406 001005          BNE   10$         ;NO
3015 017410 004737 027060          JSR   PC,SWLOOP
3016 017414 022100          LINEAR
3017 017416 000137 017252          JMP   ADMON      ;DO LINEAR TEST
3018
3019 017422 022737 000130 002360 10$: CMP   #'X,ANSW
3020 017430 001037          BNE   15$
3021 017432 104401 037637          TYPE  ,MASS37     ;WHAT MUX
3022 017436 104410          RDOCT
3023 017440 012637 002232          MOV   (SP)+,TEMP  ;:POP STACK INTO TEMP
3024 017444 001404          BEQ   20$         ;:NO MUX 0
3025 017446 122737 000007 002232          CMPB  #7,TEMP     ;MUX TO HIGH?
3026 017454 002003          BGE   16$
3027 017456 104401 033661          TYPE  ,M20
3028 017462 000763          BR    17$        ;?
3029 017464 006337 002232          ASL   TEMP
3030 017470 006337 002232          ASL   TEMP
3031 017474 006337 002232          ASL   TEMP
3032 017500 006337 002232          ASL   TEMP
3033 017504 006337 002232          ASL   TEMP
3034 017510 013737 002232 002152          MOV   TEMP,MXNUM
3035 017516 004737 027060          JSR   PC,SWLOOP
3036 017522 023710          MUX1      ;DO THIS TEST
3037 017524 000137 017252          JMP   ADMON
3038 017530 122737 000110 002360 15$: CMPB  #'H,ANSW
3039 017536 001004          BNE   14$
3040 017540 104401 037502          TYPE  ,MASS36     ;TYPE ALL OPTIONS
3041 017544 000137 017252          JMP   ADMON
3042
3043 017550 104401 033661          TYPE  ,M20
3044 017554 000137 017252          JMP   ADMON      ;UNKNOWN CHARACTER.
3045
3046
3047
3048
3049

```

.SBTTL CALIBRATION OF A14 AND MUX


```

3051
3052 017560 104401 037637          ADCALB: TYPE      ,MASS37          ;WHICH MUX NUMBER YOU WANT TO
3053 017564 004737 027026          JSR      PC,GCODE
3054 017570 004737 025502          JSR      PC,CNTRC
3055
3056 017574 013746 000004          MOV      ERRVEC,-(SP)          ;TEST-ZERO FOR A/D
3057 017600 012737 020422 000004    MOV      #20$,ERRVEC          ;;PUSH ERRVEC ON STACK
3058 017606 104410
3059 017610 012637 002232          RDOCT
3060 017614 122737 000007 002232    MOV      (SP)+,TEMP          ;;POP STACK INTO TEMP
3061 017622 002003
3062 017624 104401 033661          CMPB    #7,TEMP
3063 017630 000753          BGE     1$
3064
3065 017632 006337 002232          TYPE    ,M20
3066 017636 006337 002232          BR      ADCALB              ;?
3067 017642 006337 002232          ASL     TEMP
3068 017646 006337 002232          ASL     TEMP
3069 017652 006337 002232          ASL     TEMP
3070 017656 104401 037703          ASL     TEMP
3071 017662 104410          5$:   TYPE    ,MASS38          ;WHICH CHANNEL?-ONLY FOR A014
3072 017664 012637 002770          RDOCT
3073 017670 105737 002232          MOV      (SP)+,TEMP3          ;;POP STACK INTO TEMP3
3074 017674 001022          TSTB   TEMP
3075 017676 022737 000301 002776    BNE     2$
3076 017704 001007          CMP     #301,GBITE          ;IS IT DIFF MODE
3077
3078 017706 122737 000007 002770    BNE     3$
3079 017714 002130          CMPB    #7,TEMP3
3080 017716 104401 037757          BGE     4$
3081 017722 000755          TYPE    ,MASS39
3082
3083 017724 122737 000017 002770    BR      5$
3084 017732 002121          CMPB    #17,TEMP3          ;IS CHANNEL TOO HIGH?
3085 017734 104401 040022          BGE     4$
3086 017740 000746          TYPE    ,MASS40          ;OK
3087
3088 017742 113777 002232 163030    3$:   BR      5$
3089 017750 152777 000004 162232    CMPB    #17,TEMP3          ;IS CHANNEL TOO HIGH?
3090 017756 117737 163016 002766    BGE     4$
3091 017764 042737 177400 002766    TYPE    ,MASS40          ;OK
3092 017772 142777 000004 162210    BR      5$
3093 020000 122737 000342 002766    CMPB    #5$,TEMP3          ;TOO HIGH CHANNEL FOR SINGLE END
3094 020006 001007          BNE     6$
3095 020010 122737 000037 002770    BNE     6$
3096 020016 002067          CMPB    #37,TEMP3          ;HERE ONLY IF MUX-SET MUX #
3097 020020 104401 040022          BGE     4$
3098 020024 000714          TYPE    ,MASS40          ;SET GBIT
3099
3100 020026 005000          BR      5$
3101 020030 122737 000322 002766    2$:   MOV     TEMP,@STAT1
3102 020036 001007          BIS     #GBIT,@CSR
3103 020040 122737 000017 002770    MOV     @STAT1,TEMP2
3104 020046 002053          BIC    #177400,TEMP2
3105 020050 104401 037757          BIC    #GBIT,@CSR
3106 020054 000700          CMPB    #342,TEMP2
3107
          BNE     6$
          CMPB    #37,TEMP3
          BGE     4$
          TYPE    ,MASS39
          BR      5$
          CLR     R0
          CMPB    #322,TEMP2
          BNE     7$
          CMPB    #17,TEMP3
          BGE     4$
          TYPE    ,MASS39
          BR      5$
          ;SET GAIN =1
          ;IS IT A156 DIFF
          ;IS CHANNEL TOO HIGH?
          ;NO
          ;YES
    
```

```

3108 020056 122737 000323 002766 7$: CMPB #323,TEMP2 ;IS IT A157
3109 020064 001031 BNE 8$ ;NO, UNKNOWN GENERIC CODE.
3110 020066 122737 000017 002770 CMPB #17,TEMP3 ;IS CHANNEL TOO HIGH?
3111 020074 002003 BGE 9$ ;NO
3112 020076 104401 037757 TYPE ,MASS39 ;YES
3113 020102 000665 BR 5$
3114
3115 020104 104401 040075 9$: TYPE ,MASS41 ;HERE ONLY IF A157
3116 020110 104410 RDOCT ;WHICH GAIN?
3117 020112 012637 002772 MOV (SP)+,TEMP4 ;;POP STACK INTO TEMP4
3118 020116 005000 CLR R0
3119 020120 023760 002772 003010 11$: CMP TEMP4,GAINB(R0) ;FIND IF GAIN IS CORRECT
3120 020126 001413 BEQ 21$ ;DO THE CALIBRATION FOR A157
3121 020130 062700 000002 ADD #2,R0
3122 020134 005760 003010 TST GAINB(R0)
3123
3124 020140 001367 BNE 11$
3125 020142 104401 040165 TYPE ,MASS42 ;WRONG GAIN SELECTED
3126 020146 000756 BR 9$
3127
3128
3129 020150 104401 036307 8$: TYPE ,MASS11 ;UNKNOWN GENERIC CODE
3130 020154 000601 BR ADCALB
3131
3132
3133 020156 156077 003032 162616 21$: BISB GAIN(R0),@STAT2 ;SET GAIN
3134 020164 104401 042443 TYPE ,GAINMG
3135 020170 016046 003010 MOV GAINB(R0),-(SP) ;;SAVE GAINB(R0) FOR TYPEOUT
3136 020174 104402 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
3137 020176 153777 002770 162574 4$: BISB TEMP3,@STAT1
3138 020204 104401 040213 TYPE ,MASS43 ;CONNECT VOLTAGE SOURCE, TYPE CR
3139 020210 004737 025662 JSR PC,CRTST ;WAIT FOR CR
3140 020214 104401 040250 TYPE ,MASS44 ;TYPE HEADER -- OCTAL VOLTAGE--
3141 020220 004737 024626 14$: JSR PC,CONV7 ;DO 7 CONV & AVERAGE THEM
3142 020224 012701 000002 MOV #2,R1
3143 020230 005037 002150 13$: CLR YLOOP ;WAIT
3144 020234 005237 002150 64$: INC YLOOP
3145 020240 023727 002150 177777 64$: CMP YLOOP,#-1
3146 020246 001372 BNE 64$
3147 020250 005301 DEC R1
3148 020252 001366 BNE 13$
3149 020254 032737 000004 003062 BIT #BIT2,AVRSAV
3150 020262 001410 BEQ 40$
3151 020264 005237 002244 INC INTDAT
3152 020270 013746 002244 MOV INTDAT,-(SP) ;;SAVE INTDAT FOR TYPEOUT
3153 020274 104402 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
3154 020276 005337 002244 DEC INTDAT
3155 020302 000403 BR 41$
3156 020304 013746 002244 40$: MOV INTDAT,-(SP) ;;SAVE INTDAT FOR TYPEOUT
3157 020310 104402 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
3158 020312 162737 004000 002244 41$: SUB #4000,INTDAT ;CALCULATE VOLTAGE IN MILLIVOLTS
3159 020320 013737 002244 001140 MOV INTDAT,$GDDAT
3160 020326 063737 001140 001140 ADD $GDDAT,$GDDAT ;MULTIPLY BY 5
3161 020334 063737 001140 001140 ADD $GDDAT,$GDDAT

```



```
3157 020342 063737 002244 001140      ADD      INTDAT,$GDDAT
3158 020350 042737 177770 003062      BIC      #177770,AVRSAV
3159 020356 013704 003062      MOV      AVRSAV,R4
3160 020362 116404 003064      MOV      AVRTBL(R4),R4
3161 020366 060437 001140      ADD      R4,$GDDAT
3162
3163 020372 104401 033361      TYPE     ,M10          ;TAB
3164 020376 104401 033361      TYPE     ,M10
3165 020402 013746 001140      MOV      $GDDAT,-(SP)  ;;SAVE $GDDAT FOR TYPEOUT
                                ;GO TYPE--DECIMAL ASCII WITH SIGN
                                ;CR,LF
3166 020410 104401 035646      TYPDS
3167 020414 004737 025502      TYPE     ,MASSO
3168 020420 000677      JSR      PC,CNTRC      ;CONTROL C?
3169 020422 022626      BR       14$          ;DO IT AGAIN.
20$: 3170 020424 104401 034465      CMP      (SP)+,(SP)+
3171 020430 012637 000004      TYPE     ,M84
3172 020434 152777 000001      MOV      (SP)+,ERRVEC ;SELECTED MUX DID NOT RESPOND
3173 020442 105777 162336      BISB    #RBIT,@CSR    ;POP STACK INTO ERRVEC
3174 020446 000137 017560      TSTB    @LBYTE        ;RESET TIME-OUT
3175      JMP     ADCALB
3176      .SBTTL LOGIC TEST OF A014
```

```
3178
3179 020452
                                ADLOG:
                                :*****
020452 000004
3180 020454 112737 000023 001206 TST23: SCOPE
3181
3182 020462 004737 016652 JSR PC,KLEER ;CLEAR THE IOCM WORLD
3183 020466 012746 000000 MOV #PRO,-(SP) ;SET PSW TO PRIORITY 0
020472 012746 020500 MOV #64$,-(SP)
020476 000002 RTI
020500 000240 64$: NOP
3184 020502 152777 000001 161500 BISB #RBIT , @CSR ;CLEAR RANDOM INTERRUPT
3185 020510 005037 001142 CLR $BDDAT
3186 020514 117737 162262 001142 MOVB @STAT2,$BDDAT ;STORE DATA IN $BDAT
3187 020522 132777 000001 162252 BITB #BIT0 , @STAT2 ;CHECK BUSY BIT CLEAR
3188 020530 001401 BEQ 1$ ;IF NO ERROR CONTINUE
3189 020532 104035 ERROR!35 ;ELSE REPORT ERROR
3190
3191 020534 132777 000002 162240 1$: BITB #BIT1,@STAT2 ;CHECK CONV DONE IS CLEAR
3192 020542 001401 BEQ 2$ ;IF NO ERROR CONTINUE
3193 020544 104035 ERROR!35 ;ELSE REPORT ERROR
3194
3195 020546 132777 000004 162226 2$: BITB #BIT2,@STAT2 ;CHECK ERROR BIT IS CLEAR
3196 020554 001401 BEQ 3$ ;IF NO ERROR CONTINUE
3197 020556 104035 ERROR!35 ;ELSE REPORT ERROR
3198
3199 020560 132777 000010 162214 3$: BITB #BIT3,@STAT2 ;CHECK MUX TIME OUT IS CLEAR
3200 020566 001401 BEQ 4$ ;IF NO ERROR CONTINUE
3201 020570 104035 ERROR!35 ;ELSE TYPE ERROR
3202
3203 020572 132777 000360 162202 4$: BITB #360,@STAT2 ;CHECK GAIN BITS
3204 020600 001401 BEQ BSYCON ;IF NO ERROR CONTINUE
3205 020602 104035 ERROR!35 ;ELSE REPORT ERROR
3206 020604 012777 020750 161402 BSYCON: MOV #10$,@VECTO
3207 020612 012777 000340 161376 MOV #PR7,@VECTOA
3208 020620 152777 000001 161362 BISB #RBIT,@CSR ;CLEAR INTERRUPTS
3209 020626 105777 162146 TSTB @STAT1
3210 020632 012700 177777 MOV #-1 , RO ;INITIALIZE COUNTER
3211 020636 152777 000001 162136 BISB #BIT0 , @STAT2 ;START CONVERSION
3212 020644 005200 1$: INC RO ;SRATR COUNTER
3213 020646 022700 000300 CMP #300 , RO ;CHECK FOR COUNT OF 100
3214 020652 001002 BNE 8$ ;CHECK FOR COUNT OF 100
3215 020654 104040 ERROR!40 ;STILL WAIT FOR CONVERSION
3216 020656 000474 BR 7$ ;CONVERSION NOT COMPLETED IN500MS
3217 020660 132777 000002 162114 8$: BITB #BIT1 , @STAT2 ;CHECK FOR CONVERSION DONE
3218 020666 001766 BEQ 1$ ;IF NOT DONE THEN LOOP
3219 020670 132777 000001 162104 2$: BITB #BIT0 , @STAT2 ;VERIFY BUSY CLEAR
3220 020676 001402 BEQ 3$ ;IF CLEAR CONTINUE
3221 020700 104036 ERROR!36 ;TELL BUSY NOT CLEAR
3222 020702 000462 BR 7$
3223 020704 132777 000004 162070 3$: BITB #BIT2,@STAT2 ;CHECK ERROR CLEAR
3224 020712 001402 BEQ 4$ ;CONTINUE IF CLEAR
3225 020714 104037 ERROR!37 ;ELSE REPORT NOT CLEAR
3226 020716 000454 BR 7$
3227 020720 105777 161264 4$: TSTB @CSR ;TST INTERRUPT FLAG
3228 020724 100402 BMI 5$ ;IT IS SET
3229 020726 104041 ERROR!41
```



```

3230 020730 000447          BR      7$
3231 020732 152777 000100 161250 5$:  BISB  #EBIT,@CSR      :ENEBLE INTERRUPT
3232 020740 000240          NOP
3233 020742 000240          NOP
3234 020744 104041          ERROR!41      :NO INTERRUPT
3235 020746 000440          BR      7$
3236
3237
3238 020750 022626          10$:  CMP   (SP)+,(SP)+      :ADJUST STOCK POINTER
3239 020752 142777 000100 161230  BICB  #EBIT,@CSR
3240 020760 132777 000200 161222 13$:  BITB  #FBIT,@CSR      :INTERRUPT SET?
3241 020766 001427          BEQ   12$           :NO
3242
3243 020770 005037 002242          CLR   TEMP1
3244 020774 117737 161212 002232  MOVB  @IAR,TEMP      :FIND WHICH I/O INTERRUPTED
3245 021002 123737 002232 002172  CMPB  TEMP,TADDR    :IS IT MUT
3246 021010 001417          BEQ   7$           :YES
3247 021012 053737 002224 002232  BIS   BASE,TEMP     :ASSEMBLE ADDRESS OF INTERRUPTING MODULE
3248 021020 152777 000001 161162  BISB  #RBIT,@CSR    :CLEAR INTERRUPT
3249 021026 105777 161200          TSTB  @TEMP
3250 021032 005237 002242          INC   TEMP1
3251 021036 022737 000400 002242  CMP   #400,TEMP1
3252 021044 001345          BNE   13$
3253 021046 104041          ERROR!41      :NO INTERRUPT
3254 021050 152777 000001 161132 7$:  BISB  #RBIT,@CSR    :CLEAR INTERRUPTS
3255 021056 105777 161722          TSTB  @LBYTE
3256 021062 004737 025414          JSR   PC,CLRINT
3257 021066 012746 000000          MOV   #PRO,-(SP)    :SET PSW TO PRIORITY 0
      021072 012746 021100          MOV   #64$,-(SP)
      021076 000002          RTI
      021100 000240          64$:  NOP

3258
3259
3260 021102          CHSEL:
3261 021102 005000          CLR   R0           :SET TO START WITH CHANEL ZERO
3262 021104 005001          CLR   R1           :CLEAR REFERENCE REGISTER
3263 021106 105077 161666          CLRB  @STAT1      :CLEAR CHANEL TO ZERO
3264 021112 105777 161662          TSTB  @STAT1      :VERIFY CHANEL IS ZERO
3265 021116 001406          BEQ   1$           :IF ZERO CONTINUE
3266 021120 005037 001140          CLR   $GDDAT      :CHECK FOR CHANEL 0
3267 021124 117737 161650 001142  MOVB  @STAT1,$BDDAT :GET ACTUAL CHANEL
3268 021132 104042          ERROR!42      :ELSE REPORT NOT CLEAR
3269 021134 005200          1$:  INC   R0           :GET NEXT CHANEL TO CHECK
3270 021136 110077 161636          MOVB  R0,@STAT1   :WRITE CHANEL SELECTED
3271 021142 117701 161632          MOVB  @STAT1,R1   :READ CHANEL SELECTED
3272 021146 020001          CMP   R0,R1       :VERIFY CORRECT CHANEL
3273 021150 001406          BEQ   2$           :CONTINUE IF MATCH
3274 021152 010037 001140          MOV   R0,$GDDAT   :STORE CHANEL WRITTEN
3275 021156 010137 001142          MOV   R1,$BDDAT   :STORE CHANEL READ
3276 021162 104042          ERROR!42      :ELSE REPORT ERROR
3277 021164 000431          BR   CHNSEL
3278 021166 022737 000301 002776 2$:  CMP   #301,GBITE   :CHECK FOR DIFF MODE
3279 021174 001404          BEQ   3$           :IF NOT DIFF MODE
3280 021176 022700 000017          CMP   #17,R0      :CHECK FOR 17 CHANELS
3281 021202 001354          BNE   1$           :GET NEXT CHANEL UNTIL 17 DONE
3282 021204 000403          BR   4$           :EXIT WHEN DONE
3283 021206 022700 000007          3$:  CMP   #7,R0      :ELSE CHECK 7 DIFF CHANELS
    
```

```
3284 021212 001350          BNE      1$          ;GET NEXT CHANEL UNTIL 7 DONE
3285 021214 132777 000004 161560 4$: BITB    #BIT2,@STAT2 ;CHECK IF ERROR BIT CLEAR
3286 021222 001401          BEQ      5$          ;
3287 021224 104055          ERROR!55
3288 021226 105077 161546          5$: CLRB    @STAT1      ;SET CHANELS TO ZERO
3289 021232 001406          BEQ      CHNSEL     ;IF ZERO - DONE
3290 021234 005037 001140          CLR      $GDDAT     ;STORE CHANEL WRITTEN
3291 021240 117737 161534 001142          MOVB    @STAT1,$BDDAT ;STORE CHANEL READ
3292 021246 104042          ERROR!42          ;ELSE REPORT ERROR
3293
3294 021250          CHNSEL:
3295 021250 022737 000301 002776          CMP     #301 ,GBITE ;CHECK IF IN DIFF MODE
3296 021256 001003          BNE     1$          ;IN NOT DO SING END MODE
3297 021260 012700 000010          MOV     #10 , RO    ;START WITH CH10 IN DIFF MODE
3298 021264 000402          BR      2$          ;AND START CHECK
3299 021266 012700 000020          1$: MOV     #20 , RO    ;START WITH CH20 IN SING END MODE
3300 021272 152777 000001 160710 2$: BISB    #RBIT , @CSR ;CLEAR ANY INTERRUPTS
3301 021300 105777 161500          TSTB    @LBYTE     ;CLEAR RIF BIT
3302 021304 110077 161470          4$: MOVB    RO , @STAT1 ;LOAD NON-EXISTANT CHANEL
3303 021310 132777 000004 161464          BITB    #BIT2 , @STAT2 ;CHECK ERROR BIT SET
3304 021316 001002          BNE     5$          ;CONTINUE IF SET
3305 021320 104043          ERROR!43          ;ELSE REPORT NOT SET
3306 021322 000412          BR      ADGAN
3307
3308 021324 132777 000200 160656 5$: BITB    #BIT7,@CSR   ;TEST IF ERROR INTERRUPTED
3309 021332 001002          BNE     6$          ;
3310 021334 104044          ERROR!44          ;NO INTERRUPT ON ERROR
3311 021336 000404          BR      ADGAN
3312 021340 005200          6$: INC     RO        ;UPDATE TO NEXT CHANEL
3313 021342 022700 000040          CMP     #40 , RO    ;CHECK FOR LAST CHANEL
3314 021346 001351          BNE     2$          ;CONTINUE IF NOT DONE
3315
3316 021350 012700 000020          ADGAN: MOV     #20,RO    ;GET FIRST GAIN VALUE
3317 021354 152777 000001 160626 1$: BISB    #RBIT , @CSR ;CLEAR INTERRUPTS
3318 021362 105077 161412          CLRB    @STAT1
3319 021366 105777 161412          TSTB    @LBYTE
3320 021372 110077 161404          MOVB    RO , @STAT2 ;WRITE GAIN
3321 021376 132777 000004 161376          BITB    #BIT2 , @STAT2 ;CHECK FOR ERROR
3322 021404 001002          BNE     2$          ;BRANCH IF ERROR
3323 021406 104045          ERROR!45          ;ELSE REPORT NO ERROR
3324 021410 000405          BR      3$
3325 021412 062700 000020          2$: ADD     #20,RO    ;GET NEXT GAIN
3326 021416 022700 000400          CMP     #400,RO    ;CHECK FOR LAST
3327 021422 001354          BNE     1$          ;IF NOT CONTINUE
3328 021424 152777 000001 160556 3$: BISB    #RBIT , @CSR ;CLEAR INTERRUPTS
3329 021432 105777 161346          TSTB    @LBYTE
3330
3331 021436 005001          ADTBIT: CLR     R1      ;SET DIFF CHANEL NUMBER
3332 021440 005003          CLR     R3        ;SET SEND CHANEL NUMB
3333 021442 005037 002230          CLR     NORAMP    ;CLEAR NO RAMP FLAG
3334 021446 152777 000010 160534          BISB    #TBIT , @CSR ;SET T-BIT
3335 021454 122737 000321 002776 2$: CMPB    #321,GBITE ;CHECK GEN CODE
3336 021462 001403          BEQ     3$
3337 021464 110177 161310          MOVB    R1 , @STAT1 ;WRITE DIFF CHANEL NUMBER
3338 021470 000402          BR      12$
3339 021472 110377 161302          3$: MOVB    R3 , @STAT1 ;SELECT SEND CHANEL NUMBER
3340 021476 004737 024626          12$: JSR     PC , CONV7 ;GO READ A/D DATA
```


3341	021502	005701			TST	R1			
3342	021504	001011			BNE	5\$:IF NOT CHECK NEXT CHANEL
3343	021506	023727	002244	007400	CMP	INTDAT,#7400			:ELSE CHECK +V OUTPUT ALL ONES
3344	021514	002042			BGE	9\$:IF OK CHECK NEXT CHANEL
3345	021516	012737	007400	001140	MOV	#7400,\$GDDAT			:STORE LOWER LIMIT
3346	021524	104046			ERROR!46				:FAILED ALL ONES CHECK CHO
3347	021526	000435			BR	9\$			
3348	021530	022701	000001		4\$: 5\$: CMP	#1 , R1			:CHECK FOR CHANEL #1
3349	021534	001011			BNE	7\$:ELSE GO TO NEXT CHANEL
3350	021536	023727	002244	000400	CMP	INTDAT,#400			:CHECK IF ABS ZERO IS IN LIMIT
3351	021544	003426			BLE	9\$:BRANCH IF OK
3352	021546	012737	000400	001140	MOV	#400,\$GDDAT			:STORE UPPER LIMIT
3353	021554	104046			ERROR!46				:ELSE ERROR
3354	021556	000421			6\$: 7\$: BR	9\$:GET NEXT CHANEL
3355	021560	023727	002244	004005	CMP	INTDAT,#4005			:CHECK HIGH LIMIT OF ZERO VALUE
3356	021566	003404			BLE	8\$:CONTINUE IF OK
3357	021570	012737	004005	001140	MOV	#4005,\$GDDAT			
3358	021576	104046			ERROR!46				
3359	021600	023727	002244	003773	8\$: CMP	INTDAT,#3773			:CHECK LOW LIMIT OF ZERO VALUE
3360	021606	002023			BGE	11\$:CONTINUE IF OK
3361	021610	012737	003773	001140	MOV	#3773,\$GDDAT			
3362	021616	104046			ERROR!46				:ELSE REPORT ERROR
3363	021620	000416			BR	11\$			
3364	021622	005701			9\$: TST	R1			:IS CHANEL #0 SELECTED
3365	021624	001004			BNE	10\$:IF NOT CHECK NEXT CHANEL
3366	021626	005201			INC	R1			:ELSE UPDATE TO NEXT CHANEL
3367	021630	062703	000002		ADD	#2,R3			:UPDATE SEND CHANEL
3368	021634	000707			BR	2\$:AND GO CHECK IT
3369	021636	022701	000001		10\$: CMP	#1 , R1			:IS CHANEL #1 SELECTED
3370	021642	001005			BNE	11\$:IF NOT CONTINUE
3371	021644	062701	000002		ADD	#2 , R1			:ELSE UPDATE TO CHANEL #3
3372	021650	062703	000004		ADD	#4,R3			:UPDATE SEND CHANEL
3373	021654	000677			BR	2\$:AND GO CHECK IT
3374	021656	122737	000321	002776	11\$: CMPB	#321,GBITE			:TEST IF RAMP IS WORKING
3375	021664	001404			BEQ	14\$			
3376	021666	112777	000002	161104	MOVB	#2,@STAT1			:SET RAMP FOR SE
3377	021674	000403			BR	15\$			
3378	021676	112777	000004	161074	14\$: 15\$: MOVB	#4,@STAT1			:SET RAMP FOR DIFF
3379	021704	005000			CLR	R0			
3380	021706	005001			CLR	R1			
3381	021710	004737	024626		17\$: JSR	PC,CONV7			:DO CONVERSIONS
3382	021714	022737	007776	002244	CMP	#7776,INTDAT			:WAIT FOR TOP OF RAMP
3383	021722	003412			BLE	16\$			
3384	021724	105200			INCB	R0			
3385	021726	001370			BNE	17\$			
3386	021730	005201			INC	R1			
3387	021732	022701	000050		CMP	#50,R1			
3388	021736	002364			BGE	17\$			
3389	021740	104056			ERROR!56				:RAMP IS NOT WORKING
3390	021742	005237	002230		INC	NORAMP			
3391	021746	000421			BR	18\$			
3392									
3393	021750	005001			16\$: CLR	R1			
3394	021752	005000			CLR	R0			
3395	021754	004737	024626		20\$: JSR	PC,CONV7			:WAIT FOR BOTTOM OF RUMP
3396	021760	022737	000002	002244	CMP	#2,INTDAT			
3397	021766	002011			BGE	18\$			

```
3398 021770 105200          INCB      R0          ;WAIT UP TO 6SEC
3399 021772 001370          BNE       20$
3400 021774 005201          INC       R1
3401 021776 022701 000050      CMP       #50,R1
3402 022002 002364          BGE       20$
3403 022004 104056          ERROR!56
3404 022006 005237 002230      INC       NORAMP      ;RUMP IS BROKEN
3405 022012 105077 160762      CLRB     @STAT1      ;SELECT CHANNEL 0 ALL ONES
3406 022016 152777 000020 160164 18$:  BISB     #DBIT , @CSR ;SET THE D-BIT
3407 022024 004737 024626          JSR      PC, CONV7   ;GO DO CONVERSION
3408 022030 012737 003774 001140      MOV      #3774,$GDDAT ;GET LOW END TOLER
3409 022036 023737 001140 001142      CMP      $GDDAT,$BDDAT
3410 022044 003401          BLE      43$
3411 022046 104050          ERROR!50
3412 022050 012737 004004 001140 43$:  MOV      #4004,$GDDAT ;GET HIGH END TOLER
3413 022056 023737 001140 001142      CMP      $GDDAT,$BDDAT
3414 022064 002001          BGE      44$
3415 022066 104050          ERROR!50
3416 022070 004737 016652      44$:  JSR      PC,KLEER    ;CLEAR THE WORLD
3417 022074 000004          SCOPE
3418 022076 000207          RTS      PC          ;EXIT
3419
3420          .SBTTL  LINEARITY OF A014 TEST
```


3422	022100	012737	000024	001206	LINEAR:	MOV	#24,\$TESTN	
3423	022106	005037	002234			CLR	RERROR	
3424	022112	005737	002230			TST	NORAMP	
3425	022116	001407				BEQ	40\$	
3426	022120	005737	002240			TST	AFLAG	
3427	022124	001003				BNE	41\$	
3428	022126	104056				ERROR!	56	
3429	022130	104401	035144			TYPE	,MASS51	
3430	022134	000207			41\$:	RTS	PC	
3431	022136	152777	000001	160044	40\$:	BISB	#RBIT,@CSR	;CLEAR IOCM RIF FLAG
3432	022144	105777	160632			TSTB	@STAT2	
3433	022150	005037	002364			CLR	CONV	
3434	022154	005037	002352			CLR	ACOUNT	
3435	022160	005037	002354			CLR	BCOUNT	
3436	022164	152777	000010	160016		BISB	#TBIT,@CSR	;SET T BIT
3437	022172	004737	025414			JSR	PC,CLRINT	
3438	022176	004737	024576			JSR	PC,SETRAM	;SET RAMP
3439	022202	004737	024626		1\$:	JSR	PC,CONV7	
3440	022206	022737	007777	002244		CMP	#7777,INTDAT	;WAIT FOR TOP OF RAMP
3441	022214	001372				BNE	1\$	
3442	022216	005237	002364			INC	CONV	;WAIT FOR 50 CONVERSIONS AT 7777
3443	022222	022737	000050	002364		CMP	#50,CONV	
3444	022230	001364				BNE	1\$	
3445	022232	152777	000001	160542	3\$:	BISB	#BIT0,@STAT2	
3446	022240	132777	000002	160534	16\$:	BITB	#BIT1,@STAT2	
3447	022246	001774				BEQ	16\$	
3448	022250	152777	000001	157732		BISB	#RBIT,@CSR	
3449	022256	117737	160522	002366		MOVB	@LBYTE,DATALO	
3450	022264	117705	160516			MOVB	@HBYTE,R5	
3451	022270	110537	002367			MOVB	R5,DATALO+1	
3452	022274	022737	007776	002366		CMP	#7776,DATALO	;IS RAMP GOING DOWN
3453	022302	003753				BLE	3\$	
3454	022304	013705	002366			MOV	DATALO,R5	
3455	022310	152777	000001	157672	5\$:	BISB	#RBIT,@CSR	;CLEAR DONE FLAG
3456	022316	152777	000001	160456		BISB	#BIT0,@STAT2	;START CONVERSION
3457	022324	117737	160454	002366		MOVB	@LBYTE,DATALO	;READ LOW BYTE
3458	022332	117705	160450			MOVB	@HBYTE,R5	;READ HIGH BYTE
3459	022336	110537	002367			MOVB	R5,DATALO+1	
3460	022342	013705	002366			MOV	DATALO,R5	;STORE IN TFXR LOC
3461	022346	010500				MOV	R5,R0	
3462	022350	005237	002352			INC	ACOUNT	;STORE # OF CONVERSIONS THRU RAMP
3463	022354	001401				BEQ	7\$	
3464	022356	000402				BR	8\$	
3465	022360	005237	002354		7\$:	INC	BCOUNT	;INCREAM IF ACOUNT OVERFLOW >32564
3466	022364	005700			8\$:	TST	R0	;IS IT END OF RAMP ?
3467	022366	001405				BEQ	10\$	
3468	022370	132777	000002	160404	2\$:	BITB	#BIT1,@STAT2	;CHECK CONVER DONE
3469	022376	001774				BEQ	2\$;LOOP IF NOT DONE
3470	022400	000743				BR	5\$;ENABLE INTERR,START CONV
3471	022402	005001			10\$:	CLR	R1	
3472	022404	006237	002354		11\$:	ASR	BCOUNT	;FIND AVERAGE OF CONVERSIONS PER POINT
3473	022410	006037	002352			ROR	ACOUNT	;BY DIVIDING # OF CONVERSIONS BY 4096
3474	022414	005201				INC	R1	;POINTS
3475	022416	022701	000014			CMP	#12.,R1	
3476	022422	001370				BNE	11\$	
3477	022424	005337	002352			DEC	ACOUNT	
3478	022430	013703	002352			MOV	ACOUNT,R3	

3479	022434	013702	002352			MOV	ACOUNT,R2		
3480	022440	006202				ASR	R2		;LOW LIMIT AVERAGE OF CONVERSIONS/POINT
3481	022442	005503				ADC	R3		
3482	022444	060203				ADD	R2,R3		;HIGH LIMIT AVERAGE " "
3483	022446	005203				INC	R3		
3484	022450	005302				DEC	R2		
3485	022452	005037	002370		49\$:	CLR	CONVCT		
3486	022456	005037	002364			CLR	CONV		
3487	022462	152777	000001	157520		BISB	#RBIT,@CSR		;CLEAR INTERRUPTS
3488	022470	105777	160304			TSTB	@STAT1		
3489	022474	004737	024626		13\$:	JSR	PC,CONV7		
3490	022500	022737	007777	002244		CMP	#7777,INTDAT		;WAIT FOR TOP OF RAMP
3491	022506	001372				BNE	13\$		
3492	022510	005237	002364			INC	CONV		;WAIT 50 CONVERSIONS AT TOP OF RAMP
3493	022514	022737	000050	002364		CMP	#50,CONV		
3494	022522	001364				BNE	13\$		
3495	022524	152777	000001	160250	14\$:	BISB	#BIT0,@STAT2		
3496	022532	132777	000002	160242	17\$:	BITB	#BIT1,@STAT2		
3497	022540	001774				BEQ	17\$		
3498	022542	152777	000001	157440		BISB	#RBIT,@CSR		
3499	022550	117737	160230	002366		MOVB	@LBYTE,DATALO		
3500	022556	117705	160224			MOVB	@HBYTE,R5		
3501	022562	110537	002367			MOVB	R5,DATALO+1		
3502	022566	022737	007776	002366		CMP	#7776,DATALO		;IS RAMP GOING DOWN ?
3503	022574	001353				BNE	14\$		
3504	022576	013705	002366			MOV	DATALO,R5		
3505	022602	010500				MOV	R5,R0		;INITIALIZE LOCATIONS
3506	022604	010237	002352			MOV	R2,ACOUNT		;SET THIS TWO COUNTERS INITIALLY TO MIN COUNT
3507	022610	010237	002354			MOV	R2,BCOUNT		
3508	022614	005037	002356			CLR	CCOUNT		
3509	022620	005037	002362			CLR	DCOUNT		
3510	022624	005037	002372			CLR	ECOUNT		
3511	022630	005004				CLR	R4		
3512	022632	152777	000001	157350	33\$:	BISB	#RBIT,@CSR		;CLEAR INTERRUPTS
3513	022640	152777	000001	160134		BISB	#BIT0,@STAT2		;START CONVER
3514	022646	117737	160132	002366		MOVB	@LBYTE,DATALO		;READ LOW BYTE
3515	022654	117705	160126			MOVB	@HBYTE,R5		;READ HIGH BYTE
3516	022660	110537	002367			MOVB	R5,DATALO+1		
3517	022664	013705	002366			MOV	DATALO,R5		;STORE IN TFXR LOC
3518	022670	010501				MOV	R5,R1		;STORE LAST CONV
3519	022672	160001				SUB	R0,R1		;GET DIFFERENCE BETWEEN LAST AND PREVIOUS ONE
3520	022674	005401				NEG	R1		
3521	022676	006301				ASL	R1		
3522	022700	005261	002356			INC	CCOUNT(R1)		;INC APPROPRIATE COUNTER
3523	022704	005701				TST	R1		;CHECK IF LAST READING WAS LOWER THEN ONE BEFORE
3524	022706	003005				BGT	31\$		
3525									
3526	022710	132777	000002	160064	32\$:	BITB	#BIT1,@STAT2		;WAIT FOR END OF CONV
3527	022716	001774				BEQ	32\$		
3528	022720	000744				BR	33\$;START NEXT CONV
3529									
3530	022722	023702	002352		31\$:	CMP	ACOUNT,R2		;CHECK FOR MIN # OF CONV
3531	022726	103430				BLO	34\$;ERROR
3532	022730	023703	002352			CMP	ACOUNT,R3		;CHECK FOR MAX
3533	022734	101025				BHI	34\$;ERROR
3534	022736	013737	002354	002352	35\$:	MOV	BCOUNT,ACOUNT		
3535	022744	013737	002356	002354		MOV	CCOUNT,BCOUNT		


```
3536 022752 013737 002362 002356      MOV      DCOUNT,CCOUNT
3537 022760 013737 002372 002362      MOV      ECOUNT,DCOUNT
3538 022766 005037 002372      CLR      ECOUNT
3539 022772 005300      DEC      R0                      ;SET NEXT LOWER CONV FOR CCOUNT
3540 022774 003345      BGT     32$                      ;CHECK FOR END OF RAMP
3541 022776 005300      DEC      R0                      ;CHECK LAST TWO CONV
3542 023000 022700 177775      CMP     #-3,R0
3543 023004 001346      BNE     31$
3544 023006 000414      BR      22$
3545 023010 012764 000002 002374 34$:  MOV     #2,RAMP1(R4)             ;STORE ERROR
3546 023016 060064 002374      ADD     R0,RAMP1(R4)
3547 023022 013764 002352 002514      MOV     ACOUNT,RAMP2(R4)
3548 023030 005724      TST     (R4)+                   ;INC ERROR POINTER
3549 023032 022704 000050      CMP     #40.,R4
3550 023036 001337      BNE     35$
3551
3552 023040 005704      22$:  TST     R4                      ;ANY ERRORS ?
3553 023042 001436      BEQ     24$                      ;NO,EXIT
3554 023044 005237 002234      INC     RERROR
3555 023050 022737 000003 002234      CMP     #3,RERROR              ;DO IT 3 TIMES UNTILL NO ERRORS
3556 023056 001402      BEQ     25$
3557 023060 000137 022452      JMP     49$
3558 023064 104401 034565      25$:  TYPE     ,FM53
3559 023070 005005      CLR     R5                      ;LINEARITY ERROR
3560 023072 016537 002374 002754 23$:  MOV     RAMP1(R5),BLAST
3561 023100 016537 002514 001142      MOV     RAMP2(R5),$BDDAT
3562 023106 010237 002354      MOV     R2,BCOUNT
3563 023112 010337 002356      MOV     R3,CCOUNT
3564 023116 104052      ERROR!52
3565 023120 062705 000002      ADD     #2,R5                   ;INC ERROR TABLE POINTER
3566 023124 020405      CMP     R4,R5                   ;LAST ERROR ?
3567 023126 001361      BNE     23$
3568 023130 022704 000050      CMP     #40.,R4
3569 023134 001001      BNE     24$
3570 023136 104054      ERROR!54
3571 023140 004737 025414 24$:  JSR     PC,CLRINT
3572 023144 004737 016652      JSR     PC,KLEER
3573 023150 000207      RTS     PC
3574
3575
```

```
3578 .SBTTL A/D MONOTONICITY TEST
3579
3580 :*****
3581 : MONTONICITY TEST
3582 :*****
3583 023152 012737 000025 001206 RUMP: MOV #25,$TESTN
3584 023160 012746 000000 MOV #PRO,-(SP) ;SET PSW TO PRIORITY 0
      023164 012746 023172 MOV #64$,-(SP)
      023170 000002 RTI
      023172 000240 64$: NOP
3585 023174 005737 002230 TST NORAMP
3586 023200 001407 BEQ 1$
3587 023202 005737 002240 TST AFLAG
3588 023206 001003 BNE 2$
3589 023210 104056 ERROR!56 ;NO RAMP
3590 023212 104401 035144 TYPE ,MASS51
3591 023216 000207 2$: RTS PC
3592 023220 1$:
3593 023220 152777 000001 156762 BISB #RBIT,@CSR ;CLEAR DONE FLAG
3594 023226 105777 157550 TSTB @STAT2
3595 023232 152777 000010 156750 BISB #TBIT,@CSR ;SET T BIT
3596 023240 004737 025414 JSR PC,CLRINT
3597 023244 005004 RAMPST: CLR R4 ;CLEAR ERROR POINTER
3598 023246 004737 024576 JSR PC,SETRAM ;SET RAMP
3599 023252 152777 000001 157522 1$: BISB #BIT0,@STAT2
3600 023260 132777 000002 157514 2$: BITB #BIT1,@STAT2
3601 023266 001774 BEQ 2$
3602 023270 152777 000001 156712 BISB #RBIT,@CSR
3603 023276 117737 157502 002366 MOVB @LBYTE,DATALO
3604 023304 117737 157476 002367 MOVB @HBYTE,DATALO+1
3605 023312 005737 002366 TST DATALO ;WAIT FOR -10.240 VOLTS
3606 023316 001355 BNE 1$
3607 023320 005037 002754 BEG: CLR BLAST ;STORE FIRST CONVERSION
3608 023324 005037 002756 CLR LAST
3609 023330 005037 002760 CLR LASTCN
3610 023334 152777 000001 157440 STCO: BISB #BIT0,@STAT2 ;START CONVERSION
3611 023342 132777 000002 157432 13$: BITB #BIT1,@STAT2
3612 023350 001774 BEQ 13$
3613 023352 152777 000001 156630 BISB #RBIT,@CSR
3614 023360 117737 157420 002366 MOVB @LBYTE,DATALO
3615 023366 117737 157414 002367 MOVB @HBYTE,DATALO+1
3616 023374 013737 002366 002756 MOV DATALO, LAST
3617 023402 013737 002366 001142 MOV DATALO,$BDDAT
3618 023410 023737 002754 002756 CMP BLAST, LAST
3619 023416 001431 BEQ 3$
3620 023420 005337 002756 DEC LAST
3621 023424 023737 002754 002756 CMP BLAST, LAST ;COMPARE BLAST+1WITH LAST
3622 023432 001416 BEQ 2$
3623 023434 062737 000002 002756 ADD #2, LAST
3624 023442 023737 002754 002756 CMP BLAST, LAST
3625 023450 001401 BEQ 1$
3626 023452 000431 BR 4$ ;ERROR MESSAGE
3627 023454 005737 002760 1$: TST LASTCN ;IS LASTCN=0 ?
3628 023460 001410 BEQ 3$ ;YES
3629 023462 005337 002754 DEC BLAST ;MOVE LAST CONVERSION BEFORE LAST LOCATION
3630 023466 000405 BR 3$
3631 023470 005737 002760 2$: TST LASTCN ;IS LASTCN=0 ?
```



```
3632 023474 001002      BNE      3$
3633 023476 005237 002754      INC      BLAST
3634 023502 023727 002756 007777 3$:      CMP      LAST, #7777      ; IS LAST=10.240 VOLTS ?
3635 023510 001003      BNE      5$
3636 023512 012737 000001 002760      MOV      #1, LASTCN      ; SET FLAG AT TOP OF RAMP
3637 023520 005737 002756      TST      LAST      ; ARE WE STILL AT MINUS 10 VOLTS ?
3638 023524 001303      BNE      STCO
3639 023526 005737 002760      TST      LASTCN      ; IS RAMP GOING UP ?
3640 023532 001700      BEQ      STCO
3641 023534 000431      BR       27$
3642 023536 013737 002754 001140 4$:      MOV      BLAST, $GDDAT      ; CHECK FOR ERRORS
3643 023544 005737 002760      TST      LASTCN      ; GOOD DATA FOR ERROR MESSAGE
3644 023550 001003      BNE      7$      ; IS RAMP GOING UP ?
3645 023552 062737 000002 002754      ADD      #2, BLAST      ; YES, DECREMENT LAST CONVERSION
3646 023560 005337 002754      DEC      BLAST      ; RAMP GOING DOWN
3647 023564 013764 001140 002374      MOV      $GDDAT,RAMP1(R4)      ; ADJUST BLAST AFTER ERROR
3648 023572 013764 001142 002514      MOV      $BDDAT,RAMP2(R4)
3649 023600 013764 002760 002634      MOV      LASTCN,RAMP3(R4)
3650 023606 062704 000002      ADD      #2,R4      ; STORE RAMP SLOPE IN TABLE
3651 023612 022704 000050      CMP      #40.,R4      ; INCREMENT TABLE POINTER
3652 023616 001331      BNE      3$      ; IS TABLE FULL
3653 023620 005704      TST      R4      ; NO
3654 023622 001425      BEQ      OUT      ; ANY ERRORS ?
3655 023624 104401 034677      TYPE      ,MASS55      ; NO, EXIT
3656 023630 005003      CLR      R3      ; OUT OF RANGE BY TWO BITS
3657 023632 016337 002374 001140 28$:      MOV      RAMP1(R3), $GDDAT      ; POINTER FOR ERROR TABLE
3658 023640 016337 002514 001142      MOV      RAMP2(R3), $BDDAT
3659 023646 016337 002634 002760      MOV      RAMP3(R3), LASTCN
3660 023654 104051      ERROR!51
3661 023656 062703 000002      ADD      #2,R3      ; INCREMENT DATA POINTER
3662 023662 020403      CMP      R4,R3      ; LAST ERROR ?
3663 023664 001362      BNE      28$      ; NO
3664 023666 020427 000050      CMP      R4,#40.      ; TOO MANY ERRORS ?
3665 023672 001001      BNE      OUT
3666 023674 104054      ERROR!54
3667
3668 023676 004737 025414      OUT:     JSR      PC,CLRINT
3669 023702 004737 016652      JSR      PC,KLEER
3670 023706 000207      RTS      PC
3671
```

```

3673
3674          .SBTTL  LOGIC TEST OF A156 AND A157
3675
3676 023710 012737 000026 001206 MUX1:  MOV    #26,$TESTN
3677 023716 012737 000156 002166      MOV    #156,$MUT
3678 023724 013737 003000 002176      MOV    STAT1,TBADDR
3679 023732 105077 157044          CLR   @STAT2
3680 023736 005037 001142          CLR   $BDDAT
3681 023742 013746 000004          MOV    ERRVEC,-(SP)      ;;PUSH ERRVEC ON STACK
3682 023746 012737 024570 000004      MOV    #30$,ERRVEC
3683 023754 113777 002152 157016      MOV   MXNUM,@STAT1      ;SET MUX #
3684 023762 013737 002152 001140      MOV   MXNUM,$GDDAT
3685 023770 117737 157004 001142      MOV   @STAT1,$BDDAT      ;CHECK IF MUX RESPONDS
3686 023776 123737 001140 001142      CMPB  $GDDAT,$BDDAT
3687 024004 001403          BEQ   1$
3688 024006 104060          ERROR!60                ;ERROR IN MUX REGISTER
3689 024010 000137 024554          JMP   13$
3690 024014 113777 002152 156756 1$:  MOV   MXNUM,@STAT1
3691 024022 004737 027026          JSR   PC,GCODE
3692 024026 122737 000342 002776      CMPB  #342,GBITE        ;SE GENERIC CODE?
3693 024034 001455          BEQ   2$
3694 024036 122737 000322 002776      CMPB  #322,GBITE        ;DIF MODE A156
3695 024044 001412          BEQ   20$
3696 024046 122737 000323 002776      CMPB  #323,GBITE        ;A157?
3697 024054 001406          BEQ   20$                ;YES, DIFFERENTIAL MODE
3698 024056 013737 002776 002770      MOV   GBITE,TEMP3
3699 024064 104061          ERROR!61                ;WRONG GENERIC CODE
3700 024066 000137 024554          JMP   13$
3701 024072 012705 000017          20$: MOV   #17,R5
3702 024076 005004          CLR   R4
3703 024100 012737 000040 001140      MOV   #40,$GDDAT
3704 024106 113777 002152 156664      MOV   MXNUM,@STAT1
3705 024114 113777 001140 156660 43$:  MOV   $GDDAT,@STAT2      ;SET TOO HIGH CHANNEL
3706 024122 132777 000004 156652      BITB  #BIT2,@STAT2      ;CHECK IF ERROR SET
3707 024130 001001          BNE   42$
3708 024132 104062          ERROR!62                ;ERROR BIT NOT SET
3709 024134 004737 025502          42$: JSR   PC,CNTRC
3710 024140 152777 000001 156042      BISB  #RBIT,@CSR        ;CLEAR ERROR BIT
3711 024146 105777 156626          TSTB  @STAT1
3712 024152 005237 001140          INC   $GDDAT
3713 024156 005204          INC   R4
3714 024160 022704 000020          CMP   #20,R4
3715 024164 001353          BNE   43$
3716 024166 000402          BR    21$
3717 024170 012705 000037          2$:  MOV   #37,R5
3718 024174 010537 001140          21$: MOV   R5,$GDDAT        ;CHECK ALL CHANNELS
3719 024200 053737 002152 001140      BIS   MXNUM,$GDDAT
3720 024206 113777 001140 156564 4$:  MOV   $GDDAT,@STAT1
3721 024214 117737 156560 001142      MOV   @STAT1,$BDDAT
3722 024222 123737 001140 001142      CMPB  $GDDAT,$BDDAT
3723 024230 001403          BEQ   3$
3724 024232 104063          ERROR!63                ;CHANN SELECT ERROR
3725 024234 004737 025502          JSR   PC,CNTRC
3726 024240 005337 001140          3$:  DEC   $GDDAT
3727 024244 023737 002152 001140      CMP   MXNUM,$GDDAT
3728 024252 001355          BNE   4$
3729

```


3730	024254	005000			CLR	R0		
3731	024256	152777	000001	155724	BISB	#RBIT,@CSR		
3732	024264	005037	001140		CLR	\$GDDAT		
3733	024270	117737	156506	001142	MOVB	@STAT2,\$BDDAT		;CHECK IF ERROR,GAIN & DONE CLEAR
3734	024276	105737	001142		TSTB	\$BDDAT		
3735	024302	001001			BNE	5\$		
3736	024304	104064			ERROR!64			;STATUS REG ERROR
3737	024306	152777	000360	156466	5\$:	BISB	#360,@STAT2	;SET WRONG GAIN
3738	024314	132777	000004	156460		BITB	#BIT2,@STAT2	
3739	024322	001001			BNE	6\$		
3740	024324	104066			ERROR!66			;ERROR BIT NOT SET
3741	024326	152777	000001	155654	6\$:	BISB	#RBIT,@CSR	
3742	024334	105777	156442		TSTB	@STAT2		;CLEAR ERROR
3743	024340	001001			BNE	7\$		
3744	024342	104067			ERROR!67			;ERROR BIT NOT CLEAR
3745	024344	005001			7\$:	CLR	R1	
3746	024346	156177	003032	156426	22\$:	BISB	GAIN(R1),@STAT2	
3747	024354	152777	000010	155626		BISB	#TBIT,@CSR	
3748	024362	113777	002152	156410		MOVB	MXNUM,@STAT1	
3749	024370	152777	000001	156404		BISB	#BIT0,@STAT2	;START CONVERSION
3750	024376	005200			9\$:	INC	R0	
3751	024400	001002			BNE	8\$		
3752	024402	104070			ERROR!70			;CONVERSION NEVER DONE
3753	024404	000463			BR	13\$		
3754	024406	132777	000002	156366	8\$:	BITB	#BIT1,@STAT2	;WAIT FOR CONV DONE
3755	024414	001770			BEQ	9\$		
3756	024416	132777	000004	156356		BITB	#BIT2,@STAT2	
3757	024424	001002			BNE	10\$		
3758	024426	000452			BR	13\$		
3759	024430	104071			ERROR!71			;ERROR BIT SET AFTER CONV.
3760	024432	012737	003774	001140	10\$:	MOV	#3774,\$GDDAT	
3761	024440	117737	156340	001142		MOVB	@LBYTE,\$BDDAT	
3762	024446	117737	156334	001143		MOVB	@HBYTE,\$BDDAT+1	
3763	024454	042737	170000	001142		BIC	#170000,\$BDDAT	;GET CONVERSION RESULTS
3764	024462	023737	001140	001142		CMP	\$GDDAT,\$BDDAT	
3765	024470	003401			BLE	11\$		
3766	024472	104072			ERROR!72			;CHECK LOW LIMIT
3767	024474	012737	004004	001140	11\$:	MOV	#4004,\$GDDAT	
3768	024502	023737	001140	001142		CMP	\$GDDAT,\$BDDAT	
3769	024510	002001			BGE	12\$		
3770	024512	104072			ERROR!72			;CHECK HIGH LIMIT
3771	024514	117737	156260	001142	12\$:	MOVB	@STAT1,\$BDDAT	
3772	024522	001403			BEQ	23\$		
3773	024524	005037	001140		CLR	\$GDDAT		
3774	024530	104073			ERROR!73			;MUX # NOT CLEAR AFTER CONVERSION
3775	024532	122737	000323	002776	23\$:	CMPB	#323,GBITE	
3776	024540	001005			BNE	13\$		
3777	024542	062701	000002		ADD	#2,R1		
3778	024546	005761	003032		TST	GAIN(R1)		
3779	024552	001275			BNE	22\$		
3780	024554	004737	016652		13\$:	JSR	PC,KLEER	
3781	024560	012637	000004		MOV	(SP)+,ERRVEC		::POP STACK INTO ERRVEC
3782	024564	000004			SCOPE			
3783	024566	000207			RTS	PC		
3784	024570	022626			30\$:	CMP	(C0)+,(SP)+	
3785	024572	104065			ERROR!65			
3786	024574	000767			BR	13\$		

3787


```
3790          .SBTTL  SUBROUTINES
3791
3792
3793
3794
3795 024576 022737 000321 002776 SETRAM: CMP      #321,GBITE
3796 024604 001004          BNE      1$
3797 024606 112777 000004 156164          MOVB   #4,@STAT1
3798 024614 000403          BR      2$
3799 024616 112777 000002 156154 1$:      MOVB   #2,@STAT1
3800 024624 00C207          2$:      RTS      PC
3801
3802          ; THIS SUBROUTINE DOES 8 CONVERSIONS AND STORES AVERAGE IN INTDAT
3803 024626 117737 156146 003054 ;CONV7: MOVB   @STAT1,ST1SAV          ;SAVE STAT1
3804 024634 117737 156142 003056          MOVB   @STAT2,ST2SAV          ;SAVE STAT2
3805 024642 005037 002244          CLR      INTDAT
3806 024646 005037 002774          CLR      TEMP5
3807 024652 152777 000001 156122 2$:      BISB   #BIT0,@STAT2          ;START CONVERSION
3808 024660 132777 000002 156114 1$:      BITB   #BIT1,@STAT2          ;WAIT FOR CONVERSION DONE
3809 024666 001774          BEQ      1$
3810 024670 152777 000001 155312          BISB   #RBIT,@CSR
3811 024676 117737 156102 002232          MOVB   @LBYTE,TEMP          ;ASSEMBLE DATA
3812 024704 117737 156076 002233          MOVB   @HBYTE,TEMP+1
3813 024712 042737 170000 002232          BIC    #170000,TEMP
3814 024720 063737 002232 002244          ADD    TEMP,INTDAT          ;ADD TO PREVIOUS DATA
3815 024726 005237 002774          INC    TEMP5
3816 024732 113777 003054 156040          MOVB   ST1SAV,@STAT1
3817 024740 113777 003056 156034          MOVB   ST2SAV,@STAT2
3818 024746 022737 000010 002774          CMP    #10,TEMP5          ;IS IT LAST ONE
3819 024754 001336          BNE     2$                  ;NO
3820 024756 013737 002244 003062          MOV    INTDAT,AVRSAV          ;SAVE TOTAL
3821 024764 006237 002244          ASR    INTDAT          ;FIND AVERAGE
3822 024770 006237 002244          ASR    INTDAT
3823 024774 006237 002244          ASR    INTDAT
3824 025000 042737 170000 002244          BIC    #170000,INTDAT
3825 025006 013737 002244 001142          MOV    INTDAT,$BDDAT
3826 025014 000207          RTS      PC
3827
3828          ; THIS SUBROUTINE IS USED TO CHECK IF
3829          ; A BITE OF DATA IN $GDDAT=$BDDAT.
3830          ; IT CHECKS AS MANY BYTES AS SPECIFIED
3831          ; BY      BYTNUM
3832
3833 025016          TSTBYT:
3834 025016 013746 002170          MOV    BYTNUM,-(SP)          ;;PUSH BYTNUM ON STACK
3835 025022 013746 002172          MOV    TADDR,-(SP)          ;;FUSH TADDR ON STACK
3836 025026 117737 155140 001142 2$:      MOVB   @TADDR,$BDDAT          ;MOV DATA TO BDDAT
3837 025034 123737 001140 001142          CMPB   $GDDAT,$BDDAT          ;IS DATA OK
3838 025042 001404          BEQ    1$
3839 025044 013737 002172 002176          MOV    TADDR,TBADDR
3840 025052 104007          ERROR!7
3841 025054 005237 002172          1$:      INC    TADDR          ;NEXT BYTE
3842 025060 005337 002170          DEC    BYTNUM
3843 025064 001360          BNE     2$
3844 025066 012637 002172          MOV    (SP)+,TADDR          ;;POP STACK INTO TADDR
3845 025072 012637 002170          MOV    (SP)+,BYTNUM          ;;POP STACK INTO BYTNUM
3846 025076 000207          RTS      PC
```

```
3846  
3847 025100          TSTONE:  
      025100 013746 002170      MOV    BYTNUM,-(SP)      ;;PUSH BYTNUM ON STACK  
3848 025104 013746 002172      MOV    TADDR,-(SP)      ;;PUSH TADDR ON STACK  
3849 025110 117737 155056 001142 2$:  MOVB  @TADDR,$BDDAT      ;MOV DATA TO BDDAT  
3850 025116 123737 001140 001142      CMPB  $GDDAT,$BDDAT      ;IS DATA OK  
3851 025124 001404          BEQ    1$  
3852 025126 013737 002172 002176      MOV    TADDR,TBADDR  
3853 025134 104011          ERROR!11  
3854 025136 005237 002172      1$:  INC    TADDR              ;NEXT BYTE  
3855 025142 005337 002170      DEC    BYTNUM  
3856 025146 001360          BNE    2$  
3857 025150 012637 002172      MOV    (SP)+,TADDR      ;;POP STACK INTO TADDR  
3858 025154 012637 002170      MOV    (SP)+,BYTNUM     ;;POP STACK INTO BYTNUM  
3859 025160 000207          RTS    PC
```



```
3861                                     ;THIS SUBROUTINE IS USED TO SET AS MANY
3862                                     ;BYTES AS SPECIFIED BY BYTNUM IN MUT
3863
3864 025162                               SETBYT:
3865 025162 013746 002170                 MOV     BYTNUM,-(SP)      ;;PUSH BYTNUM ON STACK
3866 025166 013746 002172                 MOV     TADDR,-(SP)     ;;PUSH TADDR ON STACK
3867 025172 113777 001140 154772 1$:     MOV     $GDDAT,@TADDR   ;SET DATA IN MUT
3868 025200 005237 002172                 INC     TADDR
3869 025204 005337 002170                 DEC     BYTNUM
3870 025210 001370                         BNE     1$
3871 025212 012637 002172                 MOV     (SP)+,TADDR    ;;POP STACK INTO TADDR
3872 025216 012637 002170                 MOV     (SP)+,BYTNUM  ;;POP STACK INTO BYTNUM
                                     RTS     PC
```

```
3874  
3875  
3876 ;THESE TWO SUBROUTINES ARE USED IN MAPPING THE SYSTEM  
3877 025224 005004 GENCOD: CLR R4  
3878 025226 012703 002026 MOV #GENER,R3  
3879 025232 022301 3$: CMP (R3)+,R1  
3880 025234 001404 BEQ 1$  
3881 025236 062704 000006 ADD #6,R4  
3882 025242 005713 TST (R3)  
3883 025244 001372 BNE 3$  
3884 025246 000207 1$: RTS PC  
3885  
3886  
3887  
3888  
3889 025250 005004 TABLE: CLR R4  
3890 025252 012703 002026 MOV #GENER,R3  
3891 025256 022301 3$: CMP (R3)+,R1  
3892 025260 001404 BEQ 1$  
3893 025262 062704 000002 ADD #2,R4  
3894 025266 005713 TST (R3)  
3895 025270 001372 BNE 3$  
3896 025272 010022 1$: MOV R0,(R2)+  
3897 025274 010125 MOV R1,(R5)+  
3898 025276 000207 RTS PC
```



```
3900 ;THIS SUBROUTINE CHECKS IF A BIT GET SET AND CLEAR IN CSR
3901
3902 025300 113777 001140 154702 BITSET: MOVB $GDDAT,@CSR ;LOAD TESTED BIT
3903 025306 005037 001142 CLR $BDDAT
3904 025312 005037 002150 CLR YLOOP ;WAIT
      025316 005237 002150 64$: INC YLOOP
      025322 023727 002150 000007 CMP YLOOP,#7
      025330 001372 BNE 64$
3905 025332 117737 154652 001142 MOVB @CSR,$BDDAT ;READ CSR
3906 025340 023727 001206 000005 CMP $TESTN,#5
3907 025346 001403 BEQ 3$
3908 025350 023727 001206 000004 CMP $TESTN,#4
3909 025356 001003 3$: BNE 2$
3910 025360 142737 000200 001142 BICB #FBIT,$BDDAT
3911 025366 123737 001140 001142 2$: CMPB $GDDAT,$BDDAT ;COMPARE THEM
3912 025374 001002 BNE 1$
3913 025376 062716 000002 ADD #2,(SP) ;NO ERROR
3914 025402 000207 1$: RTS PC
3915
3916 ;THIS IS INTERRUPT ROUTINE FOR LINE CLOCK
3917 ;WHEN DIAGNOSTIC DOES NOT USE IT
3918
3919 025404 NOCLK: RTI
3920 025404 000002
3921
3922 ;THIS IS INTERRUPT ROUTINE FOR CLOCK TO COUNT TICKS
3923
3924 025406 005237 002226 COUNT: INC CLK
3925 025412 000002 RTI
3926
```

```
3928                                     ;THIS SUBROUTINE CLEARS UNEXPECTED INTERRUPTS FROM DBUS
3929 025414 005037 002770          CLRINT: CLR      TEMP3
3930 025420 005037 002772          CLR      TEMP4
3931 025424 132777 000200 154556 2$: BITB    #FBIT,@CSR      ;IS INTERRUPT PENDING?
3932 025432 001422                BEQ      1$                ;NO
3933 025434 117737 154552 002770    MOVB    @IAR,TEMP3
3934 025442 053737 002224 002770    BIS     BASE,TEMP3
3935 025450 152777 000001 154532    BISB   #RBIT,@CSR
3936 025456 105777 155306          TSTB   @TEMP3
3937 025462 005237 002772          INC     TEMP4
3938 025466 032737 000400 002772    BIT     #BIT8,TEMP4
3939 025474 001753                BEQ     2$
3940 025476 104023                ERROR!23
3941 025500 000207                1$:   RTS      PC          ;UNABLE TO CLEAR INTERRUPT
3942
3943
3944                                     ;THIS SUBROUTINE CHECKS FOR CONTROL C
3945
3946
3947 025502 117746 153454          CNTRC: MOVB    @$TKB,-(SP)
3948 025506 042716 000200          BIC     #BIT7,(SP)      ;CLEAR PARITY BIT
3949 025512 122716 000003          CMPB   #3,(SP)        ;CONTROL C?
3950 025516 001007                BNE     1$                ;NO
3951 025520 004737 016652          JSR    PC,KLEER
3952 025524 012737 005044 000004    MOV    #TMOVEC,ERRVEC
3953 025532 000137 004546          JMP    MONIT
3954 025536 122716 000001          1$:   CMPB   #1,(SP)
3955 025542 001003                BNE     2$
3956 025544 022626                CMP     (SP)+,(SP)+
3957 025546 000177 155306          JMP    @RETURN
3958 025552 005726                2$:   TST   (SP)+
3959 025554 000207                RTS     PC
3960
3961
3962 025556 004737 025502          KBINT: JSR    PC,CNTRC
3963 025562 000002                RTI
```



```
3965                                     ;THIS SUBROUTINE IS USED IN THE LOOP TEST
3966                                     ;AND IT IS SIMILAR TO THE TSTBYT ROUTINE
3967
3968 025564                                CHKBYT:
025564 013746 002242                       MOV     TEMP1,-(SP)      ;;PUSH TEMP1 ON STACK
3969 025570 013746 002170                   MOV     BYTNUM,-(SP)    ;;PUSH BYTNUM ON STACK
3970 025574 117737 154442 001142 2$:      MOVB   @TEMP1,$BDDAT    ;;READ DATA FROM INPUT MODULE
3971 025602 123737 001140 001142          CMPB   $GDDAT,$BDDAT
3972 025610 001410                          BEQ     1$
3973 025612 013737 002172 002176          MOV     TADDR,TBADDR
3974 025620 012637 002170                   MOV     (SP)+,BYTNUM   ;;POP STACK INTO BYTNUM
3975 025624 012637 002242                   MOV     (SP)+,TEMP1    ;;POP STACK INTO TEMP1
3976 025630 000207                          RTS     PC              ;ERROR
3977 025632 005237 002242 1$:            INC     TEMP1          ;GO TO NEXT BYTE
3978 025636 005337 002170                   DEC     BYTNUM
3979 025642 001354                          BNE     2$
3980 025644 012637 002170                   MOV     (SP)+,BYTNUM   ;;POP STACK INTO BYTNUM
3981 025650 012637 002242                   MOV     (SP)+,TEMP1    ;;POP STACK INTO TEMP1
3982 025654 062716 000002                   ADD     #2,(R6)        ;BYPASS ERROR IN MAINLINE CODE
3983 025660 000207                          RTS     PC
3984
3985
3986
3987
3988                                     ;THIS SUBROUTINE STARTS CONVERSION
3989                                     ;THIS SUBROUTINE WAITS FOR A <CR> AND
3990                                     ;MONITORS CNTRLC
025662 104406                                CRTST: RDCHR
3991 025664 012637 002360                   MOV     (SP)+,ANSW     ;SAVE ANSWER
3992 025670 004737 025502                   JSR     PC,CNTRC      ;CONTROL C ?
3993 025674 122737 000015 002360          CMPB   #15,ANSW       ;A <CR> ?
3994 025702 001367                          BNE     CRTST         ;WRONG CHARACTER, JUST WAIT
3995 025704 000207                          RTS     PC
3996
```

```
3998                                     ;THIS SUBROUTINE ALLOWS FIELD ENGINEER TO SELECT ANY
3999                                     ;DATA PATTERN FOR OUTPUT MODULES
4000
4001
4002 025706 142777 000020 154274 SETPTN: BICB   #DBIT,@CSR
4003 025714 142777 000004 154266         BICB   #GBIT,@CSR
4004 025722 104401 036600         1$:    TYPE   ,MASS19           ;SELECT A DATA PATTERN
4005 025726 104410
4006 025730 012637 001140         RDOCT
4007 025734 113777 001140 154230         MOV    (SP)+,$GDDAT      ;;POP STACK INTO $GDDAT
4008 025742 005237 002172         MOVB   $GDDAT,@TADDR    ;OUTPUT PATTERN TO MODULE
4009 025746 005337 002170         INC    TADDR
4010 025752 001363
4011 025754 000207         DEC    BYTNUM
4012                                     BNE    1$
4013                                     RTS    PC
4014
4015                                     ;THIS SUBROUTINE IS USED IN FIELD TEST TO CONTINUOUSLY
4016                                     ;MONITOR DATA FROM INPUT MODULE.DATA IS PRINTED ONLY
4017                                     ;DURING FIRST PASS UNLESS THERE IS A CHANGE IN THE DATA
4018                                     MONDAT: CLR    R1           ;CLEAR PASS REGISTER
4019 025756 005001 002170         6$:    MOV    BYTNUM,R0      ;# OF BYTES PER MODULE
4020 025760 013700
4021 025764 005300         5$:    DEC    R0
4022 025766 117737 154200 001140 1$:    MOVB   @TADDR,$GDDAT    ;STORE DATA
4023 025774 005701
4024 025776 001016         TST    R1           ;IS IT FIRST PASS?
4025 026000 117760 154166 002154 2$:    BNE    4$           ;NO
4026 026006 013746 002172         MOVB   @TADDR,DBUFF(R0)  ;STORE DATA IN TABLE
4027 026012 104402
4028 026014 104401 037122         MOV    TADDR,-(SP)    ;;SAVE TADDR FOR TYPEOUT
4029 026020 013746 001140         TYPOC
4030 026024 104402
4031 026026 104401 035646         TYPE   ,MASS27      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4032 026032 000404
4033 026034 123760 001140 002154 4$:    MOV    $GDDAT,-(SP)  ;;COLON & TAB
4034 026042 001356
4035 026044 005237 002172         TYPOC
4036 026050 005700
4037 026052 001344
4038 026054 163737 002170 002172 3$:    TYPE   ,MASS0      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4039 026062 004737 025502
4040 026066 012701 000001
4041 026072 000732
4042                                     BR    3$           ;CR & LF
4043                                     BR    3$
4044                                     CMPB  $GDDAT,DBUFF(R0)  ;ANY CHANGE IN DATA ?
4045                                     BNE   2$           ;YES
4046                                     INC   TADDR        ;GO TO NEXT BYTE IN MODULE
4047                                     TST   R0           ;LAST BYTE IN MODULE ?
4048                                     BNE   5$           ;NO
4049                                     SUB   BYTNUM,TADDR    ;RESTORE ADDRESS OF MUT
4050                                     JSR   PC,CNTRC      ;TYPE CONTROL C TO RETURN TO MONITOR
4051                                     MOV   #1,R1         ;NEW PASS
4052                                     BR    6$
```



```
4038
4039 ;THIS SUBROUTINE IS USED TO WRAP AROUND DATA 'LOOP TEST'
4040
4041
4042 026074 004737 016652          LOPTST: JSR    PC,KLEER
4043 026100 020127 000141          CMP     R1,#141 ;IS IT M5010?
4044 026104 001131                  BNE     1$ ;YES
4045 026106 012737 000125 001140   MOV     #125,$GDDAT
      026114 012737 000004 002170   MOV     #4,BYTNUM
      026122 004737 025162          JSR     PC,SETBYT ;OUTPUT DATAT PATTERN
      026126 005037 002150          CLR     YLOOP ;WAIT
      026132 005237 002150          64$: INC     YLOOP
      026136 023727 002150 001777   CMP     YLOOP,#1777
      026144 001372                  BNE     64$
      026146 012737 000004 002170   MOV     #4,BYTNUM
      026154 004737 025564          JSR     PC,CHKBYT ;COMPARE DATA
      026160 104021                  ERROR!21
4046 026162 012737 000252 001140   MOV     #252,$GDDAT
      026170 012737 000004 002170   MOV     #4,BYTNUM
      026176 004737 025162          JSR     PC,SETBYT ;OUTPUT DATAT PATTERN
      026202 005037 002150          CLR     YLOOP ;WAIT
      026206 005237 002150          65$: INC     YLOOP
      026212 023727 002150 001777   CMP     YLOOP,#1777
      026220 001372                  BNE     65$
      026222 012737 000004 002170   MOV     #4,BYTNUM
      026230 004737 025564          JSR     PC,CHKBYT ;COMPARE DATA
      026234 104021                  ERROR!21
4047 026236 012737 000377 001140   MOV     #377,$GDDAT
      026244 012737 000004 002170   MOV     #4,BYTNUM
      026252 004737 025162          JSR     PC,SETBYT ;OUTPUT DATAT PATTERN
      026256 005037 002150          CLR     YLOOP ;WAIT
      026262 005237 002150          66$: INC     YLOOP
      026266 023727 002150 001777   CMP     YLOOP,#1777
      026274 001372                  BNE     66$
      026276 012737 000004 002170   MOV     #4,BYTNUM
      026304 004737 025564          JSR     PC,CHKBYT ;COMPARE DATA
      026310 104021                  ERROR!21
4048 026312 012737 000000 001140   MOV     #0,$GDDAT
      026320 012737 000004 002170   MOV     #4,BYTNUM
      026326 004737 025162          JSR     PC,SETBYT ;OUTPUT DATAT PATTERN
      026332 005037 002150          CLR     YLOOP ;WAIT
      026336 005237 002150          67$: INC     YLOOP
      026342 023727 002150 001777   CMP     YLOOP,#1777
      026350 001372                  BNE     67$
      026352 012737 000004 002170   MOV     #4,BYTNUM
      026360 004737 025564          JSR     PC,CHKBYT ;COMPARE DATA
      026364 104021                  ERROR!21
4049 026366 000530                  BR     2$
4050 026370          1$: MOV     #125,$GDDAT
      026370 012737 000125 001140   MOV     #2,BYTNUM
      026376 012737 000002 002170   JSR     PC,SETBYT ;OUTPUT DATAT PATTERN
      026404 004737 025162          CLR     YLOOP ;WAIT
      026410 005037 002150          68$: INC     YLOOP
      026414 005237 002150          CMP     YLOOP,#1777
      026420 023727 002150 001777   BNE     68$
      026426 001372                  MOV     #2,BYTNUM
      026430 012737 000002 002170
```

```
026436 004737 025564 JSR PC,CHKBYT ;COMPARE DATA
026442 104021 ERROR!21
4051 026444 012737 000252 001140 MOV #252,$GDDAT
026452 012737 000002 002170 MOV #2,BYTNUM
026460 004737 025162 JSR PC,SETBYT ;OUTPUT DATAT PATTERN
026464 005037 002150 CLR YLOOP ;WAIT
026470 005237 002150 69$: INC YLOOP
026474 023727 002150 001777 CMP YLOOP,#1777
026502 001372 BNE 69$
026504 012737 000002 002170 MOV #2,BYTNUM
026512 004737 025564 JSR PC,CHKBYT ;COMPARE DATA
026516 104021 ERROR!21
4052 026520 012737 000377 001140 MOV #377,$GDDAT
026526 012737 000002 002170 MOV #2,BYTNUM
026534 004737 025162 JSR PC,SETBYT ;OUTPUT DATAT PATTERN
026540 005037 002150 CLR YLOOP ;WAIT
026544 005237 002150 70$: INC YLOOP
026550 023727 002150 001777 CMP YLOOP,#1777
026556 001372 BNE 70$
026560 012737 000002 002170 MOV #2,BYTNUM
026566 004737 025564 JSR PC,CHKBYT ;COMPARE DATA
026572 104021 ERROR!21
4053 026574 012737 000000 001140 MOV #0,$GDDAT
026602 012737 000002 002170 MOV #2,BYTNUM
026610 004737 025162 JSR PC,SETBYT ;OUTPUT DATAT PATTERN
026614 005037 002150 CLR YLOOP ;WAIT
026620 005237 002150 71$: INC YLOOP
026624 023727 002150 001777 CMP YLOOP,#1777
026632 001372 BNE 71$
026634 012737 000002 002170 MOV #2,BYTNUM
026642 004737 025564 JSR PC,CHKBYT ;COMPARE DATA
026646 104021 ERROR!21
4054 026650 000207 2$: RTS PC
4055
4056
```



```

4058 ;THIS SUBROUTINE SETS UP ADDRESSES FOR A/D A014
4059
4060 026652 013737 002172 003004 ADADDR: MOV TADDR,LBYTE
4061 026660 013746 002172 MOV TADDR,-(SP) ;:PUSH TADDR ON STACK
4062 026664 013737 002172 003006 MOV TADDR,HBYTE
4063 026672 005237 003006 INC HBYTE
4064 026676 013737 003006 003000 MOV HBYTE,STAT1
4065 026704 005237 003000 INC STAT1
4066 026710 013737 003000 003002 MOV STAT1,STAT2
4067 026716 005237 003002 INC STAT2
4068 026722 013737 002172 002176 MOV TADDR,TBADDR
4069 026730 013746 000004 MOV ERRVEC,-(SP) ;:PUSH ERRVEC ON STACK
4070 026734 012737 027010 000004 MOV #1$,ERRVEC
4071 026742 105777 154036 TSTB @LBYTE
4072 026746 005237 002172 INC TADDR
4073 026752 105777 154030 TSTB @HBYTE
4074 026756 005237 002172 INC TADDR
4075 026762 105777 154012 TSTB @STAT1
4076 026766 005237 002172 INC TADDR
4077 026772 105777 154004 TSTB @STAT2
4078 026776 012637 000004 2$: MOV (SP)+,ERRVEC ;:POP STACK INTO ERRVEC
4079 027002 012637 002172 MOV (SP)+,TADDR ;:POP STACK INTO TADDR
4080 027006 000207 RTS PC
4081
4082 027010 022626 1$: CMP (SP)+,(SP)+
4083 027012 104026 ERROR!26 ;REG DID NOT RESPOND
4084 027014 012737 005044 000004 MOV #TMOVEC,ERRVEC
4085 027022 005205 INC R5
4086 027024 000764 BR 2$
4087
4088
4089 027026 152777 000004 153154 GCODE: BISB #GBIT,@CSR ;LOAD GBITE WITH GENERIC CODE
4090 027034 117737 153740 002776 MOVB @STAT1,GBITE
4091 027042 042737 177400 002776 BIC #177400,GBITE
4092 027050 142777 000004 153132 BICB #GBIT,@CSR
4093 027056 000207 RTS PC
4094 ;THIS SUBROUTINE RUNS A TEST AND CHECKS SW13 AND SW14
4095 027060 005037 001210 SWLOOP: CLR $PASS
4096 027064 011637 002236 MOV (SP),XXX ;GET SUBR ADDR
4097 027070 017737 153142 002236 MOV @XXX,XXX ;ANOTHER LEVEL OF DEFERRED
4098 027076 004777 153134 1$: JSR PC,@XXX
4099 027102 005237 001210 INC $PASS
4100 027106 023737 001210 002246 CMP $PASS,PASCNT
4101 027114 001370 BNE 1$
4102 027116 032737 020000 000176 BIT #BIT13,SWREG
4103 027124 001005 BNE 2$
4104 027126 013746 002246 MOV PASCNT,-(SP) ;:SAVE PASCNT FOR TYPEOUT
027132 104402 TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
4105 027134 104401 033363 TYPE ,M11
4106 027140 032737 040000 000176 2$: BIT #BIT14,SWREG
4107 027146 001344 BNE SWLOOP
4108 027150 062716 000002 ADD #2,(R6) ;ADVANCE RETURN PC
4109 027154 000207 RTS PC

```

```

4128
4129 032074      103      102      111  EM1:  .ASCIZ  /CBIT NOT CLEARING IOCM/
4130 032123      111      117      103  EM2:  .ASCIZ  /IOCM TEST/<15><12>
4131 032136      103      123      122  EM2Y: .ASCIZ  /CSR DATA ERROR WHEN TESTING /
4132 032172      040      040      102  EM2X: .ASCIZ  / BIT /<15><12>
4133 032204      120      103      011  DH1:  .ASCIZ  /PC TST# GDDAT BDDAT PASS/
4134 032235      111      117      103  EM3:  .ASCIZ  /IOCM IS NOT RESPONDING/<15><12>
4135 032266      104      102      125  EM4:  .ASCIZ  /DBUS BIT IS FAILING/<15><12>
4136 032314      116      117      040  EM5:  .ASCIZ  /NO INTERRUPT /<15><12>
4137 032334      120      103      011  DH5:  .ASCIZ  /PC TST# MODULE ADDR PASS/
4138 032367      127      122      117  EM6:  .ASCIZ  /WRONG PATTERN IN IAR AFTER INTERRUPT/<15><12>
4139 032436
4140 032436      104      101      124  EM7:  .ASCIZ  /DATA ERROR/<15><12>
4141 032453      120      103      011  DH7:  .ASCIZ  /PC MODULE ADDRESS TST# GDDAT BDDAT PASS/
4142 032523      124      105      123  EM10: .ASCIZ  /TEST ABORTED, IOCM ERROR/<15><12>
4143 032556      104      101      124  EM11: .ASCIZ  /DATA NOT CLEAR AFTER 10 SEC/<15><12>
4144 032614      111      116      124  EM12: .ASCIZ  /INTERRUPT TOO LATE /<15><12>
4145 032644      122      111      106  EM13: .ASCIZ  /RIF BIT NOT CLEARING COS REGISTERS /<15><12>
4146 032712      116      117      040  EM14: .ASCIZ  /NO INTERRUPT IN MAINTENANCE MODE/<15><12>
4147 032755      122      111      106  EM15: .ASCIZ  /RIF BIT ISN'T CLEARING INTERRUPT/<15><12>
4148 033020      103      114      105  EM16: .ASCIZ  /CLEARING DBIT ISN'T CLEARING CSR/<15><12>
4149 033063      103      117      123  EM20: .ASCIZ  /COS REGISTER DATA ERROR/<15><12>
4150 033115      114      117      117  EM21: .ASCIZ  /LOOP ERROR ON 6010-5010 OR 5011/<15><12>
4151 033157      120      103      011  DH21: .ASCIZ  /PC GDDAT BDDAT PASS/
4152 033203      101      104      104  EM22: .ASCIZ  /ADDRESS TEST WITH MBIT SET ISN'T WORKING /<15><12>
4153 033257      125      116      101  EM23: .ASCIZ  /UNABLE TO CLEAR INTERRUPT /<15><12>
4154 033314      120      103      011  DH23: .ASCIZ  /PC TST# IAR/
4155 033330      116      015      012  M1:   .ASCIZ  /N/<15><12>
4156 033334      131      015      012  M0:   .ASCIZ  /Y/<15><12>
4157 033340      122      000
4158 033342      107      040      000  M2:   .ASCIZ  /R/
4159 033345      124      040      000  M3:   .ASCIZ  /G /
4160 033350      104      040      000  M4:   .ASCIZ  /T /
4161 033353      115      040      000  M5:   .ASCIZ  /D /
4162 033356      105      040      000  M6:   .ASCIZ  /M /
4163 033361      011      000
4164 033363      040      120      101  M10:  .ASCIZ  /
4165 033407      115      117      104  M11:  .ASCIZ  / PASSES COMPLETED/<15><12>
4166 033455      015      012      124  EM24: .ASCIZ  /MODULE UNDER TEST IS NOT RESPONDING/<15><12>
4167 033477      123      040      040  M13:  .ASCIZ  <15><12>/TEST OPTIONS:/<15><12>
4168 033520      104      040      040  M14:  .ASCIZ  /S SYSTEM TEST/<15><12>
4169 033544      101      040      040  M15:  .ASCIZ  /D DIGITAL MODULE/<15><12>
4170 033567      115      040      040  M16:  .ASCIZ  /A ANALOG MODULE/<15><12>
4171 033623      130      040      040  M17:  .ASCIZ  /M MAP OF DBUS INTERFACES/<15><12>
4172 033642      111      040      040  M18:  .ASCIZ  /X EXERCIZER/<15><12>
4173 033661      040      040      040  M19:  .ASCIZ  /I IOCM TEST/<15><12>
4174 033670      124      040      040  M20:  .ASCIZ  / ?/<15><12>
4175 033707      061      060      060  M21:  .ASCIZ  /T SET SWREG/<15><12>
4176 033737      064      060      060  M22:  .ASCIZ  /100000 HALT ON ERROR/<15><12>
4177 033766      062      060      060  .ASCIZ  /40000 LOOP ON TEST/<15><12>
4178 034033      061      060      060  .ASCIZ  /20000 INHIBIT ERROR & EOP PRINT/<15><12>
4179 034064      015      012      123  .ASCIZ  /10000 LOOP ON ERROR/<15><12>
4180 034076      106      040      040  M23:  .ASCIZ  <15><12>/SWREG =/
4181 034116      127      040      040  M24:  .ASCIZ  /F FIELD TEST/<15><12>
4182 034135      114      040      040  M25:  .ASCIZ  /W LOOP TEST/<15><12>
4183 034167      116      125      115  M26:  .ASCIZ  /L LOAD ITERATION COUNT/<15><12>
4184 034224      101      130      130  M27:  .ASCIZ  /NUMBER OF ITERATIONS(OCT) = /
M28:  .ASCIZ  /AXXX ANALOG MODULE TEST (A014,A630)/<15><12>

```



```
4185 034273 114 111 116 EM25: .ASCIZ /LINE CLOCK IS NOT INTERRUPTING/<15><12>
4186 034334 015 012 102 M30: .ASCII <15><12>/BEFORE RUNNING THIS TEST MAKE SURE CUSTOMER WIRES ARE DISCONNECTED/
4187 034440 015 012 127 .ASCIZ <15><12>/WHEN READY TYPE CR/
4188 034465 015 012 123 M84: .ASCIZ <15><12> /SELECTED MUX DIDN'T RESPOND/
4189 034523 015 012 124 EM54: .ASCIZ <15><12>/TOO MANY ERRORS, TEST ABORTED/<15><12>
4190 034565 015 012 114 EM53: .ASCII <15><12>/LINEARITY TEST ERROR/<15><12>
4191 034615 105 122 122 .ASCIZ /ERRPC ADDRESS TSTPNT MIN MAX WAS (# OF CONVERS)/<15><12>
4192 034677 015 012 115 MASS55: .ASCII <15><12>/MONOTONICITY ERROR BY TWO BITS OR MORE/<15><12>
4193 034751 105 122 122 .ASCIZ /ERRPC ADDRESS GDDAT BDDAT RAMP (UP=0,DOWN=1)/<15><12>
4194 035030 015 012 105 EM55: .ASCIZ <15><12>/ERROR BIT SET AFTER SETTING CHANNEL REG./
4195 035103 015 012 124 EM56: .ASCIZ <15><12>%TEST RAMP IN A/D ISN'T WORKING%
4196 035144 126 105 122 MASS51: .ASCIZ /VERIFY IT BY RUNNING LOGIC TEST/<15><12>
4197 035206 015 012 125 EM57: .ASCIZ <15><12>/UNKNOWN GENERIC CODE/
4198 035235 105 122 122 DH57: .ASCIZ/ERRPC ADDRESS GCODE/
4199 035261 015 012 115 EM60: .ASCIZ <15><12>/MUX SELECT REGISTER ERROR/
4200 035315 015 012 127 EM61: .ASCIZ <15><12>/WRONG GENERIC CODE WITH MUX SELECTED/
4201 035364 015 012 123 EM63: .ASCIZ <15><12>/SELECTED MUX DID'T RESPOND/
4202 035421 105 122 122 DH61: .ASCIZ/ERRPC MODULE ADDRESS MUX# TST# GDDAT BDDAT/
4203 035474 105 122 122 DH62: .ASCIZ/ERRPC MODULE ADDRESS MUX# TST#/
4204 035533 015 012 122 EM62: .ASCIZ <15><12>/RIF BIT NOT CLEARING ERROR BIT/
4205 035574 015 012 115 EM73: .ASCIZ <15><12>/MUX REGISTER NOT CLEAR AFTER CONVERSION/
4206 035646 015 012 000 MASS0: .ASCIZ <15><12>
4207 035651 101 104 104 MASS2: .ASCIZ /ADDRESS /
4208 035663 040 040 040 MASS3: .ASCIZ / M5010 NONISOLATED 32 BIT DC INPUT/<15><12>
4209 035732 040 040 040 MASS4: .ASCIZ / M5011 NONISOLATED 16 BIT DC INPUT/<15><12>
4210 036001 040 040 040 MASS5: .ASCIZ / M5012 ISOLATED 16 BIT DC INPUT/<15><12>
4211 036045 040 040 040 MASS6: .ASCIZ / M5013 8 BIT AC INPUT/<15><12>
4212 036077 040 040 040 MASS7: .ASCIZ / M6010 NONISOLATED 32 BIT DC OUTPUT/<15><12>
4213 036147 040 040 040 MASS8: .ASCIZ / M6011 16 BIT ONESHOT OUTPUT/<15><12>
4214 036210 040 040 040 MASS9: .ASCIZ / M6012 ISOLATED 8 BIT DC OUTPUT/<15><12>
4215 036254 040 040 040 MASS10: .ASCIZ / M6013 8 BIT AC OUTPUT/<15><12>
4216 036307 040 040 040 MASS11: .ASCIZ / UNKNOWN GENERIC CODE /<15><12>
4217 036343 015 012 127 MASS12: .ASCIZ <15><12>/WHICH MODULE OR TEST OPTION (H=HELP)? /
4218 036415 015 012 124 MASS13: .ASCIZ <15><12>/TYPE ADDRESS OF MUT /
4219 036447 111 116 124 MASS14: .ASCIZ /INTERRUPT TOO LATE/<15><12>
4220 036474 015 012 105 MASS15: .ASCIZ <15><12>/END OF MODULE TEST/<15><12>
4221 036523 105 116 104 MASS16: .ASCIZ /END OF IOCM TEST/<15><12>
4222 036546 105 116 104 MASS17: .ASCIZ /END OF AUTO TEST/<15><12>
4223 036572 040 040 040 MASS18: .ASCIZ / /<15><12>
4224 036600 124 131 120 MASS19: .ASCIZ /TYPE ONE BYTE DATA PATTERN /
4225 036637 124 131 120 MASS20: .ASCIZ /TYPE CONTROL-C TO RETURN TO MONITOR/<15><12>
4226 036705 124 131 120 MASS21: .ASCIZ /TYPE ADDRESS OF OUTPUT MUT/
4227 036740 124 131 120 MASS22: .ASCIZ /TYPE ADDRESS OF INPUT MUT/
4228 036772 103 117 116 MASS23: .ASCIZ /CONNECT OUTPUT TO INPUT MODULE/<15><12>
4229 037033 105 116 104 MASS25: .ASCIZ /END OF LOOP TEST/<15><12>
4230 037056 127 122 117 MASS26: .ASCIZ /WRONG MODULE-OUTPUT MUST BE M6010/<15><12>
4231 037122 072 011 000 MASS27: .ASCIZ /: /
4232 037125 127 122 117 MASS28: .ASCIZ /WRONG MODULE-INPUT MUST BE EITHER M5010 OR M5011/<15><12>
4233 037210 040 040 040 MASS29: .ASCIZ / A630 FOUR CHANNEL DAC MODULE /<15><12>
4234 037253 040 040 040 MASS30: .ASCIZ % A014 A/D CONVERTER -SINGLE ENDED%<15><12>
4235 037321 040 040 040 MASS31: .ASCIZ % A014 A/D CONVERTER - DIFFERENTIAL MODE%<15><12>
4236 037375 011 115 125 MASS32: .ASCIZ / MUX# /
4237 037404 011 011 101 MASS33: .ASCIZ / A156 - SINGLE-ENDED/<15><12>
4238 037434 011 011 101 MASS34: .ASCIZ / A156 - DIFFERENTIAL MODE/<15><12>
4239 037471 011 011 101 MASS35: .ASCIZ / A157/<15><12>
4240 037502 015 012 104 MASS36: .ASCII <15><12> /D - LOGIC TEST/<15><12>
4241 037524 103 040 055 .ASCII /C - CALIBRATION/<15><12>
```



```
4242 037545      115      040      055      .ASCII /M - MONOTONICITY TEST/<15><12>
4243 037574      130      040      055      .ASCII /X - MUX TEST/<15><12>
4244 037612      114      040      055      .ASCIZ  /L - LINEARITY TEST/<15><12>
4245 037637      015      012      127      MASS37: .ASCIZ  <15><12> /WHICH MUX # DO YOU WANT TO TEST? /
4246 037703      015      012      127      MASS38: .ASCIZ  <15><12> %WHICH A/D CHANNEL YOU WANT TO CALIBRATE? %
4247 037757      015      012      124      MASS39: .ASCIZ  <15><12> /TOO HIGH CHANNEL FOR DIFF MODE/<15><12>
4248 040022      015      012      124      MASS40: .ASCIZ  <15><12> /TOO HIGH CHANNEL FOR SINGLE-ENDED MODE/<15><12>
4249 040075      015      012      127      MASS41: .ASCIZ  <15><12>/WHAT GAIN YOU WANT TO TEST? (1,10,20,50,100,200,1000)/
4250 040165      127      122      117      MASS42: .ASCIZ  /WRONG GAIN SELECTED/<15><12>
4251 040213      103      117      116      MASS43: .ASCIZ  /CONNECT VOLTAGE SOURCE (CR)/
4252 040250      015      012      117      MASS44: .ASCIZ  <15><12> /OCTAL AVERAGE VOLTAGE(MV)/<15><12>
4253 040310      125      116      105      MASS45: .ASCIZ  /UNEXPECTED TIME-OUT TRAP-LAST PC BEFORE TRAP = /
4254 040370      015      012      127      MASS46: .ASCIZ  <15><12> /WHICH TEST (D,C,M,L,X,H=HELP)? /
4255 040432      040      040      040      MASS48: .ASCIZ  / M5012-YA 16 BIT TTL INPUT/<15><12>
4256 040471      040      040      040      MASS49: .ASCIZ  / M6010-YA 32 BIT TTL OUTPUT/<15><12>
4257 040531      116      117      040      MASS50: .ASCIZ  %NO I/O MODULE PRESENT%<15><12>
4258 040561      015      012      101      EM27:   .ASCIZ  <15><12>/A REGISTER DID NOT CLEAR ON INIT /
4259 040625      015      012      101      EM26:   .ASCIZ  <15><12>/A REGISTER DID NOT RESPOND WHEN ADDRESSED/
4260 040701      015      012      040      EM33:   .ASCIZ  <15><12>/ CHANNEL COMPARATOR ERROR/
4261 040735      015      012      101      DH33:   .ASCIZ  <15><12>/ADDRESS          CHNUM  STBITS  CH0      CH1      CH2      CH3/
4262 041004      015      012      115      EM34:   .ASCIZ  <15><12>/MORE THEN ONE CHANNEL FAILS DURING COMPARATOR TEST/
4263 041071      015      012      105      DH34:   .ASCIZ  <15><12>/ERRPC MODULE ADDRESS STBITS  CH0      CH1      CH2      CH3/
4264 041147      015      012      105      DH40:   .ASCIZ  <15><12>/ERRPC TST#   GENERIC /
4265 041175      015      012      104      EM30:   .ASCIZ  <15><12>/DATA ERROR DURING REGISTER TEST/
4266 041237      015      012      105      DH30:   .ASCIZ  <15><12>/ERRPC MODULE GDDATA BDDATA REGADR /
4267 041304      123      124      101      EM35:   .ASCIZ  /STATUS REGISTER ERROR AFTER INITIALIZE/<15><12>
4268 041355      102      125      123      EM36:   .ASCIZ  /BUSY BIT NOT CLEAR AFTER CONVERSION/<15><12>
4269 041423      105      122      122      EM37:   .ASCIZ  /ERROR BIT SET AFTER CONVERSION/<15><12>
4270 041464      103      117      116      EM40:   .ASCIZ  /CONVERSION NOT DONE AFTER .5MSEC/<15><12>
4271 041527      116      117      040      EM41:   .ASCIZ  /NO INTERRUPT AFTER CONVERSION DONE/
4272 041572      103      110      101      EM42:   .ASCIZ  /CHANNEL SELECTION REGISTER ERROR/
4273 041633      105      122      122      EM43:   .ASCIZ  /ERROR BIT NOT SET AFTER SETTING WRONG GAIN/<15><12>
4274 041710      105      122      122      EM44:   .ASCIZ  /ERROR BIT WILL NOT INTERRUPT/<15><12>
4275 041747      105      122      122      EM45:   .ASCIZ  /ERROR BIT NOT SET AFTER WRONG GAIN/<15><12>
4276 042014      115      101      111      EM46:   .ASCIZ  /MAINTANCE REFERENCE VOLTAGE TEST FAILED/<15><12>
4277 042066      122      101      115      EM47:   .ASCIZ  /RAMP IS NOT WORKING/<15><12>
4278 042114      104      101      124      EM50:   .ASCIZ  /DATA IS NOT ZERO WITH D BIT SET/<15><12>
4279 042156      015      012      104      M80:    .ASCIZ  <15><12>/DBIT DOES NOT DISABLE ALL OUTPUTS /
4280 042223      015      012      101      M81:    .ASCIZ  <15><12>/ADJUSTMENT OF VOLTAGE OUTPUT/
4281 042262      015      012      104      M82:    .ASCIZ  <15><12>/DO YOU WANT TO ADJUST CURRENT OUTPUT (Y-N)? /
4282 042342      015      012      105      M83:    .ASCIZ  <15><12>/END OF CALIBRATION/
4283 042367      015      012      104      DANGC:  .ASCIZ  <15><12> /DAC DID NOT RESPOND WITH GENERIC CODE 261/
4284 042443      015      012      107      GAINMG: .ASCII  <15><12> /GAIN X INPUT VOLTAGE=VOLTAGE READING/
4285 042511      015      012      107      .ASCIZ  <15><12> /GAIN = /
4286 042523      015      012      123      DANSEL: .ASCIZ  <15><12> /SELECT TEST (A,T,D OR H=HELP)/
4287 042563      015      012      124      DACHN:  .ASCIZ  <15><12> /TYPE CHANNEL # DESIRED (0,1,2,3) /
4288 042631      015      012      101      TEXT:   .ASCII  <15><12> /A - CALIBRATION TEST/
4289 042657      015      012      124      .ASCII  <15><12> /T - INTERNAL COMPARATOR TEST/
4290 042715      015      012      104      .ASCIZ  <15><12> /D - D-BIT TEST /
4291
4292 042737      015      012      104      CTXTV:  .ASCIZ  <15><12>/DO YOU WANT TO ADJ REFER. VOLTAGE (Y-N)? /
4293
4294
4295 043014      015      012      122      .ASCIZ  <15><12> /R66 - CALIB POT FOR CH1 CURRENT ZERO/<15><12>
4296 043065      015      012      122      .ASCII  <15><12> /R80 - CALIB POT FOR CH2 CURRENT GAIN/
4297
4298 043133      015      012      123      CM1:    .ASCIZ  <15><12> /STEP 1:          +10.240V TOLERANCE 2MV (CR)/
```



```

4299 043202    015    012    123 CM2:  .ASCIZ <15><12> /STEP 2:  5.120V  TOLERANCE 2MV (CR)/
4300 043252    015    012    123 CM3:  .ASCIZ <15><12> /STEP 1:  0.000V  TOLERANCE 5MV (CR)/
4301 043322    015    012    123 CM4:  .ASCIZ <15><12> /STEP 2:  +10.230V  TOLERANCE 40MV (CR)/
4302
4303
4304 043375    015    012    123 CM7:  .ASCII <15><12> /STEP 2:  CURRENT OFFSET ADJ.  9.8MV  TOLER. 5MV/
4305 043455    015    012    040      .ASCIZ <15><12> /          OR 19.5 MICRO-AMPS.  TOLER. 10 MICRO-AMPS.  (CR)/
4306
4307 043547    015    012    123 CM9:  .ASCII <15><12> /STEP 1:  CURRENT GAIN ADJ.  TO +10.000V TOLER. 5MV/
4308 043631    015    012    040      .ASCIZ <15><12> /          OR 20.000MA TOLER. 10 MICRO-AMPS (CR)/
4309 043711    015    012    123 CM10: .ASCII <15><12> /STEP 2:  CURRENT OFFSET ADJ.  TO +2.000V TOLER. 5MV/
4310 043774    015    012    040      .ASCIZ <15><12> /          OR 4.000 MA.TOLER. 10 MICRO AMPS (CR)/
4311 044055    015    012    110 CM11: .ASCIZ <15><12>/HAS CALIB BEEN RE-VERIFIED (Y-N) /
4312 044121    015    012    104 CM12: .ASCIZ <15><12>/DO YOU WISH TO CALIB THE 4-20 MA RANGE (Y-N)? /
4313 044202    015    012    101 EM31: .ASCIZ <15><12>/ALL COMPARATOR STATUS BITS SHOULD BE SET/
4314 044255    015    012    101 EM32: .ASCIZ <15><12>/ALL COMPARATOR STATUS BITS SHOULD BE CLEAR/
4315 044332    015    012    120 DH31: .ASCIZ <15><12>/PC#  ADDRESS EXPECT WAS/
4316
4317 044364    001132  002166  002242 DT17:  .WORD  $ERRPC, $MUT,  TEMP1,  $TESTN, $GDDAT, $BDDAT, $PASS,
4318 044404    001132  002166  002200 DT20:  .WORD  $ERRPC, $MUT,  COSADR, $TESTN, $GDDAT, $BDDAT, $PASS,
4319 044424    001132  001206  002770 DT23:  .WORD  $ERRPC, $TESTN,  TEMP3,
4320 044434    001132  002166  002176 DT7:   .WORD  $ERRPC, $MUT,  TBADDR, $TESTN, $GDDAT, $BDDAT, $PASS,
4321 044454    001132  001206  001140 DT1:   .WORD  $ERRPC, $TESTN, $GDDAT, $BDDAT, $PASS,
4322 044470    001132  001206  002166 DT5:   .WORD  $ERRPC, $TESTN, $MUT,  TADDR, $PASS,
4323 044504    001132  001140  001142 DT21:  .WORD  $ERRPC, $GDDAT, $BDDAT, $PASS,
4324 044516    001132  002166  001140 DT30:  .WORD  $ERRPC, $MUT,  $GDDAT, $BDDAT,  TEMP,
4325 044532    001132  002172  002304 DT33:  .WORD  $ERRPC, TADDR,  CHNUM,  TEMP3,  ACOUNT, BCOUNT, ACOUNT, BCOUNT,
4326 044554    001132  002166  002172 DT34:  .WORD  $ERRPC, $MUT,  TADDR,  TEMP3,  ACOUNT, BCOUNT, ACOUNT, BCOUNT,
4327 044576    001132  002172  001140 DT51:  .WORD  $ERRPC, TADDR,  $GDDAT, $BDDAT,  LASTCN,
4328 044612    001132  002172  002754 DT52:  .WORD  $ERRPC, TADDR,  BLAST,  BCOUNT, CCOUNT, $BDDAT,
4329 044630    001132  002770  002766 DT57:  .WORD  $ERRPC, TEMP3,  TEMP2,
4330 044640    001132  002166  002176 DT61:  .WORD  $ERRPC, $MUT,  TBADDR,  MXNUM,  $TESTN, $GDDAT, $BDDAT,
4331 044660    001132  002166  002176 DT62:  .WORD  $ERRPC, $MUT,  TBADDR,  MXNUM,  $TESTN,
4332 044674    001132  002172  002770 DT31:  .WORD  $ERRPC, TADDR,  TEMP3,  TEMP,
4333 044706      000      000      000 DF51:  .BYTE  0,0,0,0
4334 044712      000      000      001 DF52:  .BYTE  0,0,1,1,1
4335      000001      .END

```


ABASE = 000000	ASWREG= 000000	CKICT 016302	DH5 032334	EM36 041355
ACDW1 = 000000	ATABL 001326	CLK 002226	DH57 035235	EM37 041423
ACDW2 = 000000	ATESTN= 000000	CLKADR 002206	DH61 035421	EM4 032266
ACOUNT 002352	ATOD 017152	CLKVC 002250	DH62 035474	EM40 041464
ACPUOP= 000000	AUNIT = 000000	CLKVCA 002252	DH7 032453	EM41 041527
ADADDR 026652	AUSWR = 000000	CLRINT 025414	DIGIT 005056	EM42 041572
ADCALB 017560	AUTO 005442	CM1 043133	DISPLA 001156	EM43 041633
ADDW0 = 000000	AVECT1= 000000	CM10 043711	DISPRE 000174	EM44 041710
ADDW1 = 000000	AVECT2= 000000	CM11 044055	DMUT 002202	EM45 041747
ADDW10= 000000	AVRSV 003062	CM12 044121	DSWR = 177570	EM46 042014
ADDW11= 000000	AVRTBL 003064	CM2 043202	DTST 014636	EM47 042066
ADDW12= 000000	BASE 002224	CM3 043252	DT1 044454	EM5 032314
ADDW13= 000000	BCOUNT 002354	CM4 043322	DT17 044364	EM50 042114
ADDW14= 000000	BEG 023320	CM7 043375	DT20 044404	EM53 034565
ADDW15= 000000	BITSET 025300	CM9 043547	DT21 044504	EM54 034523
ADDW2 = 000000	BIT0 = 000001	CNTRC 025502	DT23 044424	EM55 035030
ADDW3 = 000000	BIT00 = 000001	CONV 002364	DT30 044516	EM56 035103
ADDW4 = 000000	BIT01 = 000002	CONVCT 002370	DT31 044674	EM57 035206
ADDW5 = 000000	BIT02 = 000004	CONV7 024626	DT33 044532	EM6 032367
ADDW6 = 000000	BIT03 = 000010	COSADR 002200	DT34 044554	EM60 035261
ADDW7 = 000000	BIT04 = 000020	COUNT 025406	DT5 044470	EM61 035315
ADDW8 = 000000	BIT05 = 000040	CR = 000015	DT51 044576	EM62 035533
ADDW9 = 000000	BIT06 = 000100	CRLF = 000200	DT52 044612	EM63 035364
ADEVCT= 000000	BIT07 = 000200	CRTST 025662	DT57 044630	EM7 032436
ADEVM = 000000	BIT08 = 000400	CSR 002210	DT61 044640	EM73 035574
ADGAN 021350	BIT09 = 001000	CTXTV 042737	DT62 044660	ERROR = 104000
ADLOG 020452	BIT1 = 000002	DACHN 042563	DT7 044434	ERRVEC= 000004
ADMON 017252	BIT10 = 002000	DACMON 013610	EBIT = 000100	EXERC 016752
ADRET 017222	BIT11 = 004000	DACSTR 013560	ECOUNT 002372	FBIT = 000200
ADTBIT 021436	BIT12 = 010000	DACTST 014516	EMTVEC= 000030	FIELD 006612
ADTST 017064	BIT13 = 020000	DAGTST 016504	EM1 032074	F5010 016756
AENV = 000000	BIT14 = 040000	DALLY 016730	EM10 032523	F5011 016756
AENVM = 000000	BIT15 = 100000	DANGC 042367	EM11 032556	F5012 016770
AFATAL= 000000	BIT2 = 000004	DANSEL 042523	EM12 032614	F5013 017002
AFLAG 002240	BIT3 = 000010	DATAL0 002366	EM13 032644	F6010 017114
AMADR1= 000000	BIT4 = 000020	DATST 014522	EM14 032712	F6011 017050
AMADR2= 000000	BIT5 = 000040	DBIT = 000020	EM15 032755	F6012 017034
AMADR3= 000000	BIT6 = 000100	DBUFF 002154	EM16 033020	F6013 017034
AMADR4= 000000	BIT7 = 000200	DCAMON 013524	EM17 032436	GAIN 003032
AMAMS1= 000000	BIT8 = 000400	DCHAN 002320	EM2 032123	GAINMG 042443
AMAMS2= 000000	BIT9 = 001000	DCMN 013634	EM2X 032172	GAINTB 003010
AMAMS3= 000000	BLAST 002754	DCOUNT 002362	EM2Y 032136	GBIT = 000004
AMAMS4= 000000	BPTVEC= 000014	DCOUT 013620	EM20 033063	GBITE 002776
AMSGAD= 000000	BSYCON 020604	DCTST 014614	EM21 033115	GCODE 027026
AMSGLG= 000000	BTABL 001566	DDBIT 014100	EM22 033203	GENCOD 025224
AMSGTY= 000000	BYTNUM 002170	DDISP = 177570	EM23 033257	GENER 002026
AMTYP1= 000000	CBIT = 000002	DECCHN 016550	EM24 033407	GODN 016434
AMTYP2= 000000	CCOUNT 002356	DF51 044706	EM25 034273	GOUP 016416
AMTYP3= 000000	CHKBYT 025564	DF52 044712	EM26 040625	GTADRS 014036
AMTYP4= 000000	CHNSEL 021250	DH1 032204	EM27 040561	HBYTE 003006
ANSW 002360	CHNUM 002304	DH21 033157	EM3 032235	HDATA0 002332
APASS = 000000	CHSEL 021102	DH23 033314	EM30 041175	HDATA1 002336
APRIOR= 000000	CH0 002306	DH30 041237	EM31 044202	HDATA2 002342
APTCSU= 000040	CH1 002310	DH31 044332	EM32 044255	HDATA3 002346
APTENV= 000001	CH2 002312	DH33 040735	EM33 040701	HT = 000011
APTSIZ= 000200	CH3 002314	DH34 041071	EM34 041004	IAR 002212
APTSPO= 000100	CH4 002316	DH40 041147	EM35 041304	ICHAN 002326

INTCOM	015256	MASS41	040075	M6011B	012204	STKLMT=	177774	TST10	010250
INTDAT	002244	MASS42	040165	M6012	011526	ST1SAV	003054	TST11	010344
INTLD	016214	MASS43	040213	M6013	011526	ST2SAV	003056	TST12	010570
IOCM	007474	MASS44	040250	M7	033356	SWLOOP	027060	TST13	010672
IOTVEC=	000020	MASS45	040310	M80	042156	SWR	001154	TST14	011334
KBINT	025556	MASS46	040370	M81	042223	SWREG	000176	TST15	011526
KBVEC	002302	MASS48	040432	M82	042262	SWREGS	005414	TST16	011720
KLEER	016652	MASS49	040471	M83	042342	SW0	= 000001	TST17	012216
KOUNT	002350	MASS5	036001	M84	034465	SW00	= 000001	TST2	007630
LAST	002756	MASS50	040531	NOCLK	025404	SW01	= 000002	TST20	013046
LASTCN	002760	MASS51	035144	NORAMP	002230	SW02	= 000004	TST21	014224
LBYTE	003004	MASS55	034677	OUT	023676	SW03	= 000010	TST22	015256
LDATA0	002330	MASS6	036045	PASCNT	002246	SW04	= 000020	TST23	020452
LDATA1	002334	MASS7	036077	PATT	002160	SW05	= 000040	TST3	007674
LDATA2	002340	MASS8	036147	PIRQ	= 177772	SW06	= 000100	TST4	007740
LDATA3	002344	MASS9	036210	PIRQVE=	000240	SW07	= 000200	TST5	010070
LF =	000012	MBIT =	000040	PRO	= 000000	SW08	= 000400	TST6	010140
LINEAR	022100	MOD	002254	PR1	= 000040	SW09	= 001000	TST7	010204
LOGIC	006302	MODUL	002114	PR2	= 000100	SW1	= 000002	TYPDS =	104405
LONGIN	002204	MONDAT	025756	PR3	= 000140	SW10	= 002000	TYPE =	104401
LOOP	006362	MONIT	004546	PR4	= 000200	SW11	= 004000	TYPOC =	104402
LOPTST	026074	MUT	002062	PR5	= 000240	SW12	= 010000	TYPON =	104404
MAPE	007012	MUX1	023710	PR6	= 000300	SW13	= 020000	TYPOPT	004476
MASS0	035646	MXNUM	002152	PR7	= 000340	SW14	= 040000	TYPOS =	104403
MASS10	036254	M0	033334	PS	= 177776	SW15	= 100000	UPREG	014476
MASS11	036307	M1	033330	PSW	= 177776	SW2	= 000004	VCHAN	002324
MASS12	036343	M10	033361	PWRVEC=	000024	SW3	= 000010	VECT0	002214
MASS13	036415	M11	033363	RAMPST	023244	SW4	= 000020	VECT0A	002216
MASS14	036447	M13	033455	RAMP1	002374	SW5	= 000040	VECT1	002220
MASS15	036474	M14	033477	RAMP2	002514	SW6	= 000100	VECT1A	002222
MASS16	036523	M15	033520	RAMP3	002634	SW7	= 000200	XCHAN	002322
MASS17	036546	M16	033544	RBIT	= 000001	SW8	= 000400	XXDP	002164
MASS18	036572	M17	033567	RDCHR	= 104406	SW9	= 001000	XXX	002236
MASS19	036600	M18	033623	RDLIN	= 104407	TABLE	025250	YLOOP	002150
MASS2	035651	M19	033642	RDOCT	= 104410	TADDR	002172	\$APTHD	001100
MASS20	036637	M2	033340	RERROR	002234	TADDR1	002174	\$ATYC	031070
MASS21	036705	M20	033661	RESREG=	104412	TBADDR	002176	\$ATY1	031044
MASS22	036740	M21	033670	RESVEC=	000010	TBIT	= 000010	\$ATY3	031052
MASS23	036772	M22	033707	RETURN	003060	TBITVE=	000014	\$ATY4	031062
MASS25	037033	M23	034064	ROTDAT	016452	TEMP	002232	\$AUTOB	001150
MASS26	037056	M24	034076	ROTFLG	002764	TEMP1	002242	\$BASE	001256
MASS27	037122	M25	034116	ROTPAT	002762	TEMP2	002766	\$BDADR	001136
MASS28	037125	M26	034135	RUMP	023152	TEMP3	002770	\$BDDAT	001142
MASS29	037210	M27	034167	R6	= %000006	TEMP4	002772	\$CDW1	001262
MASS3	035663	M28	034224	R7	= %000007	TEMP5	002774	\$CDW2	001264
MASS30	037253	M3	033342	SAVREG=	104411	TEXT	042631	\$CHARC	030610
MASS31	037321	M30	034334	SCOPE =	000004	TIOCM	007542	\$CMTAG	001114
MASS32	037375	M4	033345	SETBYT	025162	TKVEC	= 000060	\$CM3 =	000000
MASS33	037404	M5	033350	SETCLK	004336	TMOVEC	005044	\$CNTLG	027627
MASS34	037434	M5010	010570	SETPTN	025706	TPVEC	= 000064	\$CNTLU	027622
MASS35	037471	M5011	012216	SETRAM	024576	TRAPVE=	000034	\$CPUOP	001230
MASS36	037502	M5012	013046	STACK	= 001100	TRTVEC=	000014	\$CRLF	001177
MASS37	037637	M5013	010672	STARS	= *****	TSTADR	014132	\$DBLK	031030
MASS38	037703	M6	033353	START	004024	TSTBYT	025016	\$DDW0	001266
MASS39	037757	M6010	011334	STAT1	003000	TSTONE	025100	\$DDW1	001270
MASS4	035732	M6011	011720	STAT2	003002	TSTRGS	014224	\$DDW10	001312
MASS40	040022	M6011A	012112	STCO	023334	TST1	007542	\$DDW11	001314

\$DDW12	001316	\$ERRTB	003074	\$MADR4	001250	\$PWDRN	031716	\$TPS	001164
\$DDW13	001320	\$ERRTY	030176	\$MAIL	001202	\$PWRMG	032052	\$TRAP	031634
\$DDW14	001322	\$ERTTL	001126	\$MAMS1	001232	\$PWRUP	031770	\$TRAP2	031656
\$DDW15	001324	\$ESCAP	001174	\$MAMS2	001236	\$QUES	001176	\$TRP =	000013
\$DDW2	001272	\$ETABL	001222	\$MAMS3	001242	\$RDCHR	027364	\$TRPAD	031670
\$DDW3	001274	\$ETEND	001326	\$MAMS4	001246	\$RDLIN	027504	\$TSTM	001104
\$DDW4	001276	\$FATAL	001204	\$MBADR	001102	\$RDOCT	027656	\$TSTNM	001116
\$DDW5	001300	\$FFLG	031310	\$MFLG	031306	\$RDSZ =	000010	\$TTYIN	027612
\$DDW6	001302	\$FILLC	001172	\$MNEW	027645	\$RESRE	031576	\$TYPDS	030614
\$DDW7	001304	\$FILLS	001171	\$MSGAD	001216	\$RTNAD	006360	\$TYPE	030332
\$DDW8	001306	\$GDADR	001134	\$MSGLG	001220	\$SAVRE	031540	\$TYPEC	030544
\$DDW9	001310	\$GDDAT	001140	\$MSGTY	001202	\$SAVR6	032062	\$TYPEX	030612
\$DEVCT	001212	\$GET42	006336	\$MSWR	027634	\$SCOPE	027156	\$TYPOC	031336
\$DEVN	001260	\$HIBTS	001100	\$MTYP1	001233	\$SETUP=	000027	\$TYPON	031352
\$DOAGN	006356	\$HIOCT	030014	\$MTYP2	001237	\$STUP =	177777	\$TYPOS	031312
\$DTBL	031020	\$ICNT	001120	\$MTYP3	001243	\$SVLAD	027246	\$UNIT	001214
\$ENDAD	006346	\$ILLUP	032056	\$MTYP4	001247	\$SWR =	121000	\$UNITM	001110
\$ENDCT	006332	\$INTAG	001151	\$MUT	002166	\$SWREG	001224	\$USWR	001226
\$ENV	001222	\$ITEMB	001130	\$NULL	001170	\$SWRMK=	000000	\$VECT1	001252
\$ENVN	001223	\$LF	001200	\$NWTST=	000000	\$SW08T	027316	\$VECT2	001254
\$EOP	006302	\$LFLG	031307	\$OCNT	031534	\$TESTN	001206	\$XTSTR	027156
\$EOPCT	006324	\$LPADR	001122	\$OMODE	031536	\$TKB	001162	\$\$GET4=	000000
\$ERFLG	001117	\$LPERR	001124	\$OVER	027302	\$TKS	001160	\$\$SW08=	000024
\$ERMAX	001131	\$MADR1	001234	\$PASS	001210	\$TN =	000024	\$OFILL	031535
\$ERROR	030016	\$MADR2	001240	\$PASTM	001106	\$TPB	001166	.\$X =	001100
\$ERRPC	001132	\$MADR3	001244	\$POWER	032064	\$TPFLG	001173		

. ABS. 044717 000
ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 46808 WORDS (183 PAGES)
DYNAMIC MEMORY: 20740 WORDS (79 PAGES)
ELAPSED TIME: 00:03:13
CVPCAB/ENABLE:ABS:AMA,CVPCAB/-SP=SYSMAC/ML,CVPCAB.SRC